On Class Imbalanced Learning: Design of Non-parametric Classifiers, Performance Indices, and Deep Oversampling Strategies

Sankha Subhra Mullick
Electronics and Communication Sciences Unit
Indian Statistical Institute

A thesis submitted in partial fulfillment
of the requirements for the degree of

Doctor of Philosophy
in Computer Science

January, 2021

To my mother.

She taught me to think.
She taught me to question.

Acknowledgements

First and foremost, I would like to thank my supervisor Dr. Swagatam Das. I am immensely grateful to him not only for accepting me as his student but also for teaching me the very basic steps of independently conducting quality research. Starting from finding a problem to presenting the proposed solution in a well-written article, he was always there to help me by sharing his knowledge and experience. Without his invaluable insights and advises, this thesis could have never been a reality.

I would like to take this opportunity to thank Prof. Bhabatosh Chanda for his kind support during my Ph.D. course. I feel fortunate enough to be blessed by his guidance and encouragement throughout my evolution from a master's student to a researcher. I would also like to thank the rest of the faculty members of Electronics and Communication Sciences Unit, Indian Statistical Institute, Kolkata, for offering me their helpful suggestions. I like to express my gratitude to the staff of the Electronics and Communication Sciences Unit, the Dean's, and the Director's offices for their active support. I also like to thank my fellow research scholars, who over the years have became my treasured friends.

The people who have also played a key role in this thesis are my co-authors. Starting from my close friend and fellow research scholar Dr. Shounak Datta, to my dear students Nimagna Biswas, Saurajit Chakravorty, Imon Banerjee, and Sourish Gunesh Dhekane, each actively helped me in successfully carrying out the research works described in this thesis. I would love to take this opportunity to wish them the very best in their future endeavors.

I would also like to acknowledge Indian Statistical Institute for funding my research and Electronics and Communication Sciences Unit for providing the required infrastructure.

Last but not least, I like to thank my parents and friends; the people part of my immediate and extended family. I am immensely grateful to them for their selfless love, constant support, and encouragement through thick and thin.

Abstract

The relevance of classification is almost endless in the everyday application of machine learning. However, the performance of a classifier is only limited to the fulfillment of the inherent assumptions it makes about the training examples. For example, to facilitate unbiased learning a classifier is expected to be trained with an equal number of labeled data instances from all of the classes. However, in a large number of practical applications such as anomaly detection, semantic segmentation, disease prediction, etc. it may not be possible to gather an equal number of diverse training points for all the classes. This results in a class imbalance in the training set where some majority classes contain a significantly larger number of examples than the rest of the minority classes (usually corresponding to rare and important events). Consequently, a classifier trained in presence of class imbalance is likely to achieve better accuracy on the majority classes compared to the minority ones.

Class imbalance not only adversely affects the performance of a classifier but also leads to improper validation of its merit by inducing bias on the performance evaluation indices. We start by proposing a couple of fundamental conditions violation of which leads an index to be susceptible to an altering extent of imbalance and a varying number of classes in the test set. Under the light of these conditions, we present a theoretical study on the applicability of different indices commonly used to evaluate a classifier in the presence of class imbalance. Over the past couple of decades, a vast collection of research work attempted to modify the classifier and the training set respectively by algorithm-level and data-level approaches, such that the bias induced by class imbalance can be mitigated. We follow this direction of research by focusing on the popular Fuzzy-k-Nearest Neighbor (FkNN) classifier. We start by theoretically validating the quality of the class membership of a test point estimated by FkNN. We further demonstrate that our analysis can explain the susceptibility of FkNN to class imbalance and propose a point-specific locally adaptive class weighting strategy as a remedy. Moreover, we show that class-specific feature weights in addition to global class weights can significantly improve the immunity of FkNN against class imbalance when both types of weights are optimized using a self-adaptive variant of Differential Evolution. The advent of deep learning introduced another direction of research where attempts were made to understand the extent to which the commendable efficacy of the deep learning systems can be compromised in presence of class imbalance and propose remedial measures. We attempt to contribute in this direction by proposing an adaptive artificial oversampling technique that can be applicable to an end-to-end deep image classifier. Our model is constructed using three networks, a classifier, a convex generator, and a discriminator. An adversarial game between the classifier and the convex generator leads the latter to generate difficult artificial minority instances in the distributed feature space, while the discriminator adversarially guides the convex generator to follow the intended class distribution. As concluding remarks we discuss the future scope of research in combating the effects of class imbalance, especially in the emerging applications.

Contents

List of Figures

List of Tables

List of Notations

$\mathbb{S}$	A set of data points.
$\mathbf{s}$	A data point belonging to $\mathbb{S}$.
d	Dimension of data points belonging to $\mathbb{S}$.
S	Number of points belonging to $\mathbb{S}$.
c	Number of classes a data point can be classified into.
C	Set of c class labels i.e. $\{1, 2, \cdots, c\}$.
H (H_i)	A classifier (i-th output line estimating the posterior probability for the i-th class when the class labels are one-hot encoded).
X (X')	Training (test) set.
$\mathbf{x}$ ($\mathbf{y}$)	A training (test) point.
N (n)	Number of training (test) points.
$h(\mathbf{x})$	Original class label of the training point $\mathbf{x}$.
$\hat{h}^X(\mathbf{y})$	Class label of a test point $\mathbf{y}$ predicted by a classifier H which is trained on X.
N_i (n_i)	Number of training (test) samples belonging to the i-th class.
P_i	Prior probability of the i-th class.
IR_{pair} (IR)	Set of (maximum) imbalance ratio among all possible pairs of classes.
RRT_{pair} (RRT)	Set of (maximum) ratio of representatives in the test set among all possible pairs of classes..
Q_c ($\mathbb{Q}_c$)	A (set of) c-class confusion matrix.
q_{ij}	An element of Q_c representing the number of points from the i-th class which are predicted as members of the j-th class.
r_i	Sum of the i-th column of Q_c, i.e. total number of points predicted as members of the i-th class.
f	A classification performance evaluation index.
$\mathcal{V}_{Q_c}(f)$	Functional used to evaluate index f on Q_c.
$\mathcal{L}_{\mathbb{Q}_c}(f)$ ($\mathcal{U}_{\mathbb{Q}_c}(f)$)	Functional used to calculate the minimum (maximum) value of index f on set of all c-class confusion matrices.
$\mathcal{Q}_c(i)$	All those c-class confusion matrices where the classifier performs extremely poor on the i-th class.
$\mathcal{W}_{Q_{c(i)}}(f)$	Fractional calculating the limit of f when the performance of the classifier deteriorates on the i-th class.

γ_2 (γ_c)	GMean index for 2-class (c-class) classification problems.
ρ	Area under receiver operating characteristics curve.
ζ ($\hat{\zeta}$)	Precision index (modified).
α'	Recall index.
κ ($\hat{\kappa}$)	Area under Recall-Precision curve (modified).
β	Accuracy index.
α	Average class specific accuracy index.
ρ_o (ρ_a)	Extension of area under receiver operating characteristics curve for multi-class problems using one vs. one (one vs. all) strategy.
κ_a ($\hat{\kappa}_a$)	Extension of area under Recall-Precision curve for multi-class problems using one vs all strategy (modified).
$\bar{\rho}_a$	Weak lower limit for ρ_a.
k	Number of neighbors in k-Nearest Neighbor type classifiers.
$\mathbb{V}_k^X(\mathbf{y})$	Set of k nearest neighbors of $\mathbf{y}$ in X.
$\mathbf{v}_k$	The k-th nearest neighbor of $\mathbf{y}$ in X.
$\mathcal{I}$	Indicator function.
m	Parameter introduced in Fuzzy k-Nearest Neighbor classifier.
k'	Number of nearest neighbors considered for calculating the initial class memberships using Keller's heuristic.
$u_j(\mathbf{s})$	ground truth membership of point $\mathbf{s}$ to the j-th class.
ω_j (Ω)	Class specific weight (set of) for the j-th class.
$\hat{u}_j(\mathbf{y})$ ($\hat{u}_j^{\Omega}(\mathbf{y})$)	Membership of $\mathbf{y}$ to the j-th class as estimated by the (weighted) Fuzzy k-Nearest Neighbor classifier.
$\mathbb{R}$ ($\mathbb{R}^+$)	Set of all (positive) real numbers.
$\mathbb{Z}^+$	Set of all positive integers.
$\nabla u_j(\mathbf{s})$	Class specific gradients of the membership function with respect to the data point $\mathbf{s}$.
$\|\|.\|\|$	Euclidean norm.
p (p_j)	Distribution of the dataset (i-th class) $\mathbb{S}$.
E	Expectation.
v_d	Volume of a d-dimensional sphere of unit radius.
o (O)	Little (big) o notation.
$\mathcal{B}$	d-dimensional ball with a given center and radius.
Pr	Probability of an event.
b (A)	Temporary variables (constants).
μ_e (σ_e)	Sum of the biases (mean squared errors) of the Fuzzy k-Nearest Neighbor class membership estimator over all the c classes.
μ_+ (σ_+)	Bias (mean squared error) of the Fuzzy k-Nearest Neighbor class membership estimator over the minority class.
$\delta(\mathbf{y}, X)$	Distance of the nearest neighbor of $\mathbf{y}$ in the set of points X.
ν	The number of nearest neighbors to be considered during locally adaptive class specific weight calculation.

δ_{max} (δ_{min})	Maximum (minimum) nearest neighbor distance among a set of points.
ν_{max}, ν_{min}	Parameter introduced in locally adaptive class specific weighting strategy.
ν_{lin} (ν_{exp})	Estimation of ν using the linear (exponential) model.
$\Delta_{\Omega^{\mathcal{F}}}$	Class specific feature weighted Euclidean distance.
$\omega_{ij}^{\mathcal{F}}$ ($\Omega^{\mathcal{F}}$)	Weight (set of) for the i-th feature when the point from which the distance is being calculated belongs to the j-th class.
$\mathcal{J}$	An optimization problem.
Θ	Domain of an optimization problem $\mathcal{J}$.
$\mathbb{F}$, C_r	Parameters in Differential Evolution.
N_p	Population size in Differential Evolution.
θ ($\hat{\theta}$)	A candidate solution (optimum) of the optimization problem $\mathcal{J}$ belonging to the set Θ.
Θ_{N_p}	A population of N_p number of candidate solutions.
A_r, M, $\mathcal{M}$	Memory achieves in success history based adaptive Differential Evolution.
$\mathbb{H}$, τ	Parameters in success history based adaptive Differential Evolution.
$\mathcal{E}$ ($\mathcal{E}'$)	Error objective function used in the proposed PIFWkNN (PIFW^2kNN) classifier.
$\boldsymbol{\eta}$ ($\boldsymbol{\eta}'$)	Candidate solution of $\mathcal{E}$ ($\mathcal{E}'$).
$\mathbb{T}$ ($\mathbb{T}'$)	Set of candidate solutions $\boldsymbol{\eta}$ ($\boldsymbol{\eta}'$).
$\mathbb{T}_{N_p}$ ($\mathbb{T}'_{N_p}$)	Population of candidate solutions $\boldsymbol{\eta}$ ($\boldsymbol{\eta}'$).
G	A convex generator.
D	A discriminator.
F	Feature extraction network.
t	Conditional transient mapping unit.
g_i	Instance Generation Unit for the i-th class.
$\mathbf{z}$	Noise sample.
$\hat{J}$ (J)	Objective function used in the proposed Generative Adversarial Minority Oversampling (without discriminator) using cross-entropy loss.
L_H, L_D, L_G	Objective functions used in the proposed Generative Adversarial Minority Oversampling classifier using least square loss.
$p_i^{(g)}$	Generated distribution of the i-th class.
Λ_d, Λ_n, Λ_g	Batches of data and noise samples.
Y_d, Y_n, Y_g	Batches of class labels for data and samples.
$\bar{Y}_g$	One's compliment of Y_g.
$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma})$	Normal distribution with mean $\boldsymbol{\mu}$ and standard deviation $\boldsymbol{\sigma}$.

List of Abbreviations and Acronyms

kNN	k-Nearest Neighbor.
SVM	Support Vector Machine.
MLP	Multi-Layer Perceptron.
DBSCAN	Density-Based Spatial Clustering of Applications with Noise.
RUS	Random Undersampling.
TL	Tomek Links.
IR	Imbalance Ratio.
ACSA	Average Class Specific Accuracy.
CoNN	Condensed Nearest Neighbor.
SMOTE	Synthetic Minority Oversampling Technique.
EvO	Evolutionary Optimization.
DE	Differential Evolution.
SHADE	Success History based Adaptive Differential Evolution.
MO	Multi-objective Optimization.
PO	Pareto Optimal.
VAE	Variational Auto Encoder.
GAN	Generative Adversarial Network.
FkNN	Fuzzy k-Nearest Neighbor.
WFkNN	Weighted Fuzzy k-Nearest Neighbor.
LACW	Locally Adaptive Class Weighting.
PIFWkNN	Parameter Independent fuzzy class-specific Feature Weighted k-Nearest Neighbor.
PIFW^2kNN	PIFWkNN strategy in WFkNN and additional class weighting.
GAMO	Generative Adversarial Minority Oversampling.
MSE	Mean Squared Error.
AUROC	Area Under Receiver Operating Characteristics.
AURPC	Area Under Recall Precision Curve.
FPR	False Positive Rate.
OVO	One Versus One strategy.
OVA	One Versus All strategy.
RRT	Ratio of Representatives in the Test set.

FE	Fitness Evaluation.
TP	True Positive.
FP	False Positive.
TN	True Negative (TN).
FN	False Negative.
GA	Genetic Algorithm.
GCW	Global Class Weighting.
WSR	Wilcoxon Signed Rank test.
WRS	Wilcoxon Rank Sum test.
KH	Keller's Heuristic used in FkNN.
KDE	Kernel Density Estimation.
CM	Control Method for statistical tests.
CN	Classifier Network.
cGAN	conditional GAN.
cTMU	Conditional Transient Map-ping Unit.
IGU	Instance Generation Units.
GAMO\D	GAMO without additional Discriminator.
cDCGAN	Conditional Deep Convolutional GAN.
GAMO2pix	GAMO generated samples to realistic images.
CE	Cross Entropy loss.
LS	Least Square loss.
FID	Fréchet Inception Distance.

Chapter 1

Introduction to Class Imbalance

Summary

A classifier expects to be trained on an equal number of distinct labeled examples from all the classes. This enables the learner to enjoy an equal opportunity to learn about each of the classes and likely enable a similar performance on all of them. However, in many real-life applications, it is common for all events to not occur with equal probability. This in consequence increases the difficulty to gather examples for the classes corresponding to rare events whereas annotated representatives of commonly occurring events are available in abundance. Thus, during the formation of the training set an imbalance between the number of representatives for the different classes is often observed. Such a class imbalanced training set is likely to bias the classifier in favor of the majority classes containing a larger number of labeled instances while the performance deteriorates on the minority class suffering from the dearth of training points. This inspired the classical machine learning and the emerging deep learning communities to consider the problem of class imbalance as a long-standing challenge. In this introductory chapter, we first formally define the problem of class imbalance, discuss its relevance in detail, and then provide a brief survey highlighting the major research directions and key developments. In this chapter, we also discuss the research problems which motivated us to shape the objective of this thesis. Finally, we illustrate a road map of this thesis highlighting the primary contributions made by the rest of the chapters.

1.1 The problem of class imbalance: An overview

Over the past couple of decades, the world experienced substantial growth in research on communication and computation technologies, aiming to improve the quality of everyday life. The vastly improved commercial production facilities and the advent of modern services further helped the cause by providing the mass easy access to the newly invented technologies at an affordable cost. This resulted in a colossal increase of available multi-media data that require automated and efficient processing in real-time. For example, social media services which became a part of regular life over the past decade not only require to efficiently manage an influx of responses from billions of users around the world but also need to automatically detect and screen hate speech, offensive contents, or false information. Consequently, the world observed the ever-rising importance of machine learning; a research direction aiming to develop robust and scalable algorithms capable of performing complex data processing, information extraction, and fast crucial decision making of commendable quality compared to a human expert. The new millennium also witnessed an almost dramatic growth of the deep learning paradigm (LeCun et al., 2015; Goodfellow et al., 2016), which initially focused on a costlier group of machine learning algorithms attempting to mimic the neural information processing in the human brain. However, with the advent of new parallel computation technologies and accessibility to a large amount of data the highly efficient deep learning soon grew out to become a standalone research direction in its own right.

A machine learning algorithm depending on the learning strategy can be broadly categorized into three major groups (Duda et al., 2000). First, supervised learning, where a learner (for example, a classifier such as k-Nearest Neighbor (kNN) (Fix and Hodges Jr, 1951), Support Vector Machine (SVM) (Cortes and Vapnik, 1995), etc.) is tasked to approximate a function from a set of data instances to a set of possible labels using a set of annotated training examples. Second, unsupervised learning, where a learner (for example, a clustering algorithm such as k-Means (MacQueen, 1967), Density-Based Spatial Clustering of Applications with Noise (DBSCAN) (Ester et al., 1996), etc.) attempts to draw inference from a set of data instances in absence of a labeled set of training points. Third, semi-supervised learning (Silva et al., 2016), where a learner during training gets access to only a limited number of labeled examples in addition to a large pool of unlabelled data instances. In addition to these

three classical learning approaches deep learning introduced a set of new techniques such as self-supervised learning (Goodfellow et al., 2014) and weakly supervised (Zhou, 2018) to cater to the evolving nature of annotated data availability. to elaborate, in self-supervised learning the labels are considered as an intrinsic property of the data which can be approximated by carefully observing the unlabeled samples. Weakly supervised learning on the other hand relies on partially labeled or coarsely annotated training instances.

In this thesis, we solely focus on supervised learners, specifically classifiers. Thus let us begin by formally defining the problem of classification. Let $\mathbb{S}$ be a dataset containing S number of d-dimensional data points which can be classified into c predefined classes. Therefore, the task of classification can be described as finding a function H which maps from the set $\mathbb{S}$ of data instances to a set C of c class labels. For simplicity and without the loss of generality, we assume $C = \{1, 2, \cdots, c\}$ while considering the exceptions of some classifiers such as Multi-Layer Perceptron (MLP) (Rumelhart et al., 1986) which uses a one-hot encoding scheme to represent the class labels[1]. A classifier during training attempts to approximate $H : \mathbb{S} \rightarrow C$ using a training set $X \subseteq \mathbb{S}$ of N labelled examples. Hence, for all data instances $\mathbf{x} \in X$ the corresponding original label $h(\mathbf{x}) \in C$ must be known in advance and made available during training. Utilizing X a classifier is expected to correctly predict the class label $\hat{h}^X(\mathbf{y})$ of a new test point $\mathbf{y}$ belonging to the test set $X' \subseteq \mathbb{S}$ containing n unlabelled instances. A classifier is a data-driven modeling technique i.e. the performance is largely dependent on the quality of the training set. Ideally, a training set is expected to adhere to some regularity conditions set by the classifier as inherent assumptions. Violation of such conditions by the training set may result in improper learning and consequently an undesirable performance (Das et al., 2018). One such regularity condition is the assumption of an equal number of training examples from each of the classes to facilitate fair learning. However, in many real-life problems, it may not be always possible to respect such conditions. To elaborate, in applications such as credit card fraud detection (Pozzolo et al., 2014), medical

[1]In case of one-hot encoding a class label is expressed as a c-dimensional binary vector. Thus, a point belonging to the i-th class is labeled by a vector having 1 at the i-th dimension and 0 otherwise. Consequently, the set of classes labels C can be alternatively expressed as the standard basis of a c-dimensional coordinate space. The classification function in such a case is alternatively expressed as a mapping from the data space to a c-dimensional class space. Specifically, for a data point $\mathbf{s} \in \mathbb{S}$ the classification function outputs $H(\mathbf{s}) = \{H_1(\mathbf{s}), H_2(\mathbf{s}), \cdots, H_c(\mathbf{s})\}$, where H_i denotes the probability of $\mathbf{s}$ to belong to the i-th class. Thus, if $\mathbf{s}$ belongs to the i-th class then the ideal output (accurate prediction with the highest confidence) of the classifier should be $H_i(\mathbf{s}) = 1$ and $H_j(\mathbf{s}) = 0$ for all $j \in C \setminus \{i\}$ which matches the one-hot encoded label of $\mathbf{s}$.

diagnosis (Wahab et al., 2017), financial prediction (Sun et al., 2020), image segmentation (Bulò et al., 2017), object detection (Zhang et al., 2016b; Oksuz et al., 2020), etc. not all events corresponding to the different classes are observed with equal probability. For example, a credit card provider may find only a handful of fraudulent cases among a million daily transactions, while a disease prediction system may observe only a limited number of malignant tumor instances compared to a large number of benign cases. This in reality results in an imbalanced training set where some classes contain a larger number of labeled instances than the other ones. The classes which have an abundance of training instances are called majority classes while those with a scarcity of labeled examples are known as minority classes. A classifier that is susceptible to such class imbalance in the training set is likely to perform well on the majority classes while failing to achieve good accuracy on the minority classes (Kubat et al., 1997; Japkowicz, 2000). Let us assume that the i-th class contains N_i number of training samples in the training set X. Thus, the prior probability of the i-th class is denoted as $P_i = \frac{N_i}{N}$. We can now define class imbalance in a more formal manner as in Definition 1.1.

Definition 1.1. *A training set X is called class imbalanced if there exists a pair of distinct classes (i, j) such that $i, j \in C$, and $N_i \neq N_j$ or alternatively $P_i \neq P_j$.*

Moreover, to express the extent of class imbalance in a training set we need a measure to quantify it. Thus, in Definition 1.2 we describe the measure Imbalance Ratio (IR) as follows:

Definition 1.2 (Datta and Das (2018))**.** *For a 2-class classification problem the Imbalance Ratio (IR) is defined as the ratio of the number of points in the majority class to that of the minority class in the training set. Analogously, for a c-class classification problem IR is calculated as the maximum among all the pairwise IRs (represented by the set $IR_{pair} = \{\frac{N_i}{N_j} | i, j \in C; i \neq j\}$) among the c classes (i.e. $IR = \max IR_{pair}$). Therefore, X can be considered as imbalanced if $IR > 1$.*

To observe how such a class imbalance in the training set can adversely affect the performance of a classifier, let us consider the following illustrative example.

Example 1.1. *Let us consider a 2-class "two-moon" dataset, where the two interleaving classes namely Blue and Red are shaped in the form of horseshoes. We create a collection of*

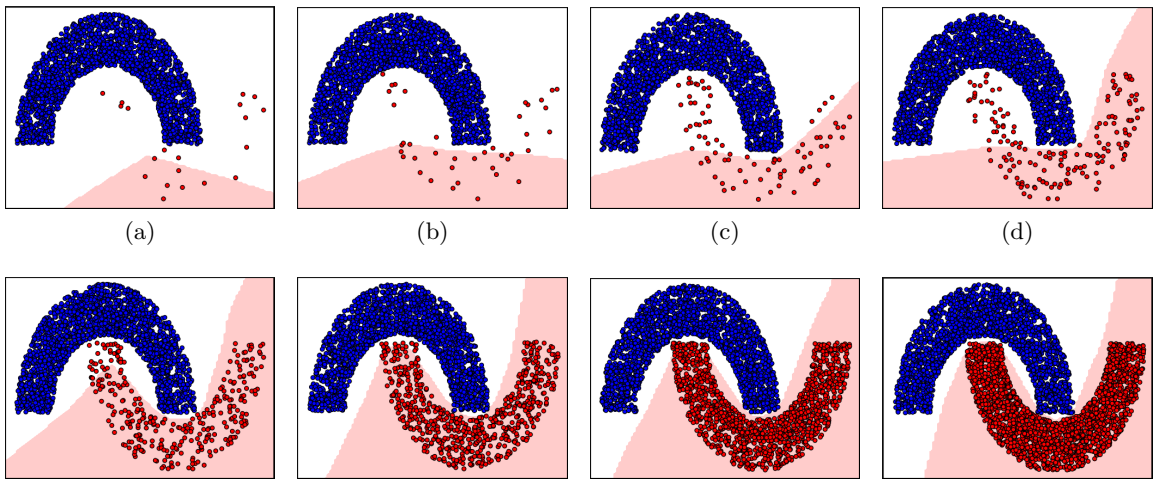


Figure 1.1: The effect of class imbalance on the decision regions (shown by pink and white respectively for the Red and the Blue class) learned by a MLP. (a) We train our classifier on the highly imbalanced training set having an IR of 100. The learned decision boundary correctly identifies only a small region of the entire Red class and consequently suffers a high misclassification on the minority instances. (b)-(g): In all these training sets the IR is gradually decreased by respectively setting it to 50, 25, 12.5, 6.25, 3.12, and 1.56. The quality of the learned decision boundary progressively improves even though some difficult regions of the minority class still remain misclassified. (h): The training set is balanced by sampling equal number of instances from both of the classes. The classifier accurately predicts both the classes and perfectly separates them with a non-linear decision boundary.

eight training sets each containing 2000 randomly sampled data instances from the Blue class. For the Red class, we gradually mitigate the extent of class imbalance over the eight training sets by sampling 20, 40, 80, 160, 320, 640, 1280, and 2000 data points. As a classifier, we choose the MLP (for easy visualization of the learned decision boundary) containing two hidden layers each having 16 hidden nodes. The outputs of the hidden layers are passed through a leaky rectified linear activation function with a leakiness of 0.1. Further, the network is trained using Adam (Kingma and Ba, 2015) optimizer with a learning rate of 0.0002. In all the cases, the training is performed for 5000 steps, where each step contains a batch of 32 examples. We train the classifier using each of the eight training sets (arranged in the order of decreasing class imbalance) and respectively plot the corresponding learned decision regions in Figures 1.1a-1.1h. In case of Figure 1.1a where the IR is maximum among the eight training sets (the minority class contains only 20 points compared to 2000 majority instances setting the IR to 100) the classifier demonstrates a significant bias towards the majority class

by misclassifying a large region of the minority Red class. From Figures 1.1b-1.1g we can see that the quality of the learned decision boundary progressively improves with increasing training examples from the minority class. However, even in Figure 1.1g, where the IR is minimum among the eight training set (IR is set to $\frac{2000}{1280} = 1.56$*) some difficult examples from the Red class still remain misclassified. Finally, in Figure 1.1h corresponding to the balanced training set, we can observe that both of the classes are correctly separated by the decision boundary. These empirical observations suggest that class imbalance during training has a significant impact on the performance of the classifier. Moreover, with an increasing imbalance between the classes, the performance on the minority class is likely to deteriorate further.*

It is observed from Example 1.1 that class imbalance may significantly deteriorate the accuracy of a MLP on the minority classes. However, minority classes usually correspond to the rare and significant events and thus, their accurate classification is often a crucial necessity. We can look back to the example of our credit card provider who needs to stress the correct identification of the less frequent fraudulent transactions to prevent malpractice. Further, a computer-aided diagnosis system needs to accurately segregate the fatal malignant tumors from the harmless ones to avoid treatment delay if not unfortunate fatalities. Thus, the challenge appears in the form of accurately classifying the minority class instances which are usually prone to misclassification due to their scarcity in the training set. Evidently, the problem of class imbalance gained sufficient attention from the machine learning community resulting in a plethora of approaches to efficiently combat its adverse effects concerning the performance on the minority class (He and Garcia, 2009; He and Ma, 2013; Branco et al., 2016; Krawczyk, 2016; Haixiang et al., 2017). The relevance of class imbalance only further increased with the advent of deep end-to-end learning systems (Johnson and Khoshgoftaar, 2019) directly handling complex data such as images, audios, videos, etc., and with the ever-growing importance of applications such as semantic segmentation, object detection, few-shot classification (Li et al., 2019), etc.

Subsequent to this overview in Section 1.1, rest of this chapter contains a characterization of the class imbalance problem detailing its effect on different datasets of varying properties in Section 1.2, brief surveys of approaches proposed to counter class imbalance in traditional

and deep classifiers respectively in Sections 1.3 and 1.4, the motivation and objective of this thesis in Section 1.5, and finally a road map of the subsequent chapters detailing their respective contributions in Section 1.6.

1.2 Characteristics of the class imbalance problem

Class imbalance is known to adversely affect the classification performance on the minority class. From Example 1.1 it is also observed that with increasing IR, the effect of class imbalance is likely to become more severe. However, the extent to which class imbalance can deteriorate the performance of a classifier is not easy to estimate. This is because the effect of class imbalance largely depends on data properties and the choice of the classifier itself. For example, let us consider a neural network classifier (López et al., 2013; Lin et al., 2017a) which minimizes a mean squared loss over all the training instances. In such cases, it is likely that the sum of small errors incurred by a large number of majority instances may overwhelm the sum of large errors for the small number of minority points. This, in effect, may lead the classifier to further stress on minimizing the loss over an already better classified majority class while ignoring the worse performing minority. In contrast, a SVM (Cortes and Vapnik, 1995) or kNN (Fix and Hodges Jr, 1951) type classifier which is otherwise independent of the total number of training samples may still suffer from class imbalance depending on data properties (Das et al., 2018) such as the presence of overlapped classes (Prati et al., 2004; García et al., 2006b; Alejo et al., 2013). Moreover, in large scale datasets, a classifier is expected to get access to a larger number of minority class training examples and thus likely to offer a better performance resisting the effects of class imbalance (Leevy et al., 2018; Zheng and Jin, 2020).

1.2.1 Overlapped classes and class imbalance

The presence of overlapped classes only worsens the effects of class imbalance. For example, let us consider an imbalanced training set where large portions of a dense majority class and a sparse minority class are overlapped. If we attempt to classify a minority test instance by the widely popular kNN classifier using such a training set, then it is highly likely that the neighborhood of the test point will contain a higher number of majority examples leading

to misclassification. The situation may also not improve for max-margin classifiers as the regularization error calculated over the overlapped section of the dataset is sensitive to the IR. However, even in the case of non-overlapping classes, the classification performance is likely to depend on the individual class distributions. We illustrate the effect of overlap on classification accuracy with the following 2-class classification problem in Example 1.2.

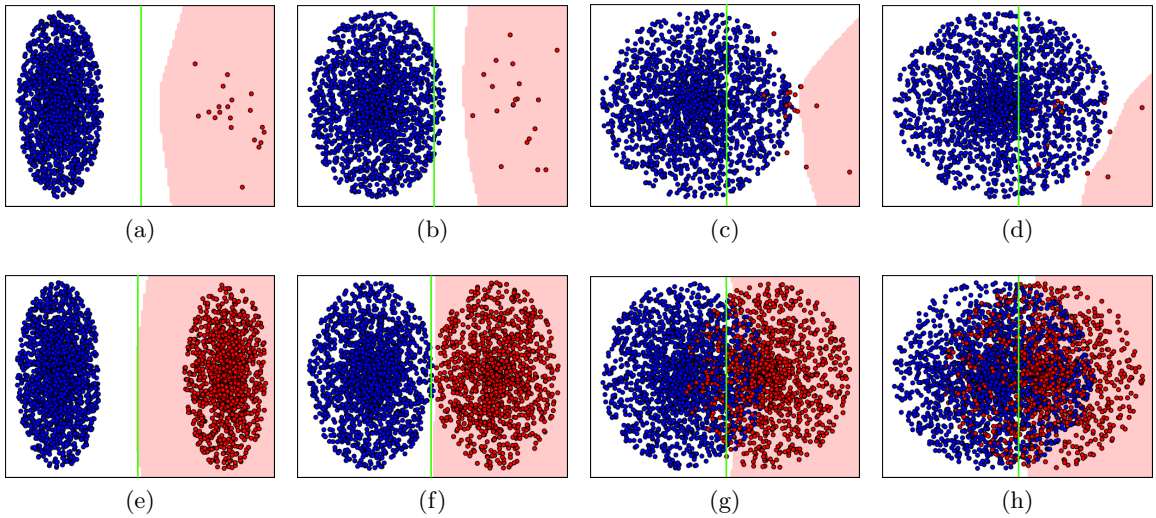


Figure 1.2: We use a 2-class dataset to illustrate how overlapped classes can worsen the impact of class imbalance. For (a)-(d) the IR is set to 100 by sampling 2000 instances from Blue and 20 examples from Red class. Whereas, for (e)-(h) the IR is improved to 1.56 by taking 2000 and 1280 samples respectively from Blue and Red class. To induce the effect of overlap the two classes are progressively drawn together over (a)-(d) and (e)-(h). We plot the decision regions of the trained MLP (pink for Red and white for Blue class) with the ideal boundary (green line). We can see that for (a)-(b) and (e)-(f) in absence of overlap the decision region for the minority class shifts further from the ideal with increasing IR. For (c) and (g) the classifier performs further deteriorates on the minority class with increasing IR. Finally, for very high overlapped cases (d) and (h) the classifier fails to achieve a commendable performance on both the classes.

Example 1.2. *We take a classification problem between two convex classes namely Blue and Red. Both classes are sampled from a ball of unit radius. The centers for the two classes are chosen such that they always lie at an equal distance from the line* $x = 0$*. Thus, the ideal decision boundary is* $x = 0$ *which is shown by the green line in Figures 1.2a-1.2h. To induce overlap, the classes are drawn closer together by shifting their centers by an equal amount towards the ideal decision boundary. We focus on two primary cases, namely a high IR of 100 and a low IR of 1.56, for each of which, we gradually increase the overlap from none to*

small, moderate, and high by decreasing the distance between the class centers to respectively 4, 2, 1, and 0.5 units. For all the cases we train a MLP (again chosen for easy visualization of the learned decision boundary) for 5000 steps using a batch size of 32 and plot the learned decision regions in Figure 1.2. The architecture of the classifier and the learning strategy are both kept similar to the one used in Example 1.1. We can see from Figures 1.2a-1.2b and 1.2e-1.2f that up to a small overlap, even though the classifier can accurately classify the minority instances with increasing IR the decision region becomes more biased towards the majority class. Moreover, in comparison to Figure 1.1a better performance on the Red class is observed in Figure 1.2a, albeit both of the training sets are non-overlapped with an equal IR. This indicates that even in the absence of overlap the individual class distributions may play a significant role behind controlling the impact of class imbalance on the classification performance over the minority class. Figures 1.2a-1.2d and Figures 1.2e-1.2h further suggest that accuracy over the minority class generally decreases with increasing overlap. Moreover, for moderate and higher degree of overlap, the performance is generally poorer on both of the classes while the minority class suffers further with higher IR as evident from Figures 1.2c-1.2d and Figures 1.2g-1.2h.

1.2.2 Varying scale of datasets and class imbalance

The advent of big data came with the opportunity to exploit the additional information obtainable from a larger number of total samples. As the probability of different events and consequently the IR remains a constant with increasing size of the training set, the minority class is likely to contain a larger number of examples. This, in turn, may aid the classifier to better learn the minority class distributions which results in improved performance. Therefore, with the increasing scale of data, the classifier is expected to learn a better decision boundary decreasing the generalization error (Fernández et al., 2017). We validate this intuition in the following Example 1.3.

Example 1.3. *In this example we consider three highly (IR is set to 100) and three low (IR is 1.56) imbalanced datasets previously used in Examples 1.1 and 1.2 which we respectively recall in Figures 1.3a-1.3f. To investigate the impact of data scale for each of the six datasets, we create a correspondingly large scale variant. For datasets with IR 100, we sample 20000*

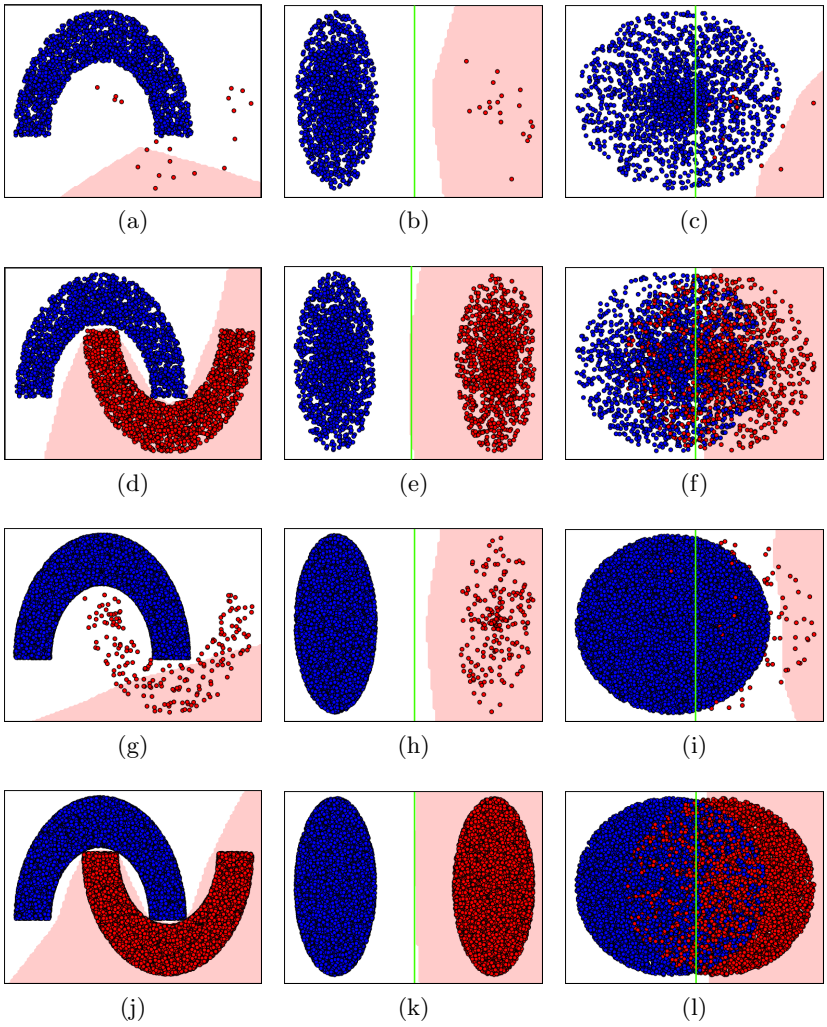


Figure 1.3: We illustrate how increased data scale may aid a classifier to better learn the minority class distribution in presence of class imbalance. (a)-(f): We recall the six datasets in Figures 1.1a, 1.2a, 1.2d, 1.1h, 1.2e, and 1.2h from Examples 1.1 and 1.2. (g)-(l): For each of the six dataset we create a corresponding larger variant by sampling 10 times more points from the two classes while retaining the original IR. We illustrate the decision regions (pink for Red and white for Blue class) learned by a multi layer pedestrian after 5000 steps and compare them with the boundary obtained for the corresponding smaller scaled dataset. From (a) and (g) we can see that with increased scale the classification performance on the minority class improves. This improvement becomes more apparent in (d) and (j) where the IR decreases from 100 to 1.56. The classification performances in (h) and (k) are respectively similar to (b) and (e) where the bias to the majority class mitigates with increasing scale and decreasing IR. Finally in (i) we can see that even after an increased scale from (c) the improvement on minority class is negligible. Similarly between (f) and (l) the classifier fails to noticeably improve its accuracy on both the classes.

points from the blue class and 200 examples from the Red class to obtain a higher scale dataset with an equal extent of imbalance. Further, when the IR of the dataset is 1.56 we select 20000 Blue points and 12800 Red points to increase the scale while maintaining the original ratio of representatives from the two classes in the training set. We train the same MLP classifier used in Examples 1.1 and 1.2 for 5000 steps on the large scale datasets and illustrate the learned decision regions in Figures 1.3g-1.3l. We can observe from Figure 1.3 that the classification accuracy on the minority class generally improves with increasing scale for non-overlapped datasets. Further, the increased scale is also likely to close down the gap between the learned and the ideal decision boundaries thus facilitating a better generalization to the test samples. However, for highly overlapped datasets, the improvement achieved by the classifier is negligible compared to the error it incurs on both the classes.

1.2.3 Class overlap, data scale, and class imbalance: A summary

We summarize the empirical findings of Sections 1.2.1 and 1.2.2 in Figure 1.4 to understand the interrelation between different extent of overlap, data scale, and class imbalance. We can empirically conclude that when IR is low both of the non-overlapped classes are likely to be classified with acceptable accuracy irrespective of data scale. However, even in such cases, the minority class may have slightly higher misclassification compared to the majority class. The increasing scale may also improve the generalization capability of the classifier. If the classes are highly overlapped and the IR is low then the minority class may suffer a high misclassification. On the other hand, if IR is high then in absence of overlap the performance on the minority class may largely depend on the corresponding class distribution. However, in high IR, if the overlapped between classes are increased then both of the classes are likely to suffer high misclassifications.

1.3 Approaches to counter the effects of class imbalance in traditional classifiers

The different approaches commonly used to offer immunity to a traditional classifier against class imbalance can be categorized into the following three groups (Krawczyk, 2016; Das et al., 2018).

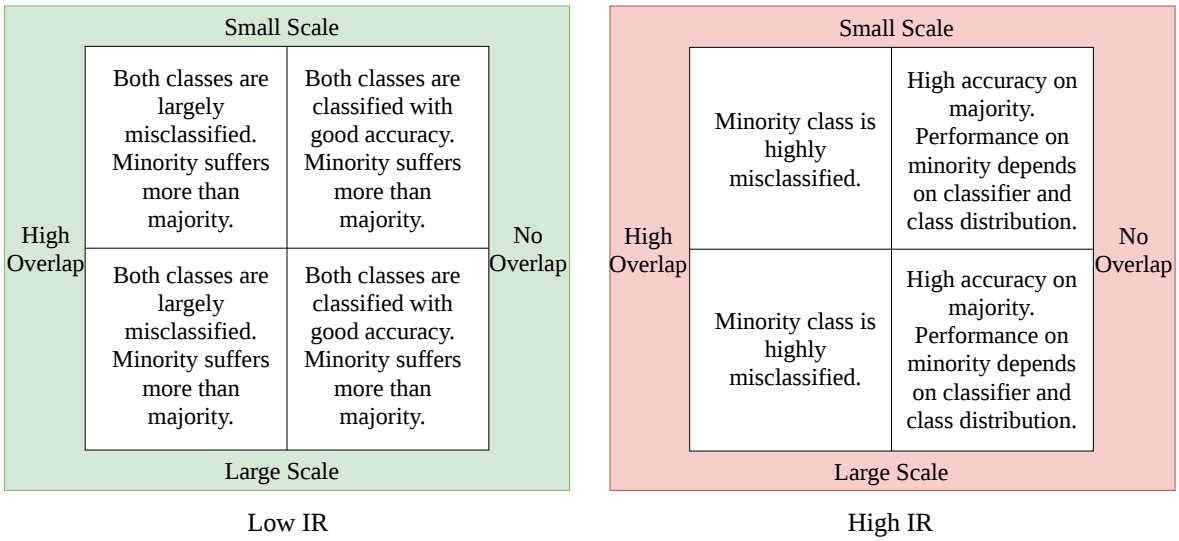


Figure 1.4: We graphically summarize the empirical findings of Sections 1.2.1 and 1.2.2 highlighting the interplay between class imbalance, overlapped classes and data scale. We consider two scenarios namely Low IR and High IR. For each of the cases we describe the likely classification performance with varying degree of overlap and differing scale of data.

- *Data-level techniques*: Here the training data itself is forcefully balanced by under-sampling (Japkowicz, 2000) the majority class oversampling the minority class (often by generating new artificial samples) (Chawla et al., 2002; Zhang and Li, 2014), or a combination thereof to mitigate the extent of class imbalance (Li et al., 2018a). Such approaches usually act as a pre-processing step before the actual training of the classifier. Data-level techniques can be considered more general as they do not rely on the choice of the classifier. However, such methods come with a risk of losing information and distorting the original class distributions.

- *Algorithm-level techniques:* Here the training data is kept unaltered while the algorithm is modified to account for the class imbalance. Such approaches primarily range over appropriate cost tuning (Ling and Sheng, 2008), boundary shifting (Imam et al., 2006), single class learning (Krawczyk et al., 2014b), active learning (Ertekin et al., 2007a), kernel specific methods (Maratea and Petrosino, 2011; Maratea et al., 2014), multi-objective optimization (Soda, 2011; Aşkan and Sayın, 2014), and classifier ensemble (Wang and Japkowicz, 2010). Among the less explored directions boundary shifting classifiers may also use additional cost tuning (Yang et al., 2009) or cost sensitivity can

be hybridized with active learning (Zhao and Hoi, 2013; Peng et al., 2020) to offer a better immunity against class imbalance.

- *Hybrid techniques*: Here data and algorithm level techniques are coupled together to ensure effective utilization of their individual advantages while compensating their shortcoming. Such approaches may focus on the hybridization of the resampling techniques with an ensemble of classifiers (Chawla et al., 2003; Seiffert et al., 2010) alongside cost-sensitive (Akbani et al., 2004) and active learning (Lee et al., 2015) based classifiers learned on resampled training sets.

1.3.1 Data-level techniques

In this section, we take a look at the different data level techniques namely undersampling, oversampling, and their combinations, which are proposed to mitigate the extent of class imbalance in the training set.

1.3.1.1 Undersampling techniques

Undersampling was initially introduced in machine learning for reducing the size of the training set to run kNN with a limited computational cost. The primary undersampling approaches include Condensed Nearest Neighbor (CoNN) (Hart, 1968) and Tomek Links (TL) (Tomek, 1976). Initially introduced for training set reduction both of the methods later found success in mitigating class imbalance as well. Interestingly, CoNN and TL employ contradictory philosophy to remove majority class training examples. CoNN prunes those majority points which have a nearest neighbor in the reduced training set sharing the same class label. Whereas, TL removes majority points which have a minority class nearest neighbor in the training set. Intuitively CoNN attempts to filter out those samples which do not play a significant role in learning the decision boundary. On the other hand, TL focuses on borderline samples that lie close to the class periphery and thus are likely to be misclassified with minute changes in the decision boundary. (Kubat et al., 1997) proposed a one-sided sampling where the concepts of both the CoNN and TL were integrated. Similar to CoNN, in one-sided sampling, an intermediate training set is created by all the minority class points, a single randomly chosen majority example, and all majority class instances which are not the

nearest neighbor of the chosen majority class point. The intermediate training set is further reduced by removing the TLs. Recently, the TL undersampling technique has been extended to further weed out the outliers, redundant, and noisy examples from the training set (Devi et al., 2017). The philosophy of TL has also been employed by Vuttipittayamongkol and Elyan (2020) while removing majority instances from the overlapped regions whereas Kang et al. (2017) did the same for CoNN by showing that for SVMs majority points lying at a distance from the decision boundary may safely be discarded. Besides these two traditional techniques and their variants, Japkowicz (2000) showed that a simple Random Undersampling (RUS) may also provide significantly improved performance by alleviating the effects of class imbalance.

Clustering based undersampling techniques (Yen and Lee, 2006, 2009; Peng et al., 2014; Ofek et al., 2017; Lin et al., 2017b; Tsai et al., 2019) were always regarded as a popular choice as the cluster centers act as a natural proxy for a collection of similar points. Clustering also helps to identify the different sub-concepts present inside a class (Stecking and Schebesch, 2012) as well as aids to better approximate the majority class distribution (Ng et al., 2014). Thus, over the years, numerous techniques utilized clustering to perform an efficient undersampling such that information of the majority class can best be preserved while balancing the training set. Density-based clustering algorithms such as DBSCAN (Ester et al., 1996) may also be proven beneficial for undersampling as in (Bunkhumpornpat and Sinapiromsaran, 2017), where it was applied to prune majority points lying in a dense overlapped region or in (Peng et al., 2014), where it was coupled with an additional kNN. Another popular approach is to use evolutionary (García et al., 2006a; García and Herrera, 2009; Ha and Lee, 2016) and meta-heuristic optimization techniques (Yu et al., 2013) to perform undersampling.

Among the other interesting directions of research, Wong et al. (2014) considered using a fuzzy rule based system (Ishibuchi et al., 1992) to perform undersampling. Fu et al. (2016) investigated the usefulness of Principal Component Analysis (Duda et al., 2000) to identify a set of majority class examples that can retain the characteristics of the corresponding class using a reduced number of samples. D'Addabbo and Maglietta (2015) designed an undersampling technique for SVMs that can be used in a parallel or distributed computing system. Koziarski (2020) proposed a radial-based sampling to reduce the computational cost. Recently, a meta-learning (Brazdil et al., 2008) based trainable undersampling technique has

been developed by Peng et al. (2019).

1.3.1.2 Oversampling techniques

Random oversampling of existing minority class points in an attempt to balance the training set has always been considered as a basic idea. However, such an oversampling technique does not provide additional information to the classifier aiding a better learning of the minority class distribution. Instead, random oversampling may lead to overfitting which deteriorates the performance of the classifier on the test set (Batista et al., 2012). The milestone development came through the proposal of SMOTE which attempts to oversample the minority class by generating new synthetic samples (Chawla et al., 2002). In SMOTE, given a minority point, a new artificial sample is generated as a convex combination of the original point itself and one of its randomly chosen minority neighbors. Following its introduction, it only took a short while for SMOTE to gain significant popularity due to its algorithmic simplicity, acceptable time complexity, easy parameter tuning, commendable performance, and theoretical validation (Elreedy and Atiya, 2019). Even though SMOTE offers an elegant solution to the oversampling problem it still has its limitation. Firstly, SMOTE can end up generating out of distribution points if the local convexity is not ensured over a small neighborhood. Secondly, SMOTE is not locality specific i.e. it cannot generate more points to aid the classifier in difficult to classify regions such as class peripheries or sparse locations. To date numerous studies attempted to resolve the shortcomings of SMOTE by proposing better performing variants, a survey of which can be found in (Fernández et al., 2018) marking the fifteenth anniversary of the technique. In Table 1.1 we list down some of the notable and recently proposed variants of SMOTE which highlights the importance of the technique in class imbalanced learning and demonstrates its relevance even to this day.

Apart from SMOTE and its variants, there are notable research works exploring the oversampling techniques to tackle the problem of class imbalance (Sáez et al., 2016). As a primary attempt (Lee, 2000) thought of generating new instances by adding noise to the existing minority training examples. Jo and Japkowicz (2004) attempted to relate the problems of class imbalance and small disjuncts (Holte et al., 1989; Ting, 1994; Weiss and Hirsh, 2000; Weiss, 2010) while proposing random oversampling inside the clusters of the majority and minority classes. This route was explored again in recent times by Tao et al. (2020). The effect of

Table 1.1: List of notable and recently developed variants of SMOTE

Algorithm	Comments
SMOTE+Tomek (Batista et al., 2004)	SMOTE generated points are cleaned by removing Tomek links.
SMOTE+ENN (Batista et al., 2004)	Oversampled training set is cleaned by removing points which are misclassified by a kNN with $k = 3$.
Borderline SMOTE (Han et al., 2005)	Minority samples near the borderline are oversampled.
ADASYN (He et al., 2008)	Difficult to classify minority class regions are oversampled.
MSMOTE (Hu et al., 2009)	Noisy samples are removed alongside SMOTE.
Safe-level SMOTE (Bunkhumpornpat et al., 2009)	Minority points which do not lie close to majority instances are oversampled.
LN-SMOTE (Maciejewski and Stefanowski, 2011)	Generalised variant of Borderline SMOTE (Han et al., 2005) and Safe-level SMOTE (Bunkhumpornpat et al., 2009).
SMOTE+FRST (Ramentol et al., 2012b)	Oversampled training set is cleaned by removing noisy points.
FRIPS+SMOTE (Verbiest et al., 2012)	Pre-processing removes the noisy instances.
MWMOTE (Barua et al., 2014)	New samples are generated in minority clusters which are hard to classify.
DBSMOTE (Bunkhumpornpat et al., 2012)	New instances are generated between a minority point and a pseudo center of a minority cluster.
FRIPS+SMOTE+FRBPS (Verbiest et al., 2014)	Post-processing filters the FRIPS+SMOTE (Verbiest et al., 2012) generated artificial samples.
Kernel-SMOTE (Mathew et al., 2015)	SMOTE is performed in the kernel space in SVM.
SMOTEIPF (Sáez et al., 2015)	A post-processing follows SMOTE to weed out potentially erroneous artificial points.
WK-SMOTE (Mathew et al., 2018)	Improved variant of Kernel-SMOTE (Mathew et al., 2015).
k-means SMOTE (Douzas et al., 2018)	Improve immunity of SMOTE against small disjuncts and noisy samples.
GSMOTE-NFM (Cheng et al., 2019)	Differing neighborhood size for safe, boundary, and outlier minority points.
G-SMOTE (Douzas and Bacao, 2019)	Geometric generalization of SMOTE.
SMOTEFUNA (Tarawneh et al., 2020)	The furthest neighbor is always chosen for generating new sample.
Adaptive-SMOTE (Pan et al., 2020)	Improved variant of ADASYN (He et al., 2008).
Eigen-SMOTE (Ye et al., 2020)	Improve performance of SMOTE variants in presence of noise and data scarcity.

random oversampling was also investigated by Menardi and Torelli (2014) who theoretically as well as empirically validated the efficacy of smoothed bootstrap re-sampling. In a differ-

ent direction, the effectiveness of evolutionary optimization techniques such as the Genetic Algorithm (Goldberg, 1989) to oversample the training set was explored by Cervantes et al. (2014); Maheshwari et al. (2011). To avoid the additional challenges posed by the overlapped regions between the classes Abdi and Hashemi (2015) proposed to oversample in the dense locations of the minority class while Pérez-Ortiz et al. (2015) attempted to oversample in a kernel induced feature space where the classes are linearly separated. A recent approach by Bellinger et al. (2015) used a denoising autoencoder (Vincent et al., 2010) to generate new artificial minority class examples. Such deep learning based methods are especially effective on the high-dimensional dataset as, unlike SMOTE they do not rely on Euclidean distance measure which is likely to fail with increasing dimensions (Aggarwal et al., 2001). In another avenue of research, attempts were made (Das et al., 2015; Sadhukhan, 2019; Kamalov, 2020; Sadhukhan and Palit, 2020) to somehow estimate the distribution of the minority class such that new artificial examples can be easily sampled. Such minority examples being sampled from a distribution approximating the original one are likely to prevent overfitting (Zhu et al., 2017). However, the effectiveness of such approaches depends on the quality of the estimated minority class distribution, which may not be accurate given the scarcity of minority samples. Further, the situation may become even more challenging with increasing data dimensions and complex class structures encouraging robust radial-based alternatives (Krawczyk et al., 2019). Recently, Zhang et al. (2019) noted that the processes of oversampling and training of the classifier should not be independent of each other. Instead, the two processes should progress simultaneously such that the artificial samples aid the learning process as per the classifier's requirement.

1.3.1.3 Hybrid sampling techniques

Instead of only undersampling the majority class or oversampling the minority class hybrid sampling approaches attempt to perform both. This aids such techniques to avoid a significant loss of data or an increased risk of overfitting while balancing the training set. Among the notable attempts (Ramentol et al., 2012a) employed a rough set based pruning with SMOTE oversampling, Wang (2014), Richhariya and Tanveer (2020), and Jian et al. (2016) focused on SVM specific approaches, while Prachuabsupakij (2015) attempted to couple a clustering based undersampling with SMOTE. An alternative direction performs undersampling

through cleaning the majority class by removing noisy samples (Kang et al., 2016) or outliers (Junsomboon and Phienthrakul, 2017) before applying oversampling often by SMOTE. Such cleaning may also be applied to minority class in an attempt to improve the quality of artificial points generated for oversampling (de Morais and Vasconcelos, 2019). Another direction proposes to balance the dataset by switching the labels of the majority class examples (Stefanowski and Wilk, 2007, 2008; Błaszczyński et al., 2010; Gónzalez et al., 2017; Fernandes and de Carvalho, 2019; Gutiérrez-López et al., 2020) undersampling the majority and oversampling the minority classes in the process.

1.3.2 Algorithm-level methods

Let us now take a look at the various algorithm-level approaches where the original training data is retained while the classifier is modified to account for the class imbalance. Even though the implementation may be classifier-specific as the classification strategy is needed to be considered explicitly for modification, most of the algorithm-level techniques follow a general set of underlying strategies to combat class imbalance such as cost-sensitive or boundary shifting. However, there can also be techniques that are only applicable to a certain family of classifiers such as kernel-based methods for SVMs (Maratea and Petrosino, 2011).

1.3.2.1 Cost sensitive methods

Traditionally, one of the most common approaches to counter the effects of class imbalance is to consider the different cost of misclassification for distinct classes (Palacios et al., 2014). Thus, in cost-sensitive classifiers, class imbalance is compensated by considering a lower misclassification cost for the majority class than that for the minority class. Designing a cost-sensitive classifier involves answering a couple of primary questions. First, how should disparate class-specific costs be incorporated in the classifier under concern? Second, how should one find the set of optimal class-specific costs? The answer to the first question depends on the classifier itself. For example, in a neural network cost sensitivity may be realized by altering the class probability calculation, adjusting the output of the network, adapting the learning rate, or modifying the loss function, and accordingly change the backpropagation of error Kukar et al. (1998); Oh (2011). In the case of decision trees (Quinlan, 1987; Ho, 1995),

differing cost of misclassification can be considered while calculating the information gain or selecting the splitting attribute (Sahin et al., 2013; Lomax and Vadera, 2013). Whereas, in the case of SVM, the opportunity of incorporating cost sensitivity is explicitly involved in the formulation (Cao et al., 2013a; Katsumata and Takeda, 2015). The second question is commonly answered by parameter tuning (Arar and Ayan, 2015), using the class priors (Tan, 2005; Castro and Braga, 2013), or optimizing the costs in a self-adaptive manner (Zhang et al., 2018a). In the following Table 1.2, we list down the notable cost-sensitive variants of classifiers such as artificial neural networks (Haykin, 2009), decision tree (Quinlan, 1987) and random forest (Ho, 1995), SVMs, Bayesian network (Pearl, 1988), and deep belief network (Hinton et al., 2006).

Table 1.2: List of notable cost sensitive classifiers

Base classifier	Cost sensitive variants
Neural network	Kukar et al. (1998), Zhi-Hua Zhou and Xu-Ying Liu (2006), Alejo et al. (2007), Pendharkar (2008), Oh (2011), Bertoni et al. (2011), Castro and Braga (2013), Cao et al. (2013b), Ghazikhani et al. (2013b), Arar and Ayan (2015), Biswas et al. (2017)
Decision tree and random forest	Drummond and Holte (2000), Sheng and Ling (2006), , Liu et al. (2010), Sahin et al. (2013), Lomax and Vadera (2013), Boonchuay et al. (2017), Siers and Islam (2018), Guermazi et al. (2018), Zhu et al. (2018)
Support vector machine	Veropoulos et al. (1999), Li et al. (2010), Masnadi-Shirazi and Vasconcelos (2010), Duan et al. (2014), Cao et al. (2013a), Datta and Das (2015), Katsumata and Takeda (2015)
Bayesian network	Jiang et al. (2013), Jiang et al. (2014)
Deep belief network	Zhang et al. (2016), Zhang et al. (2018a)

1.3.2.2 Boundary shifting methods

In class imbalanced problem the decision boundary is likely to lie at a distance from the majority class and close to (if not pass through) the minority class. Boundary shifting classifiers attempt to rectify this issue by shifting the learned boundary away from the minority class such that the classification performance can be improved. In comparison to cost-sensitive methods, boundary shifting classifiers modify the decision boundary post-learning, even though both approaches require a set of properly tuned class-specific costs. Boundary shifting is intuitively appealing for kNN type classifiers as being a lazy learner they do not involve an explicit learning phase which minimizes misclassification error giving a chance to employ cost

sensitivity. Thus, an alternative that is more in line with the kNN decision strategy is to consider the distinct contributions of training points from different classes (Mani and Zhang, 2003; Tan, 2005; Wang et al., 2008; Liu and Nakagawa, 2001; Dubey and Pudi, 2013; Liu et al., 2018). Boundary shifting in kNN can also be achieved by minimizing the distance between minority training points (Li and Zhang, 2011) and a test instance, or by considering a dynamic neighborhood containing a sufficient number of minority examples while making a decision (Zhang and Li, 2013; Zhang et al., 2017). Boundary shifting can also be used with SVMs by providing higher weight to minority class (Imam et al., 2006) or by shifting the learned decision boundary (Lin et al., 2002). Moreover, for decision trees boundary shifting may be inherently implemented through the splitting criteria (Cieslak and Chawla, 2008) without additional cost tuning.

1.3.2.3 Single class learning methods

A single or one-class classifier (Raskutti and Kowalczyk, 2004) as the name suggests is designed to predict samples from a single target class. Unlike the traditional binary classifiers, single class ones are trained only using the labeled samples from the target class. Thus, such classifiers are naturally immune to the effects of class imbalance (Krawczyk and Woniak, 2014; Krawczyk et al., 2014b,a). Interestingly, the superiority of one class classifiers compared to the binary classifiers employing remedial measures against class imbalance (such as resampling) has always been a matter of debate. On one hand, studies like (Japkowicz, 2000) vouched for binary classifiers trained on a resampled training set. On the other hand, Bellinger et al. (2012) contradicted the previous findings. Recently, Bellinger et al. (2017) attempted to shed some light on the topic by identifying the cases where each of the two types of classification strategies may be useful. As per their findings, one-class classifiers are likely to perform well on the unimodal class structure while binary classifiers tailored for handling class imbalance may outperform on complex data. However, one class classifier may still suffer on minority class due to the general scarcity of available training examples. Moreover, if the classification problem contains more than one minority class, then such methods may not be beneficial (Krawczyk et al., 2014b).

1.3.2.4 Active learning methods

Creating a training set is a costly process as it not only involves gathering of data but also the task of properly labeling them by a set of experts. In an attempt to cut down the cost one cannot compromise by gathering less data as that will restrict the availability of information during training. Thus, the only opportunity to make the process economic is to reduce the cost of labeling. This is where active learning finds its relevance. Active learners start with a limited set of labeled samples alongside a large pool of unlabelled samples. During training, the learner can query an oracle about the label of an unlabelled point and add that to the training set as per necessity. In a class imbalanced problem minority class does not contain a large number of labeled training instances. Thus, an active learner which can learn with limited data is likely to be effective in such scenarios (Ertekin et al., 2007b,a; Zhu and Hovy, 2007; Attenberg and Provost, 2010; Attenberg and Ertekin, 2013; Khanchi et al., 2017; Lin et al., 2018). Active learning may also be hybridized with cost-sensitive learning (Zhao and Hoi, 2013; Peng et al., 2020) or aided by resampled training set (Zhu and Hovy, 2007) to better handle class imbalance. One can further improve the active learning framework by modifying the selection strategy of unlabelled instances which may also help the classifier in the presence of class imbalance (Ferdowsi et al., 2013; You et al., 2014). Though efficient active learners do face the question of implementing an oracle in practice, which upon query will assign a label to an unlabelled data point (Zhang et al., 2014a; Guo and Wang, 2015; Zhang et al., 2016a).

1.3.2.5 Kernel-based methods

This class of SVM specific methods mainly apply perturbation to Radial Basis Function kernel such that the resolution (magnitude of an infinitesimal vector in the kernel induced feature space) (Amari and Wu, 1999) around the minority points can be magnified (Wu and Chang, 2003, 2004, 2005; Maratea and Petrosino, 2011; Maratea et al., 2014; Zhang et al., 2014b). However, such methods involve a costly tuning process for the different resolution parameters which, if not done carefully, may lead to overfitting. Further, even though these methods are iterative in nature, there is no guarantee that the performance will monotonically increase over iterations. Alternatively, (Zhao et al., 2011) attempted to employ a data-dependent

kernel that can condense the minority class in the feature space.

1.3.2.6 Multi-objective optimization based methods

In Multi-objective Optimization (MO) techniques (Marler and Arora, 2004) the problem is expressed in the form of simultaneous optimization of two or more objective functions that are in conflict with each other i.e. improving one will push the others away from their corresponding optima. Evidently, in MO it is not possible to find a single feasible solution that simultaneously optimizes all the contradicting objectives under concern. Instead, there exists a set of solutions called Pareto Optimal (PO). In a PO solution, no objectives can be further optimized without diminishing any of the others. Thus, each of the possibly infinite numbers of PO solutions offers a trade-off between different objectives and it depends on the user to select a solution that best suits their requirement.

In the context of class imbalance, a natural implementation is to define the objectives as individual class-specific accuracies (García et al., 2010; Aşkan and Sayın, 2014; Wang et al., 2014; Maheta and Dabhi, 2015). However, with the growing number of classes, the complexity of the problem also increases exponentially. The problem becomes even more difficult if additional objectives are considered (Bhowan et al., 2013, 2014). This issue can be resolved by using performance evaluation indices which aggregates the class-specific accuracies (Ducange et al., 2010). This approach was also followed by Soda (2011) who attempted to optimize Accuracy (Japkowicz, 2006) alongside GMean (Kubat et al., 1997) two indices which are likely to conflict in the presence of high imbalance. This is because in highly imbalanced classification problems minimizing misclassification on only the majority class is enough to improve Accuracy while GMean only improves when the classifier performs equivalently well on all the classes. Similarly, (Chira and Lemnaru, 2015) proposed optimizing Recall (accuracy on the minority class) and Precision (fraction of the correctly classified minority class points and the total number of points predicted as minority class instances) for two-class classification problems. Alternatively, MO can be used in class imbalanced learning for data pre-processing such as feature and instance selection (Fernández et al., 2015a), balancing an imbalanced training set (Li et al., 2018b), or discretization (as required by the learner) (Tahan and Asadi, 2018). MO can also be employed in a classifier ensemble framework designed to withstand the effects of class imbalance (Yang et al., 2020).

1.3.2.7 Classifier ensemble methods

This family of techniques attempts to improve classification performance in the presence of class imbalance by an ensemble of cost-sensitive (Nikolaou et al., 2016) and boundary shifting (Nikolaou and Brown, 2015) component classifiers. Such techniques may choose to modify AdaBoost (Freund et al., 1999; Rätsch et al., 2001) type boosting (Schapire, 1990) based ensemble approaches to account for class imbalance (Joshi et al., 2001; Sun et al., 2007; Song et al., 2009). In AdaBoost, a set of component weak learners is sequentially built such that each learner stresses on accurate prediction of previously misclassified training points with the help of a point specific weighting scheme. The final prediction is made by weighted voting of the component classifiers where the better performing learners get higher weights. As the minority points are prone to high misclassification they are expected to get higher weights in the later rounds stressing the weak learners to perform well on them. Thus, boosting type ensemble approaches are likely to offer a naturally better immunity against class imbalance (Seiffert et al., 2008). Evidently, in such an ensemble approach itself, there are two opportunities for incorporating cost sensitivity. First, in the point specific weighting scheme as the cost of misclassification is not equal for all the classes (Ting, 2000; Viola and Jones, 2002). Second, in the component weighting strategy (Leskovec and Shawe-Taylor, 2003; Wang and Japkowicz, 2010) guided by an error function which may not always be suitable for class imbalanced problems (Sokolova et al., 2006; Japkowicz, 2006). In the most common direction, both of these components and ensemble level opportunities for improvement are explored simultaneously. Such techniques use an ensemble of cost-sensitive classifiers employing a modified component specific set of weights or a suitable loss function to counter against the effects of class imbalance by considering the differing cost of class-specific misclassification (Fan et al., 1999; Masnadi-Shirazi and Vasconcelos, 2007, 2010; Ghazikhani et al., 2013a; Krawczyk et al., 2013; Krawczyk et al., 2014c, 2015; Tao et al., 2019; Wong et al., 2020). Alternatively, the ensemble method can also be refined by considering only the confident classifiers during prediction to reduce computational cost (Fan et al., 2002) or appropriately modified to account for class imbalance (Tan et al., 2003; Xiao et al., 2012).

1.3.3 Hybrid approaches

Data-level techniques contain the risk of information loss or overfitting while algorithm-level methods require complex classifier specific modifications. Hybrid approaches integrate the two paradigms such that the best of both can be efficiently utilized with ease. A natural direction of such techniques is to use common oversampling and undersampling methods in ensemble approaches such as bagging and boosting (Galar et al., 2011). Thus, notable methods from this family can be broadly classified into three groups all coupling resampling with boosting (Bao et al., 2016; Kozlovskaia and Zaytsev, 2017; Tang and He, 2017), bagging (Barandela et al., 2003; Tang et al., 2008; Blaszczyński and Stefanowski, 2015), and other ensemble techniques (Sobhani et al., 2014; Lim et al., 2017; Zhang et al., 2018c). In the following Table 1.3 we list down the notable attempts in this direction.

Table 1.3: List of notable and recently developed approaches hybridizing resampling with classifier ensemble.

Algorithm	Comments
Boosting based approaches:	
SMOTEBoost (Chawla et al., 2003)	SMOTE based oversampling with boosting.
JOUS-Boost (Mease et al., 2007)	Hybrid sampling with boosting.
RAMO-Boost (Chen et al., 2010)	Adaptive minority oversampling with boosting.
RUSBoost (Seiffert et al., 2010)	RUS with boosting.
EUS-Boost (Galar et al., 2013)	Evolutionary undersampling with boosting.
Thanathamathee and Lursinsap (2013)	Boundary sample generation with boosting.
Select-Boost (Prusa et al., 2016)	RUS with feature selection and boosting.
PBoost (Soleymani et al., 2018)	Undersampling with boosting.
WOTBoost (Zhang et al., 2019)	Weighted oversampling with boosting.
Bagging based approaches:	
SMOTE-Bagging (Wang and Yao, 2009)	SMOTE based oversampling with bagging.
RBBagging (Hido et al., 2009)	Undersampling with bagging.
UBBagging (Liang and Cohn, 2013)	Unevenly balanced bagging.
DTE-SBD (Sun et al., 2018)	SMOTE with ensemble of decision trees based on bagging.
MIBag (Razavi-Far et al., 2019)	Multiple imputation based oversampling with bagging.
Other ensemble based approaches:	
EUS-SVM (Kang and Cho, 2006)	Undersampling with ensemble of SVMs.
EnSVM (Liu et al., 2006)	Oversampling with ensemble of SVMs.
Rıo et al. (2014)	Resampling for big data with random forest.
Peng (2015)	Adaptive sampling with cost tuning and classifier fusion.
Sun et al. (2015)	Balancing by resampling with classifiers ensemble.
DeepBalance (Xenopoulos, 2017)	Balanced bootstraps with ensemble of deep belief networks.
Roy et al. (2018)	Oversampling with dynamic ensemble.
HUS-Boost (Popel et al., 2018)	TL and RUS based undersampling with a soft voting ensemble.
REMDD (Niu et al., 2020)	Undersampling with majority voting of candidate classifiers.

In another direction, hybrid approaches explored the coupling of cost-sensitive (Krawczyk, 2016), boundary shifting (Yang et al., 2009; Cieslak et al., 2012), and active learning (Niko-

laou and Brown, 2015) based imbalanced classifiers with resampling techniques to achieve better performance. Among these three avenues, most of the approaches follow the route of integrating a cost-sensitive classifier with resampled training set (Akbani et al., 2004; Wang et al., 2012; Hsu et al., 2015). Such approaches first attempts to mitigate the extent of class imbalance in the training set by resampling and then train a cost-sensitive learner on them. Using an empirical study on a large number of imbalanced datasets of diverse characteristics López et al. (2012) concluded that even though cost tuning and resampling individually perform somewhat equivalently their hybridization is capable to offer a significant performance boost. Intuitively, such techniques do not need to perform an excessive amount of over and undersampling which aids to restrict information loss and tendency to overfit. Further, the cost-sensitive classifier is likely to be trained on a nearly balanced training set which facilitates an easy tuning of the class-specific misclassification costs while avoiding overcompensating.

1.4 Class imbalance in deep learning

Even though deep learning based classifiers managed to provide a significant improvement in performance over their traditional counterparts (Krizhevsky et al., 2012; Szegedy et al., 2015, 2016; He et al., 2016) they are no less susceptible to the class imbalance in the training set (Buda et al., 2018; Johnson and Khoshgoftaar, 2019; Oksuz et al., 2020). A key reason behind this vulnerability is the working strategy of neural networks which learn by minimizing a loss calculated over all the training points. Therefore, in the presence of class imbalance, the sum of small errors over a large number of majority instances are likely to dominate the sum of large errors obtained over a small number of minority points. Hence, reducing the errors over the majority points will be more beneficial for minimizing the loss function which in consequence will lead the network to learn the majority class even better, while ignoring the minority (Lin et al., 2017a). Fortunately, even before the advent of deep learning (LeCun et al., 1989), the impact of class imbalance was already well-known to the machine learning community. Thus, alongside developing new deep learners the research community did not waste time improving their immunity against class imbalance (Huang et al., 2016). This primarily appeared as a less strenuous task as attempts were made to directly import the common solutions available for traditional classifiers into deep learning paradigm (Levi and

Hassner, 2015; Jaccard et al., 2017). But the limitation of this direction was soon recognized for the deep learners employing an end-to-end strategy. For example, the widely appreciated rich pool of SMOTE type oversampling schemes was found ineffective in deep learning due to the following reasons.

1. SMOTE type oversampling approaches create new samples in the feature space. However, in end-to-end deep learning, the feature extraction and classification tasks are not mutually independent. In fact, the feature extractor sits on top of the classifier and both are trained simultaneously by back-propagating the single loss calculated on the classifier's output. Therefore, introducing SMOTE into this framework becomes difficult.

2. In SMOTE one needs to find a neighboring point based on some distance metric while the new sample is generated by a convex combination of existing instances. However, deep learners can directly be applied to natural non-vector datasets containing images, videos, and audios. In such data, not only the notion of conventional distance is undefined, but also a convex combination of the samples does not ensure a realistic output. This can further be illustrated in the following Figure 1.5a where we show that in case of Fashion-MNIST dataset (Xiao et al., 2017), an image of T-shirt/top lies closer to a shirt than another T-shirt/top in terms of Euclidean distance, even though the two T-shirts/tops clearly appear more similar to us. Further, in Figure 1.5b, we can observe that the convex combination of two images of the digit "9" from the MNIST dataset (LeCun et al., 1998) does not result in any realistic digit let alone "9".

Thus, new directions emerged where not only the applicability of traditional solutions were investigated but also new deep learning specific techniques were developed. Let us now take a look at the various approaches to combat the adversarial effects of class imbalance in deep learning.

- *Data-level methods:* A primary direction imported from the literature of traditional classifiers is random undersampling. However, a deep learner usually involves a large number of parameters (often in the range of millions), proper training of which requires a significant amount of data. Thus, reducing the amount of available data by random

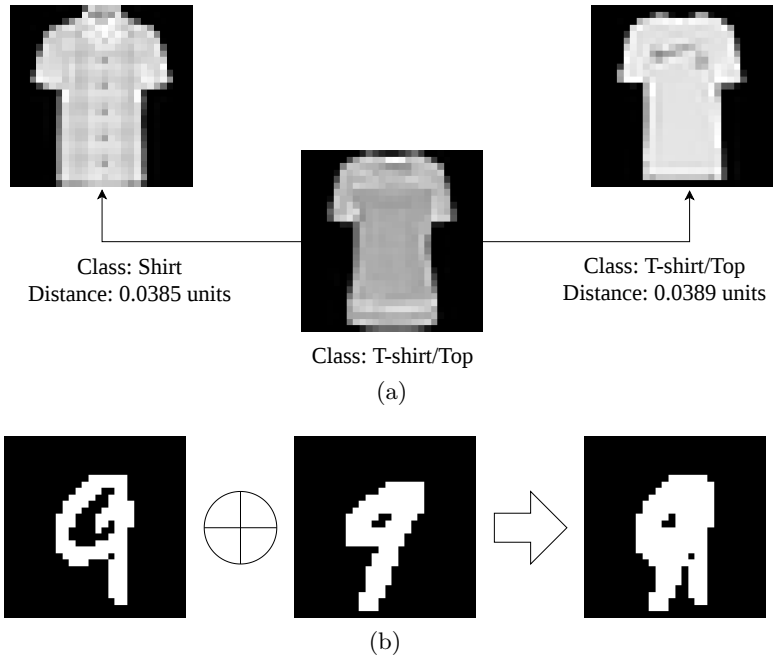


Figure 1.5: SMOTE is inapplicable on non-vector natural data such as images, videos, audios, where realism is a key constraint. (a) SMOTE requires to find the neighbors of a training example in terms of some distance measure. Unfortunately, in images the notion of distance is not defined and thus, neighbors in terms of Euclidean distance may not be meaningful in reality. We can see in Fashion-MNIST dataset a shirt lies closer to a T-shirt/top than another T-shirt/top even though visually the two T-shirts/tops are clearly similar than the T-shirt/top and shirt pair. (b) We show that convex combination of two images does not necessarily result in a realistic similar image. Here, the two images of the digit "9" taken from the MNIST dataset do not generate a new realistic "9" by convex combination.

undersampling can be considered counter-intuitive in deep learning. As a remedy, a two-phase learning is proposed where transfer learning is used to compensate for the information loss by undersampling (Lee et al., 2016; Havaei et al., 2017; Pouyanfar et al., 2018; Johnson and Khoshgoftaar, 2019b). Recently, a subspace clustering based technique was developed by You et al. (2018) which can ensure that the entire dataset can be represented through a subset of carefully selected samples resulting in an efficient undersampling.

The most common data level approach to mitigate the effect of class imbalance is to use a simple random oversampling of the minority class (Levi and Hassner, 2015; Jaccard et al., 2017). The advantage of this technique is one does not need to worry about the validity of the added samples as they are taken from the dataset itself. Manually

balancing the dataset also ensures that the inherent bias of the loss function to the majority class becomes ineffective without any complex modification. However, random oversampling does not add any new information to the dataset as well as may lead to overfitting. A straightforward solution can be replacing random oversampling with data augmentation (Afzal et al., 2019) a popular tool to address the general scarcity of data in deep learning systems (Shorten and Khoshgoftaar, 2019). Alternatively, Liu et al. (2019) came up with a fuzzy logic based synthetic minority sample generation technique while (Ando and Huang, 2017) attempted to venture the usefulness of meta-learning to perform oversampling in the deep learning paradigm. Recently, Wang et al. (2020) proposed a deep generative model that considers a joint distribution of input samples, their corresponding labels, and the latent variables relating the two.

The introduction of deep generative models namely Generative Adversarial Network (GAN) (Goodfellow et al., 2014) resulted in a breakthrough in oversampling strategies to combat class imbalance. A GAN network is composed of a couple of sub-networks, namely a generator and a discriminator. The generator to tasked to learn a mapping from points sampled from a known noise distribution to a data instance belonging to the actual data distribution. The discriminator adversarially guides the generator by checking if a generated data instance can be discriminated from the real training samples. In essence GANs attempt to minimize the divergence between the generated and the real data distribution. GANs can easily be extended for individual classes by including the class information as a condition (Mirza and Osindero, 2014). Soon after its introduction, GANs became widely popular for its simple architecture and efficient performance. This inspired researchers to propose several modified variants of GANs in an attempt to provide improved performance (Zhao et al., 2016; Arjovsky et al., 2017; Mao et al., 2017), scalability (Shaham et al., 2019), and robustness (Metz et al., 2016). Evidently, GANs can be an effective solution to generate new realistic samples to compensate for the scarcity of training instances in the minority classes (Wang et al., 2017a; Douzas and Bacao, 2018; Mariani et al., 2018; Ali-Gombe and Elyan, 2019; Liu et al., 2019; Tripathi et al., 2019; Wang et al., 2019; Hwang et al., 2019). However, GANs are susceptible to boundary distortion which may compromise

the majority class distribution (Santurkar et al., 2018). Further, GANs are prone to mode collapse, especially when a limited amount of data is available to it (Srivastava et al., 2017). A competing deep generative model Variational Auto Encoder (VAE) (Kingma and Welling, 2013) is also a popular choice to generate new data instances for oversampling (Wan et al., 2017; Zhang et al., 2018b; Guo et al., 2019). However, compared to GAN, the images produced by VAE are blurry with soft edges and limited detailing.

- *Algorithm-level methods:* Similar to the traditional classifiers, the technique to improve immunity against class imbalance by incorporating class-specific cost sensitivity is also common in deep learning (Wang et al., 2018; Sarkar et al., 2019; Cui et al., 2019). The deep learning framework also allows the classifier to learn the set of effective class-specific costs removing the need for expensive tuning (Khan et al., 2018; Johnson and Khoshgoftaar, 2019a) in the process. Alternatively, the focus can be laid on improving the quality of the learned features such that the classes become easy to distinguish (Huang et al., 2016; Ng et al., 2016; Dong et al., 2018; Nie et al., 2019; Hayat et al., 2019). Initially, designed for handling the imbalance between background and foreground in object detection problems focal loss (Lin et al., 2017a) gives lesser importance to correctly classified points while calculating the loss such that the network can focus on learning the hard to classify samples. This can be coupled with a class-specific cost tuning to further aid the classifier in the presence of class imbalance. In another prominent direction, efforts were made to tackle class imbalance by representing the problem as modeling the tail of a long-tailed distribution (Ouyang et al., 2016; Wang et al., 2017b; Li et al., 2020; Ma et al., 2020). This is intuitively appealing as the distribution of the number of training samples for a multi-class imbalanced problem is indeed a long-tailed one (Xiao et al., 2010) where the minority classes reside at the tail. Inherited from their traditional counterpart boundary shifting techniques are also widely used in deep classifiers. Here the task is to shift the learned decision boundary from the minority class either by accounting for the higher uncertainty of the classifier on such class (Khan et al., 2019) or by considering the label distribution during error calculation (Cao et al., 2019). The usefulness of ensemble learning such as boosting

was also investigated in works like (Yuan et al., 2018; Taherkhani et al., 2020).

1.5 Motivation and objective

Our discussion until this point establishes the importance of efficiently handling class imbalance during classification. We have also highlighted how the challenge still looms large even after a plethora of research works over the past couple of decades attempted to address the various issues associated with class imbalance. Further, we have observed how the problem of class imbalance is prominent in deep learning and how the traditional solutions are not directly applicable in such a paradigm. Among this vast pool of open problems, in this thesis, we primarily focus on three major issues related to class imbalance.

1. Designing an index that will be able to properly evaluate the performance of a classifier is no less important than proposing a new classifier. For the classification task, the most popular performance evaluation index is Accuracy which unfortunately fails in the presence of class imbalance (Japkowicz, 2006). This is because Accuracy is not designed to separately assess the performance of a classifier on each of the individual classes and aggregate them in a meaningful way such as Average Class Specific Accuracy (ACSA) (Huang et al., 2016) or GMean (Kubat et al., 1997). Such limitation leads indices like Accuracy to be biased towards the performance on the majority class. The problem becomes more challenging when possible concept drift (common in streaming data (Jaworski et al., 2017b,a; Duda et al., 2017; Jaworski et al., 2018; Ng et al., 2018)) and alteration in the number of classes between training and validation/test set (often observed in open set classification (Rudd et al., 2018)) are to be considered. Over the past decade, research works made significant effort to understand the effect of class imbalance on different performance evaluation indices by analyzing their character (Brzezinski et al., 2018; Ballabio et al., 2018). Remedial modifications and new indices were also proposed to achieve a fair evaluation of a classifier's performance in the presence of class imbalance (Bradley et al., 2006). However, such studies remained mostly limited to empirical validation or focused on specific applications.

2. The kNN (Fix and Hodges Jr, 1951) being one of the highly effective yet deceptively

simple non-parametric classifier has always been a widely popular choice. A modified variant Fuzzy k-Nearest Neighbor (FkNN) (Keller et al., 1985) incorporates the concept of fuzzy sets to provide for the inherent uncertainty of estimating the class distribution from a limited set of samples. FkNN also considers the lesser contribution of a distant neighbor in the kNN framework. These modifications enable FkNN to provide improved empirical performance over kNN. However, the theoretical quality assessment of the performance of the FkNN classifier did not enjoy significant attention from the research community. Further, the earlier studies ignored the fact that FkNN being a fuzzy classifier assigns class membership to test points instead of crisp labels. Moreover, FkNN like kNN is susceptible to the effect of class imbalance. However, investigating the nature of such vulnerability to class imbalance or its possible remedies were not fully explored after the introduction of FkNN.

3. In Section 1.4 we have demonstrated how class imbalance can affect the performance of a deep classifier. Further, we have discussed how popular oversampling techniques face significant challenges while being incorporated in a deep learning based classifier. However, we have mentioned that GAN can generate samples that posses the similar realistic nature of other examples. Unfortunately, GAN's generator is likely to generate points near the modes of the class distribution and may suffer from boundary distortion or mode collapse. Further, GANs are not capable of adaptively generating hard to classify samples for the classifier which will aid the learning in difficult regions. This is due to the fact that the GAN is not directly linked with the classifier.

Motivated by these shortcomings in the current literature our three primary objectives in this thesis are as follows:

1. Propose a general theoretical framework for analyzing the behavior of a performance evaluation index. The framework should contain a set of necessary conditions focusing on the possible concept drift and changes in the number of classes between training and validation/test set.

2. Present a theoretical convergence analysis of the class membership estimator employed by the FkNN classifier keeping in mind its relation with the classification performance.

The analysis should also provide an explanation for the possible performance deterioration of FkNN in the presence of class imbalance. Further, suggest remedial modifications that can provide better immunity to FkNN against the adverse effects of class imbalance.

3. Investigate the applicability of oversampling techniques in deep image classifiers. The oversampling approach should be adaptive in nature as well as capable to generate valid samples that can be mapped to realistic images.

1.6 Organization and chapter wise contributions

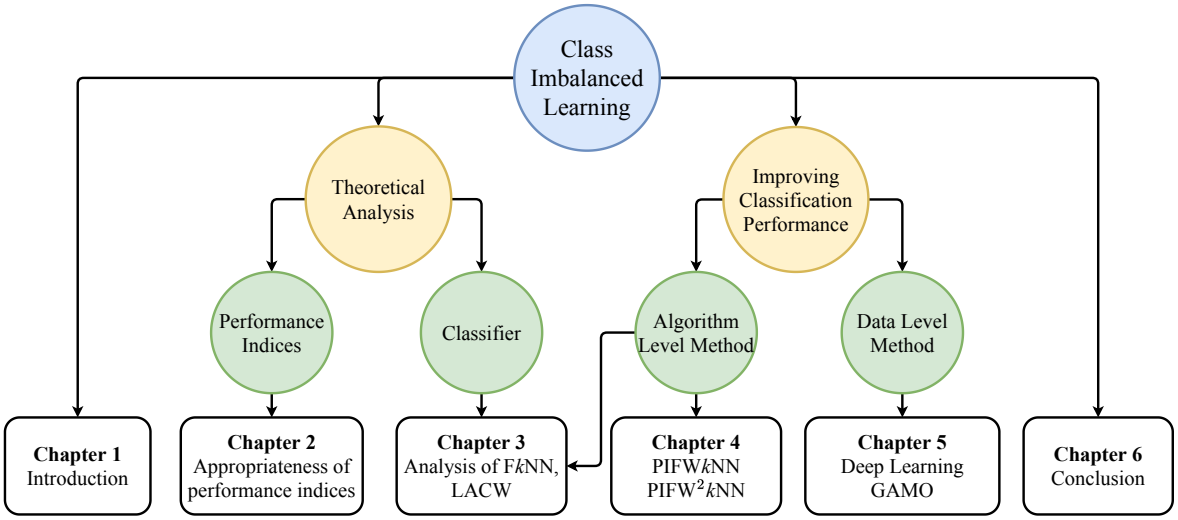


Figure 1.6: Road map of this thesis.

In order to fulfill our objectives, this thesis attempts to make a collection of contributions on the long-standing problem of class imbalance. Specifically, following the introductory Chapter 1 we analyze the applicability of different indices in the presence of class imbalance in Chapter 2. Following in Chapter 3 we characterize the effects of class imbalance on Fuzzy-k-Nearest Neighbor (FkNN) (Keller et al., 1985) classifier. We further propose a simple yet efficient point-specific Locally Adaptive Class Weighting (LACW) scheme to improve the immunity of FkNN against class imbalance. In Chapter 4 we introduced an improved variant of FkNN called Parameter Independent fuzzy class-specific Feature Weighted kNN (PIFWkNN) which can use an optimized set of class-specific feature weights alongside a global

value of k. Moreover, we propose PIFW^2kNN which introduces additional optimized class weights in PIFWkNN to improve its resilience against class imbalance. In the subsequent Chapter 5 we investigate the efficacy of oversampling techniques to combat the effect of imbalance in deep image classifiers. Finally, in Chapter 6 we present the concluding remarks and discuss the future scope of extensions listing the open problems. A brief road map of this thesis is illustrated in Figure 1.6.

1.6.1 Contributions of Chapter 2

In this chapter, we start by defining two necessary conditions (detailed in Section 2.2) that a performance evaluating index must satisfy while assessing a classifier in the presence of class imbalance. Unlike previous studies, our proposed conditions theoretically validate the applicability of an index from a more general perspective. Specifically, we want to ensure a fundamental criterion that if the class-specific performance of a classifier remains invariant then irrespective of any alteration in the validation/test set it should be evaluated uniformly as well. To elaborate, we identify two scopes of data alterations namely, the varying class priors between training and validation/test sets and the increasing number of classes. We show that satisfying the two proposed conditions ensures that the index under concern will be invariant to the two types of alterations and thus be capable of providing a fair assessment. Subsequently, we present a theoretical framework to validate the invariance of four binary and five multi-class widely used performance evaluation indices to the couple of necessary conditions. If an index is found violating any of the two conditions then a remedial modification and/or normalization is suggested. Further, we attempt to assess the interpretability of those indices which satisfy both the necessary conditions. To achieve this we define a third condition that guards an index against being biased towards the extremely poor performing class. We also present an empirical study to support our theoretical findings. Finally, under the light of the three conditions, we present a detailed discussion on the applicability of different indices in a diverse range of applications regularly affected by class imbalance.

1.6.2 Contributions of Chapter 3

Motivated by a lack of significant research on the quality assessment of the class memberships estimator in FkNN, in this chapter we attempt to bridge the gap by presenting a theoretical analysis. We start by considering the specific nature of FkNN which attempts to provide a membership for all the classes to a test point instead of crisply assigning it to a particular class. This observation leads us to investigate the convergence of the FkNN class membership estimator unlike studies focusing on the misclassification error (Yang and Chen, 1991). We specifically show that the bias and Mean Squared Error (MSE) of the class membership estimator employed by FkNN both converge to zero with the increasing availability of training examples. Such a convergence analysis has three significant advantages. First, we only need to make some elementary assumptions on the choice of membership function, class distribution, and the two parameters associated with FkNN. Second, owing to the choice of our unbounded non-negative loss functions our analysis can be directly extended to multi-class classification problems as well. Third, as an important implication, our analysis gives a straightforward explanation for the vulnerability of FkNN in class imbalanced problems. Therefore, to improve the resilience of FkNN against the effects of class imbalance we propose a point-specific Locally Adaptive Class Weighting (LACW) strategy. To compensate for the scarcity of the minority class LACW adaptively assigns a set of class-specific weights considering the locality of a given test point by employing a simple heuristic. The proposed theoretical convergence analysis is also supported by a simulation study on artificial and real-world classification datasets. Moreover, the effectiveness of the proposed LACW when coupled with the weighted variant of FkNN is validated by experimental comparison with FkNN variants tailored for handling class imbalance on real-world benchmark imbalanced datasets.

1.6.3 Contributions of Chapter 4

In this chapter, we propose an improved variant of FkNN called Parameter Independent Fuzzy class-specific feature Weighted kNN (PIFWkNN) which integrates an optimized set of class-specific feature weights along with a global choice of parameter k. To elaborate, not all features are equally important to discriminate a class from the others. Thus, an optimized

class-specific feature weighting during distance calculation can be beneficial for easing the classification. Further, the performances of kNN type classifiers are largely dependent on the choice of a global k. However, it is difficult to design an objective function which will optimize the set of class-specific feature weights as well as the choice of the parameter k while possessing necessary properties required by a mathematical optimizer. Therefore, we choose to optimize such an objective function by the help of Evolutionary Optimization (EvO) algorithms namely Differential Evolution (DE) (Das and Suganthan, 2011; Das et al., 2016). However, the performance of DE is reliant on the tuning of a couple of parameters called scale factor and crossover probability. An improved variant of DE called Success History based Adaptive DE (SHADE) (Tanabe and Fukunaga, 2013) attempts to address this issue by adaptively selecting the parameter values removing the costly tuning in the process. Hence, in PIFWkNN we employ SHADE to optimize the set of feature weights and the global k. A comparative study with kNN and FkNN variants employing feature weighting on real-world datasets highlights the efficacy of the proposed PIFWkNN. Even though class-specific feature weighting is likely to offer some immunity against the effects of class imbalance the general scarcity of minority points may still hinder a proper optimization of such weights. Thus, to additionally guard against the adversarial impact of class imbalance we propose to include a set of optimized class-specific weighting in the PIFWkNN framework. We call this classifier PIFW^2kNN which employs a modified objective function to simultaneously optimize class-specific feature weights, class-specific weights, and a global value of k. Experiments on real-world benchmark class imbalanced datasets validate the improved performance of PIFW^2kNN in comparison to the state-of-the-art classifiers capable of efficiently handling class imbalance.

1.6.4 Contributions of Chapter 5

In this chapter, we propose an end-to-end data level approach called Generative Adversarial Minority Oversampling (GAMO) which can adaptively oversample the minority class(es) to mitigate the effects of class imbalance on a deep image classifier. In essence, GAMO attempts to adversarially connect a generator with a classifier. Such a relationship ensures that the generator will be able to combat class imbalance by routinely supplying new difficult to classify samples to the classifier. However, due to the adversarial relationship with the

classifier, the generator will attempt to push the generated points towards the class periphery. If left unchecked then such a generator may end up generating out-of-class samples in an attempt to fool the classifier. To prevent this instead of a classical generator we propose a convex generator that generates a new sample for a given class as a convex combination of the existing training examples from that class. However, if the class is not convex then such a convex generator may still violate the corresponding class distribution. To address this issue we propose to include an additional discriminator in the framework which will guide the convex generator to follow the class distribution. Hence, GAMO can be expressed as a three-player game between a convex generator, a discriminator, and a classifier, where the generator attempts to fool both of the other two networks. However, GAMO generates new samples in the learned feature space whereas many applications may require balancing the original training set. Thus, we propose a GAMO2pix network which can learn the reverse mapping from the distributed feature to image space. This also helps us to assess the quality and diversity of the artificial instance generated by GAMO. An ablation study and empirical evaluation support the performance of the proposed technique.

1.6.5 Contributions of Chapter 6

In this chapter, we summarize the conclusions made by the previous chapters and discuss some open problems and future directions of research on the topic.

Chapter 2

Appropriateness of Performance Indices for Imbalanced Data Classification

Summary

Indices quantifying the performance of classifiers under class-imbalance, often suffer from distortions depending on the constitution of the test set or the class-specific classification accuracy, creating difficulties in assessing the merit of the classifier. In this chapter, we identify two fundamental conditions that a performance index must satisfy to be respectively resilient to the altering number of testing instances from each class and the number of classes in the test set. In light of these conditions, under the effect of class imbalance, we theoretically analyze four indices commonly used for evaluating binary classifiers and five popular indices for multi-class classifiers. For indices violating any of the conditions, we also suggest remedial modification and normalization. We further investigate the capability of the indices to retain information about the classification performance over all the classes, even when the classifier exhibits extreme performance on some classes. Simulation studies are performed on high-dimensional deep representations of a subset of the ImageNet dataset using four state-of-the-art classifiers tailored for handling class imbalance. Finally, based on our theoretical findings and empirical evidence, we recommend the appropriate indices that should be used to evaluate the performance of classifiers in presence of class-imbalance.

2.1 Introduction

2.1.1 Overview

Class imbalance in the training set leads the classifier to be biased in favor of the majority classes and consequently suffers from higher misclassification on the minority classes. Evidently, such bias should be properly taken under consideration during performance evaluation. This indicates the need for special indices, unlike the widely used Accuracy measure which lays more stress on the performance over the majority classes, being unsuitable in presence of class imbalance (Sokolova et al., 2006; Japkowicz, 2006).

Over the years, for a binary imbalanced classification problem indices like Recall, Specificity, and Precision (Buckland and Gey, 1994) were considered to be the basic measures of performance. However, by design, Recall (or Sensitivity) measures the accuracy over the minority (positive) class, Specificity does the same for the majority (negative) class, and Precision (Buckland and Gey, 1994) considers the fraction of positives which are accurately classified (true positive) to the number of instances predicted as positives. In other words, these three measures offer different criteria of evaluation by respectively focusing on the true positive, true negative (analogous to true positive for the negative class), and false-positive (negative instances wrongly classified as positive) counts and a good classifier is expected to optimize all of them. However, optimizing multiple indices simultaneously is difficult in practice, especially if a trade-off is required. Therefore, attempts were made to combine two or more of these basic indices together to form new measures that can consider multiple distinct aspects during evaluation as well as provide easy interpretability. For example, the GMean (Kubat et al., 1997) index is calculated by taking the geometric mean of Sensitivity and Specificity, while Area Under Receiver Operating Characteristics (AUROC) (Hand and Till, 2001) measure is found by plotting Recall against False Positive Rate (FPR). Similarly, the Precision and Recall can be combined to form the Area Under Recall Precision Curve (AURPC) (Davis and Goadrich, 2006) index.

In the case of the multi-class classification, a direct extension of the GMean index is available (Branco et al., 2016). The multi-class analog of Recall is the Average Class-Specific Accuracy (ACSA) (Huang et al., 2016). AUROC can be extended for multi-class classification problems by either the One Versus One (OVO) strategy to calculate AUROC-OVO (Hand

and Till, 2001), or by the One Versus All (OVA) strategy to find AUROC-OVA (Japkowicz, 2013). Similarly, the multi-class version of AURPC is called AURPC-OVA (Japkowicz, 2013), as the extension warrants the use of the OVA strategy. In the following Table 2.1 we briefly describe the indices which are analyzed in detail in the subsequent sections of this article.

Table 2.1: Brief description of the indices discussed in this chapter (formally detailed in Definition 2.3 and 2.5).

Index	Brief description
GMean (Kubat et al., 1997)	Geometric mean of all the class-specific accuracies. Applicable to two-class as well as multi-class classification problems.
AUROC (Hand and Till, 2001)	Can be reduced to the arithmetic mean of the class-specific accuracies in a two-class classification problem.
Precision (Buckland and Gey, 1994)	In a two-class classification problem it is defined as the fraction of true positives to the total number of instances which are classified as positives.
AURPC (Davis and Goadrich, 2006)	Reduces to the arithmetic mean of accuracy over the positive class and Precision in a two-class classification problem.
ACSA (Huang et al., 2016)	Arithmetic mean of the class-specific accuracies in a multi-class classification problem.
AUROC-OVA (Japkowicz, 2013)	Direct extension of AUROC for multi-class classification using OVA strategy.
AUROC-OVO (Hand and Till, 2001)	Direct extension of AUROC for multi-class classification using OVO strategy.
AURPC-OVA (Japkowicz, 2013)	Direct extension of AURPC for multi-class classification using OVA strategy.

2.1.2 Background

The growing number of classification performance measures inspired the research community to investigate their uniqueness, compare their applicability to class imbalanced problems in general, and evaluate their suitability for specific applications. Studies like (Daskalaki et al., 2006; Ferri et al., 2009; Ballabio et al., 2018) attempted to empirically find the inter-relation between indices, while observing their behavior under different scenarios. However, these empirical analyses are dependent on the choice of classifiers as well as the datasets, therefore failing to provide general conclusions. In contrast, a theoretical approach is taken in (Joshi, 2002) and (Liu et al., 2007) to model the change of an index value with respect to varying class imbalance. However, these preliminary approaches, in addition to being complicated did not consider the possible disparity between the training and test sets. A simpler, more structured framework was proposed by Sokolova and Lapalme (2007, 2009) which was later

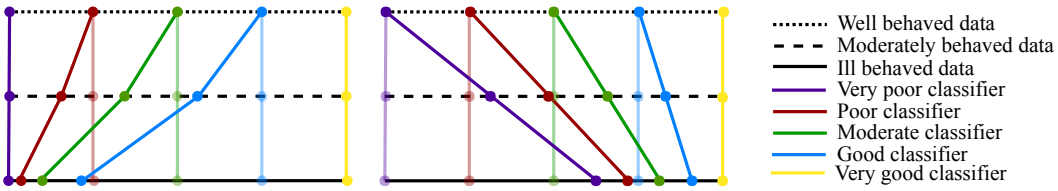
extended by Brzezinski et al. (2018). They formalized a set of transformation conditions on the confusion matrix (Kubat et al., 1997) to imitate changes in classification performance as well as alterations in the test set. An index is called invariant (or considered unaffected) by a certain transformation if its value does not change despite the transformation. Luque et al. (2019) took a different direction by building upon the measure of class imbalance proposed in (Núñez et al., 2017) and defining a set of indicators to theoretically analyze the bias of various indices in binary classification problems. Recently, the work done by Brzezinski et al. (2018) was further extended in (Brzezinski et al., 2019) for imbalanced streaming data classification (Yamazaki et al., 2007; Alaiz-Rodríguez and Japkowicz, 2008). The authors attempted to properly interpret the value returned by an index especially under the effect of the dynamically changing class priors between the training and test sets, which is fairly common for streaming data. However, these works mostly focused on the application-specific suitability of an index. This limits them from discussing on a set of necessary conditions, violation of which may deem the index as undesirable for general use, along with offering any remedial modifications to impose invariance. Moreover, they considered the key dataset properties such as the number of classes as constant. This restricts them from addressing the pivotal role that the altering number of classes may play in distorting an index, a situation common to open set classification problems (Rudd et al., 2018).

2.1.3 Motivation

In an attempt to rectify the shortcomings of the existing literature (as discussed in Section 2.1.2), we carry out a systematic theoretical study on the desirable properties of the indices. The presented properties are fundamental in the sense that they ensure the invariance of an index against the following two types of undesirable distortions:

Type 1 distortion: The fraction of representatives from a class in the test set $X' \subseteq \mathbb{S}$ (containing n data instances) may not always be similar to that of X [1]. However, under such conditions, the value returned by some performance indices (such as Precision (Bradley et al., 2006)) tends to vary with changes in the number of test points from the different classes. An example of this type of distortion is illustrated in Figure 2.1a, where the mapping within the

[1] All our analyses and discussions in this chapter are applicable to both of the validation and test sets. Thus, throughout this chapter validation and test sets are often considered as synonymous and are denoted by the same notation for simplicity.



(a) Type 1 distortion of index. (b) Type 2 distortion of index. (c) Legends.

Figure 2.1: Two types of distortions can affect an index while quantifying the performance of classifiers of varying quality over datasets posing diverse degrees of challenge. The complexity of the datasets (plotted by black lines along the horizontal axis) ranges over well behaved (dotted line), moderately behaved (dashed line), and ill behaved (solid line). The quality of the classifier (plotted as colored lines along the vertical axis) varies between very poor (magenta), poor (red), moderate (green), good (blue), and very good (yellow). The ideal behavior of an index is illustrated in the background. (a) Type 1 distortion results in the index becoming increasingly warped within its stipulated range. Here the behavior of a dataset is characterized by the variation in the class priors from the training set to those of the test sets. In well behaved data no variation takes place while the mild and high amount of disparity is respectively observed for moderately and ill behaved data. (b) Type 2 distortion results in the range of the index becoming progressively smaller. Here the number of classes remains constant in a well behaved data, while small and high increase in c respectively indicates a moderately and ill behaved data. Best viewed in color in the electronic version.

range of the index (which remains unchanged) becomes increasingly warped. The change in the fraction of representatives between training and test (or validation) sets may happen in a real-world classification problem due to a couple of reasons. Firstly, concept drift (Yamazaki et al., 2007; Alaiz-Rodríguez and Japkowicz, 2008) can result in continuous alterations of class priors (and consequently the degree of imbalance) over time. Such drifting is fairly common in imbalanced streaming data classification problems (Brzezinski et al., 2019), resulting in different extents of class imbalance in training, validation, and test sets. Secondly, prior probability shift between training, validation, and test sets also occurs in current large scale benchmark datasets such as LSUN (Yu et al., 2015) and ImageNet (Deng et al., 2009), where the class priors in the training set are not retained in the predefined validation and test sets. It is important to note that remedial measures like stratified cross-validation (López et al., 2014) cannot be efficiently applied in both of these situations.

If an index suffering from the above-mentioned distortion is used during validation, then the classifier will be improperly evaluated and consequently miscalibrated. Further, in streaming data classification, an application may require the classifier to be tested at regular in-

tervals, so that the classifier's parameters can be periodically fine-tuned according to the latest performance. Here also, the use of an index that is susceptible to this first type of distortion may lead to inappropriate judgment about the quality of the classifier and mislead the periodic retraining procedure.

Type 2 distortion: The range of possible values to be returned by some performance indices (for example, AUROC-OVO as shown in Theorem 2.3) gets diminished with an increase in the number of classes in the data. Thus, an index affected by such a distortion may fail to identify the better classifier with decreasing confidence as the number of classes increases, even when the contenders are of diverse quality. In the worst case, on a very large number of classes, due to rounding error a set of classifiers may end up being evaluated as similar, all providing commendable performance instead of reflecting their actual quality. An example of this distortion is also illustrated in Figure 2.1b, where the lower bound of the index gradually increases.

Example 2.1. *To better illustrate the two types of distortions we present an example in Figure 2.2. For Type 1 distortion, we take a two-class dataset*[2] *where each class is drawn from a normal distribution. To quantize the level of class imbalance in the test set, we use a measure called Ratio of Representatives in the Test set (RRT), which for a two-class classification problem can be expressed as* $\frac{n_{maj}}{n_{min}}$*, where* n_{maj} *and* n_{min} *are respectively the number of test points from the majority and the minority class (see Definition 2.1). A well behaved test set as shown in Figure 2.2a, is created by sampling 5000 points from each class. Progressively worse behaved test sets are formed by varying the RRT between 2,4,6,8, and 10. Now as shown in Figures 2.2a and 2.2b, let us shift a linear classifier from the best possible position (which accurately separates the two classes) to the worst (which only perfectly classifies the majority). We measure the Precision of the different classifiers on the varying test sets and plot them in Figure 2.2c. We can observe that even though the classifier maintains its quality by remaining at a fixed position, the Precision decreases with increasing RRT (for example, at the class boundary* $x = 4$ *the Precision deteriorates from 0.93 to 0.59 when RRT is altered from 1 to 10). The change in Precision is high when the classifier is of poor quality, while the gap progressively closes down with improving performance.*

In the case of Type 2 distortion, we start with a three-class dataset as in Figure 2.2d where

[2]The construction of all the datasets used in this example is detailed in Section A.1 of Appendix A.

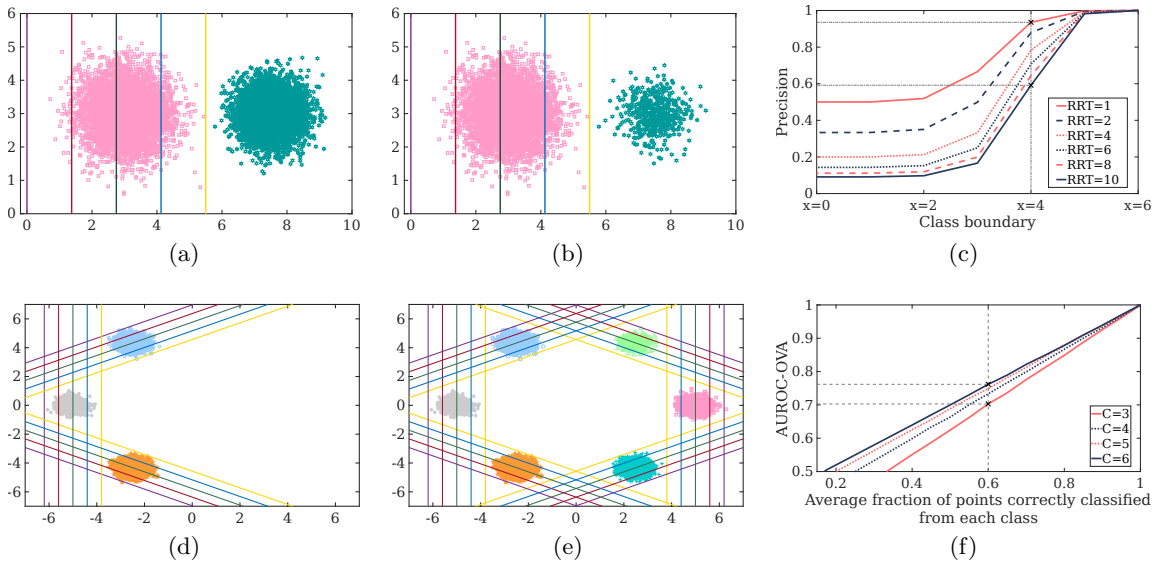


Figure 2.2: Illustrative example of the two types of distortions; (a)-(c) for Type 1 and (d)-(f) for Type 2. Please note that the quality of the linear classifiers follow the legends as in Figure 2.1. (a) Balanced two class dataset (Well behaved) (b) When RRT is set to 10 by sub-sampling from the class in the right (Ill behaved). (c) Effect of Type 1 distortion on Precision: the value returned by the index deteriorates with increasing RRT, even when the classifier remains the same. (d) Three class dataset (Well behaved) (e) The final six class dataset after adding the rest of three classes on the vertices of the regular hexagon (Ill behaved in the sense of Type 2 distortion). (f) Effect of Type 2 distortion on AUROC-OVA: progressively higher index value is produced for the similar performing classifier while c increases. Best viewed in color in the electronic version.

the classes are sampled from normal distributions centered at the three adjacent vertices of a regular hexagon. We gradually increase the number of classes to six in Figure 2.2e by similarly sampling on the rest of the three vertices in an anticlockwise order. In each case, we start with an OVA ensemble of linear classifiers which performs as worse as a uniformly random assignment and gradually moves towards the best which achieves perfect accuracy. As previous, we calculate the AUROC-OVA of the different classifiers on the various datasets and plot them in Figure 2.2f. We can observe that even when a classifier is accurately classifying the same fraction of points from each class the AUROC-OVA index gradually returns a higher value with the increasing number of classes (for example, when the classifier correctly predicts 60% points from each class on average, the AUROC-OVA reaches from 0.70 to 0.77 with c altering from 3 to 6).

Evidently, an evaluation index may suffer from either or both of these distortions with

the change in the properties of the dataset (such as the number of classes, size of the test set, and extent of class imbalance) even if the classifier retains a consistent performance. Consequently, such types of distortions primarily complicate the interpretation of a value returned by an index. For example, the classifier with moderate performance in Figure 2.1 can either be assigned a higher index value or a lower index value (compared to the ideal) depending on the nature of the distortion suffered by the performance indices. Hence, the actual index values cannot be used to properly assess the merit of a classifier. Another issue arises when the difference between the values yielded by a bad classifier and a good one gets diminished to the extent of being ignored due to the rounding of values in practical experiments.

Even when an index is found to be unaffected by the two types of distortions it may still provide a value from which adequate information about the performance of a classifier over all the classes is difficult to extract. This usually occurs in multi-class imbalanced classification problems, where high misclassification in a single class (irrespective of the classifier's performance over the other classes) results in a severely deteriorated index value.

2.1.4 Contributions of Chapter 2

In this chapter, we identify two necessary conditions (described in Section 2.2) that an index must satisfy to be considered ideal for evaluating the performance of classifiers on imbalanced datasets. Contrary to the prior works, we do not focus on the application-specific suitability of any index and aim to propose a set of constraints that will evaluate an index from a more generalized perspective. To elaborate we look at the nature of changes in the data itself which can affect an index. Evidently, it is expected that an index should remain invariant to any changes in the training/test set if the classifier performs uniformly. Therefore, ensuring such can be considered as a fundamental requirement over all other types of secondary consistency checks. In essence, under the assumption that the classifier sustains its performance over each of the classes, we formulate two transformation conditions on the confusion matrix. Invariance to both of these transformations will ensure the immunity of an index against the two types of distortions. In Table 2.2, we put our contributions in proper context with the existing works. We further summarize the contributions as follows:

Table 2.2: Summary of contributions made in this chapter in comparison to existing literature (no references are provided for original contributions).

Topics	Reference	Our contribution
Condition 2.1	Sokolova and Lapalme (2009)	Established as a necessary safeguard against Type 1 distortion.
Invariance of Recall, Precision, and AUROC to Condition 2.1	Sokolova and Lapalme (2009)	Re-validated using a mathematical framework.
Invariance of GMean, ACSA, AUROC-OVA, AUROC-OVO, and AURPC-OVA to Condition 2.1	-	Validated using a mathematical framework.
Remedial modification of Precision, AURPC, and AURPC-OVA to satisfy Condition 2.1	Bradley et al. (2006)	Validated using a mathematical framework. Established as an effective replacement to the original ones, which satisfy Condition 2.1.
Remedial normalization of AUROC-OVA to satisfy Condition 2.1	-	Proposed and validated using a mathematical framework.
Condition 2.2	-	Proposed as a necessary safeguard against Type 2 distortion.
Invariance of Recall, Precision, AUROC, GMean, ACSA, AUROC-OVO, AUROC-OVA, and AURPC-OVA to Condition 2.2	-	Validated using a mathematical framework.
Remedial normalization of AUROC-OVO, and AUROC-OVA to satisfy Condition 2.2	-	Proposed by us and validated using a mathematical framework.
Condition 2.3	-	Proposed to validate the quality of the information returned by an index which satisfies Condition 2.1 and 2.2.
Invariance of GMean, ACSA, and modified AURPC-OVA to Condition 2.3	-	Validated using a mathematical framework.

1. The first condition guards against the Type 1 distortion by ensuring invariance of an index with alterations of the size and sample distributions among the different classes in the test set. This condition was first introduced by Sokolova and Lapalme as properties I_6 and I_8 in 2009 (where the former is a special case of the later). However, they were not motivated to evaluate the effects of distortions over the indices. Therefore, their analysis did not elaborate on the implication of the properties or identify them as fundamental constraints. In this article, we bridge this gap by establishing this condition as a necessary measure against the Type 1 distortion. Moreover, in Theorems 2.1 and 2.3 under the light of the first condition we analytically discuss the properties of GMean, ACSA, AUROC-OVO, AUROCC-OVA, and AURPC-OVA, none of which were covered in the previous studies.

2. We propose the second condition which deals with the Type 2 distortion assuring invariance to the varying number of classes in the test set, as shown in Theorem 2.3.

3. We show in Theorem 2.2 that contrary to the regular Precision and AURPC indices, the modifications proposed in (Bradley et al., 2006) are indeed capable of inducing invariance under the first condition. We further propose the normalized variants of AUROC-OVO (which essentially reduces to ACSA) and AUROC-OVA, which offer immunity against the effects of the two types of distortions.

4. We also propose the third condition to ensure that in a multi-class classification problem, an index that fulfills the two fundamental desirable properties are also capable to provide sufficient information about the classifier's performance over all the classes, even when a single class suffers extremely high misclassification. We show in Theorem 2.5 that except GMean, both ACSA and AURPC-OVA offer invariance under the third condition.

In this chapter, we also present an empirical analysis on some selected subsets of ImageNet (Deng et al., 2009) in Section 2.5 to experimentally validate our theoretical findings and effectiveness of the prescribed remedies. Finally, in Section 2.6, we present a discussion on the applicability of different indices in imbalanced classification tasks, and make recommendations as per situation, and subsequently conclude in Section 2.7.

2.2 Desirable properties for performance indices

Various performance evaluating indices depend on the diverse properties (such as imbalance, number of classes, etc.) of the training and/or test set to different extents, resulting in improper/ambiguous evaluation of a classifier. This issue can be resolved by defining a set of necessary but not sufficient conditions to ensure the quality of the evaluation by an index. For a training set, the extent of class imbalance can be quantified with the help of IR as defined in Definition 1.2 in Chapter 1. However, a classifier is trained on a single training set X with a fixed predefined IR_{pair}, whereas all possible test sets might not follow a distribution of the representatives among the classes (i.e. IR_{pair}) similar to X. Therefore, analogous to IR we define RRT for quantifying the ratio of representatives among the classes in the test

set.

Definition 2.1. *For a 2-class classification problem, the RRT is defined as the proportion of the number of points from the majority class to that of the minority class, where the majority and the minority classes are named according to the training set. This definition can be extended for the c-class classification case in a manner similar to IR, where the set of a pairwise ratio of the number of the data instances among the c classes is denoted by* RRT_{pair}*, i.e.* $RRT = \max RRT_{pair}$.

Performance of the classifier on a c-class classification problem can be expressed in the form of a matrix called the *confusion matrix*, which is defined as follows:

Definition 2.2 (Kubat et al. (1997))**.** *A confusion matrix over a test set* X' *for a c-class classification problem can be defined as* $Q_c = [q_{ij}]_{c\times c}$*, where* q_{ij} *represents the number of points which actually belongs to* i^{th} *class but are predicted as a member of class* j*, for all* $i, j \in C$*. Thus, the diagonal elements i.e.* q_{ii} *are those instances of class* i *which are correctly classified while the rest are different misclassifications. Evidently, each entry in the confusion matrix must be a non-negative integer i.e.* $q_{ij} \in \mathbb{Z}^+ \cup \{0\};\ \forall i, j \in C$

There are some important properties of the confusion matrix which we detail in the following discussion.

Property 1: The sum of entries in the i^{th} row of the confusion matrix is denoted by n_i (i.e. $\sum_{j=1}^{c} q_{ij} = n_i;\ \forall i \in C$), which is the number of test points belonging to the i^{th} class. We assume that $n_i > 0$, as there should be at least one point from each class in the test set.

Property 2: The sum of entries in the i^{th} column of the confusion matrix is denoted by r_i (i.e. $\sum_{j=1}^{c} q_{ji} = r_i;\ \forall i \in C$), which is the number of test points predicted as i^{th} class.

Property 3: The total number of test points $n = \sum_{i=1}^{c}\sum_{j=1}^{c} q_{ij};\ \forall i, j \in C$.

Property 4: In case of two-class classification, the entries of Q_2 are specially named, as True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN), when the test instances from the majority and the minority classes are respectively labelled as -1 (negative) and +1(positive). Therefore, Q_2 can be formally represented as:

$$Q_2 = \begin{bmatrix} TP & FN \\ FP & TN \end{bmatrix}. \tag{2.1}$$

Before proceeding further we need to describe our primary assumption based on which the following theory will be built.

Assumption 2.1. *The class-specific performance (the fraction of correct classification as well as the proportion of misclassification to each of the other classes) of a classifier remains the same over any random subset of the dataset.*

The class-specific performance of a classifier can be considered as an equivalence relation, which can partition the set $\mathbb{Q}_c$ containing all possible c-class confusion matrices into some equivalence classes. In any of these equivalence classes, the class-specific performance of a classifier remains constant over all the classes. To elaborate, given two confusion matrices say Q_c and Q'_c[3], if the equivalence relation $q_{ij}/n_i = q'_{ij}/n'_i$, satisfies for all $i, j \in C$, then they can be considered as members of the same equivalence class, i.e. $Q_c \sim Q'_c$. In other words the equivalence property essentially corresponds to constant performance by a classifier or formally represents Assumption 2.1. Using the notion of the confusion matrix, we can now formally define a performance evaluation index as a function f mapping from the set of all possible confusion matrices $\mathbb{Q}_c$ to a real scalar quantity. Such a representation is important as it helps us define some functionals to formally describe our proposed conditions, in the following manner.

Condition 2.1. *The value of an index should not be dependent on RRT, if the classifier performs equivalently, i.e.*

$$\mathcal{V}_{Q_c}(f) = \mathcal{V}_{Q'_c}(f);\ \forall Q_c \sim Q'_c,$$

where $\mathcal{V}_{Q_c}$ is a functional evaluating the index f on the confusion matrix Q_c.

As an extension of the work by Sokolova and Lapalme (2009), in this article we propose Condition 2.1 as a necessary measure against the Type 1 distortion, violation of which may alter the value of an index with the changes of RRT in the test set even when the classifier remains the same.

[3]Evidently, $Q'_c = [q'_{ij}]_{c\times c}$, $\sum_{i=1}^{c}\sum_{j=1}^{c} q'_{ij} = n'$, $\sum_{j=1}^{c} q'_{ij} = n'_i$, and $\sum_{j=1}^{c} q'_{ji} = r'_i$ for all $i \in C$.

Condition 2.2. *The lower and the upper bounds of an index f should not be dependent on the number of classes, i.e.*

$$\mathcal{L}_{\mathbb{Q}_c}(f) = \mathcal{L}_{\mathbb{Q}_{c+1}}(f)\ \forall c \in \mathbb{Z}^+ \setminus \{1\},$$
$$\text{and } \mathcal{U}_{\mathbb{Q}_c}(f) = \mathcal{U}_{\mathbb{Q}_{c+1}}(f)\ \forall c \in \mathbb{Z}^+ \setminus \{1\},$$

where, $\mathbb{Q}_c$ and $\mathbb{Q}_{c+1}$ are respectively the sets of all possible c-class and $(c+1)$-class confusion matrices. Moreover, $\mathcal{L}$ and $\mathcal{U}$, are two functionals of f, respectively calculating the minimum and maximum value of f over all $\mathbb{Q}_c$ and $\mathbb{Q}_{c+1}$.

Condition 2.2 ensures that under the assumption of a consistent performance by a classifier, the value of an index should not be biased to differing number of classes in the test set.

If we consider a c-class confusion matrix, where $q_{ii}/n_i = \epsilon$, only for the i^{th} class ($i \in C$) while $q_{jj}/n_j \geq (1-\epsilon)$ for all the other classes ($j \in C \setminus \{i\}$), and $\epsilon = \frac{1}{c}$, then all such matrices form a set $\mathcal{Q}_c(i) \subset \mathbb{Q}_c$. In other words, $\mathcal{Q}_c(i)$, is the set of all such c-class confusion matrices where the classifier performed extremely poor only on the i^{th} class.

Condition 2.3. *An index f while evaluating a multi-class classifier should not be biased towards the misclassification of a single class, i.e. $\mathcal{W}_{\mathcal{Q}_c(i)}(f) > \mathcal{L}_{\mathbb{Q}_c}(f)$, where $\mathcal{W}$ is a functional which calculates the limit of f, as $\epsilon \to 0^+$.*

In other words, Condition 2.3 ensures that the value returned by the index will not excessively degrade if a single class is almost entirely misclassified in a multi-class classification problem. Violating Condition 2.3 will lead the index to lose information about the classifier's performance on all the other classes. Evidently, index failing to satisfy Condition 2.3 will be unable to distinguish between two classifiers, one of which achieves good class-specific accuracies on all but one class, while the other achieves high misclassification on all.

2.3 Analysis of the two-class performance evaluation indices

In this section, we analyze the characteristics of four indices, namely GMean, AUROC[4], Precision, and AURPC; which are used to evaluate the performance of a classifier in the presence of class imbalance for a two-class classification problem. We only require to validate if the indices satisfy Condition 2.1 as the other is only applicable for multi-class classification.

Definition 2.3. *For a two-class classification problem, given a confusion matrix Q_2 as in (2.1),*

1. *The GMean index, denoted by γ_2 is defined as:*

$$\gamma_2(Q_2) = \left(\left(\frac{TP}{TP+FN}\right)\left(\frac{TN}{FP+TN}\right)\right)^{\frac{1}{2}}. \tag{2.2}$$

2. *The AUROC index, denoted by ρ is defined as:*

$$\rho(Q_2) = \frac{1}{2}\left(\frac{TP}{TP+FN} + \frac{TN}{FP+TN}\right). \tag{2.3}$$

3. *The Precision index, denoted by ζ is defined as:*

$$\zeta(Q_2) = \frac{TP}{TP+FP}. \tag{2.4}$$

4. *The AURPC index, denoted by κ is defined as:*

$$\kappa(Q_2) = \frac{\alpha'(M_2) + \zeta(M_2)}{2}, \tag{2.5}$$

where, $\alpha'(Q_2) = TP/(TP+FN)$ is the Recall.

Evidently, the formal definitions of the indices do correspond to their pedagogical description in Table 2.1. We may now proceed to analyzing the behavior of the indices under the light of Condition 2.1 in the following theorem.

Theorem 2.1. *For a two-class classification problem, given two confusion matrices Q_2 and Q'_2, the following statements can shown to be true if $Q_2 \sim Q'_2$,*

[4]For mathematical simplicity we have restricted ourselves to the discrete version of the index, which is popularly used in practice.

1. *The index γ_2 satisfies Condition 2.1.*

2. *The index ρ satisfies Condition 2.1.*

3. *The index ζ does not satisfy Condition 2.1.*

4. *The index κ does not satisfy Condition 2.1.*

Proof. Let us define Q_2, as in (2.1), while Q_2' can be constructed as,

$$Q_2' = \begin{bmatrix} b_1TP & b_1FN \\ b_2FP & b_2TN \end{bmatrix},$$

where $b_1, b_2 \in \mathbb{R}^+$, $b_1 \neq b_2$ and $q_{ij}' \in \mathbb{Z}^+, \forall i,j \in \{1,2\}$. Such a form of Q_2', will ensure that $q_{ij}/n_i = q_{ij}'/n_i'$, is satisfied for all $i,j \in \{1,2\}$, or $Q_2 \sim Q_2'$. With this initial setup we start the proof of the first statement by finding the value of $\mathcal{V}_{Q_2'}(\gamma_2)$, following (2.2):

$$\begin{aligned} \mathcal{V}_{Q_2'}(\gamma_2) &= \left(\left(\frac{b_1TP}{b_1TP + b_1FN}\right)\left(\frac{b_2TN}{b_2FP + b_2TN}\right)\right)^{\frac{1}{2}}, \\ &= \left(\left(\frac{TP}{TP+FN}\right)\left(\frac{TN}{FP+TN}\right)\right)^{\frac{1}{2}} = \mathcal{V}_{Q_2}(\gamma_2). \end{aligned}$$

Therefore, the γ_2 index satisfies Condition 2.1.

The second statement can be proved in a similar manner by starting from $\mathcal{V}_{Q_2'}(\rho)$ using (2.3),

$$\begin{aligned} \mathcal{V}_{Q_2'}(\rho) &= \frac{1}{2}\left(\frac{b_1TP}{b_1(TP+FN)} + \frac{b_2TN}{b_2(FP+TN)}\right), \\ &= \frac{1}{2}\left(\frac{TP}{TP+FN} + \frac{TN}{FP+TN}\right) = \mathcal{V}_{Q_2}(\rho). \end{aligned}$$

This proves the second statement.

Similarly, we show the third statement to be true by calculating $\mathcal{V}_{Q_2'}(\zeta)$ as per (2.4),

$$\mathcal{V}_{Q_2'}(\zeta) = \frac{b_1TP}{b_1TP + b_2FP}. \tag{2.6}$$

From (2.6) it is evident that $\mathcal{V}_{Q_2'}(\zeta)$ can be equal to $\mathcal{V}_{Q_2}(\zeta)$, only when $b_1 = b_2$, which implies that ζ violates Condition 2.1.

Finally, we prove the fourth statement by finding the value of $\mathcal{V}_{Q_2'}(\kappa)$ according to (2.5),

$$\begin{aligned}\mathcal{V}_{Q_2'}(\kappa) &= \frac{b_1 TP}{b_1(TP+FN)} + \frac{b_1 TP}{b_1 TP + b_2 FP}, \\ &= \frac{TP}{TP+FN} + \frac{b_1(TP)}{b_1(TP)+b_2(FP)}.\end{aligned} \tag{2.7}$$

Therefore, from (2.7), we can conclude in a manner similar to (2.6) that $\mathcal{V}_{Q_2'} = \mathcal{V}_{Q_2}$ only holds when $b_1 = b_2$, thus completing the proof. □

From Theorem 2.1, we can see that while GMean and AUROC indices satisfy Condition 2.1, the Precision and AURPC indices do not, thus being susceptible to RRT. In other words, both of Precision and AURPC may evaluate a good classifier as a poor choice, with the increase in RRT, even when the class-specific performances are retained. This is due to the increasing number of test points from the majority class which considerably increases FP. This was first observed by Bradley et al. (2006), who proposed a solution by incorporating the class priors in the definition of Precision. In their modified definition of Precision (and consequently AURPC) the direct use of FP is replaced with the ratio of false positives to the number of majority instances. The modified Precision and AURPC, respectively called mPrecision and mAURPC, are described in the following definition.

Definition 2.4 (Bradley et al. (2006))**.** *For a two-class classification problem, given a confusion matrix Q_2 as in (2.1), where $TP + FN = n_1$ and $FP + TN = n_2$,*

1. *The mPrecision index, denoted by $\hat{\zeta}$ is defined as:*

$$\hat{\zeta}(Q_2) = \frac{TP/n_1}{(TP/n_1) + (FP/n_2)}. \tag{2.8}$$

2. *The mAURPC index, denoted by $\hat{\kappa}$ is defined as:*

$$\hat{\kappa}(Q_2) = \frac{1}{2}(\alpha'(Q_2) + \hat{\zeta}(Q_2)). \tag{2.9}$$

Theorem 2.2. *For a two-class classification problem, given two confusion matrices Q_2 and Q_2', the following statements can shown to be true, if $Q_2 \sim Q_2'$,*

1. *The index $\hat{\zeta}$ satisfies Condition 2.1.*

2. The index $\hat{\kappa}$ satisfies Condition 2.1.

Proof. Given Q_2, as in (2.1), we construct Q_2', such that $Q_2 \sim Q_2'$, in a manner similar to Theorem 2.1. Then to prove the first statement we proceed by calculating $\mathcal{V}_{Q_2'}(\hat{\zeta})$ using (2.8).

$$\begin{aligned}\mathcal{V}_{Q_2'}(\hat{\zeta}) &= \frac{b_1 TP/b_1 n_1}{b_1 TP/b_1 n_1 + b_2 FP/b_2 n_2}, \\ &= \frac{TP/n_1}{(TP/n_1) + (FP/n_2)} = \mathcal{V}_{Q_2}(\hat{\zeta}).\end{aligned}$$

This completes the proof of first statement.

We prove the second statement by finding $\mathcal{V}_{Q_2'}(\hat{\kappa})$, which from (2.8), and (2.9) can also be written as

$$\begin{aligned}\mathcal{V}_{Q_2'}(\hat{\kappa}) &= \frac{b_1 TP}{2b_1 n_1} + \frac{b_1 TP/b_1 n_1}{2b_1 TP/b_1 n_1 + 2b_2 FP/b_2 n_2} \\ &= \frac{TP}{2n_1} + \frac{TP/n_1}{(TP/2n_1) + (FP/2n_2)} = \mathcal{V}_{Q_2}(\hat{\kappa}).\end{aligned}$$

Thus, the second statement is proved, completing the proof of this Theorem. □

From Theorem 2.2 we can conclude that the proposed modification of Precision and AURPC can improve their immunity over RRT by satisfying Condition 2.1, and in the process will be able to better evaluate a classifier.

2.4 Analysis of the multi-class performance evaluation indices

In this section, we will define five multi-class evaluation indices, namely GMean, ACSA, AUROC-OVO, AUROC-OVA, and AURPC-OVA. Similar to the two-class indices we present an analysis in the perspective of the first two conditions and prescribe modification/normalization as per requirement.

Definition 2.5. *Given a c-class confusion matrix Q_c as defined in Definition 2.2,*

1. The GMean index, denoted by γ_c is defined as:

$$\gamma_c(Q_c) = \left(\prod_{i=1}^{c} \frac{q_{ii}}{n_i}\right)^{\frac{1}{c}}. \tag{2.10}$$

2. The ACSA index, denoted by α is defined as:

$$\alpha(Q_c) = \frac{1}{c}\sum_{i=1}^{c}\frac{q_{ii}}{n_i}. \tag{2.11}$$

3. The AUROC-OVO index, denoted by ρ_o is defined as:

$$\rho_o(Q_c) = \frac{1}{2c}\sum_{i=1}^{c}\left(1 + \frac{q_{ii}}{n_i} - \sum_{\substack{j=1\\ j\neq i}}^{c}\frac{q_{ji}}{(c-1)n_j}\right). \tag{2.12}$$

4. The AUROC-OVA index, denoted by ρ_a is defined as:

$$\rho_a(Q_c) = \frac{1}{2c}\sum_{i=1}^{c}\left(1 + \frac{q_{ii}}{n_i} - \frac{r_i - q_{ii}}{n - n_i}\right). \tag{2.13}$$

5. The AURPC-OVA index, denoted by κ_a is defined as:

$$\kappa_a(Q_c) = \frac{1}{2c}\sum_{i=1}^{c}\left(\frac{q_{ii}}{r_i} + \frac{q_{ii}}{n_i}\right). \tag{2.14}$$

Similar to the two-class case, here also the indices reflect their informal description from Table 2.1 to their mathematical definition. Before proceeding further, we need to first prove three supporting lemmas, which respectively comment on the range of ACSA index, and highlights the key properties of AUROC-OVO and AUROC-OVA.

Lemma 2.1. *In a c-class classification problem the value of index α lies between 0, and 1.*

Proof. According to Definition 2.2, in a c-class confusion matrix $0 \leq q_{ii}/n_i \leq 1$, for all $i \in C$. Using this and the definition of α in (2.11), we can conclude that $0 \leq \alpha \leq 1$. Specifically, $\alpha = 0$, when the classifier wrongly classified every test point i.e. $q_{ii} = 0$, for all $i \in C$, and $\alpha = 1$, if the classifier correctly predicts the class label for each member of the test set. □

Lemma 2.2. *The value $\rho_o(Q_c)$ can be expressed as a linear function of $\alpha(Q_c)$, with a constant coefficient and bias both of which are dependent on c, as follows:*

$$\rho_o(Q_c) = \frac{c}{2(c-1)}\alpha(Q_c) + \frac{c-2}{2(c-1)}. \tag{2.15}$$

Moreover, the lower bound of $\rho_o(Q_c)$ *can be expressed as* $\mathcal{L}_{\mathbb{Q}_c}(\rho_o) = \frac{c-2}{2(c-1)}$*, while the upper bound* $\mathcal{U}_{\mathbb{Q}_c}(\rho_o) = 1$*.*

Proof. We first start with a $Q_c \in \mathbb{Q}_c$, then following from (2.12) after some algebraic manipulation we express $\rho_o(Q_c)$ as:

$$\begin{aligned}
\rho_o(Q_c) &= \frac{1}{2} + \frac{1}{2c}\sum_{i=1}^{c}\frac{q_{ii}}{n_i} - \frac{1}{2c(c-1)}\sum_{i=1}^{c}\sum_{j\in C\setminus\{i\}}\frac{q_{ji}}{n_j}, \\
&= \frac{1}{2} + \frac{1}{2c}\sum_{i=1}^{c}\frac{q_{ii}}{n_i} - \frac{1}{2c(c-1)}\sum_{i=1}^{c}\frac{n_i - q_{ii}}{n_i}, \\
&= \frac{1}{2} + \frac{1}{2c}\sum_{i=1}^{c}\frac{q_{ii}}{n_i} - \frac{1}{2(c-1)}\Big(1 - \frac{1}{c}\sum_{i=1}^{c}\frac{q_{ii}}{n_i}\Big), \\
&= \frac{1}{2} + \frac{\alpha(Q_c)}{2} - \frac{1}{2(c-1)}(1-\alpha(Q_c)), \\
&= \frac{c}{2(c-1)}\alpha(Q_c) + \frac{c-2}{2(c-1)}. \qquad (2.16)
\end{aligned}$$

Interestingly, from (2.16) we can conclude that for a given c, the index $\rho_o(Q_c)$ can be expressed as a linear function of $\alpha(Q_c)$, with a constant coefficient and a bias. Now Q_c is an arbitrary matrix belonging to the set $\mathbb{Q}_c$. Therefore, we can safely extend (2.16) to:

$$\mathcal{L}_{\mathbb{Q}_c}(\rho_o) = \frac{c}{2(c-1)}\mathcal{L}_{\mathbb{Q}_c}(\alpha) + \frac{c-2}{2(c-1)}, \qquad (2.17)$$

$$\text{and, } \mathcal{U}_{\mathbb{Q}_c}(\rho_o) = \frac{c}{2(c-1)}\mathcal{U}_{\mathbb{Q}_c}(\alpha) + \frac{c-2}{2(c-1)}. \qquad (2.18)$$

Plugging the values of $\mathcal{L}_{\mathbb{Q}_c}(\alpha), \mathcal{U}_{\mathbb{Q}_c}(\alpha)$ from Lemma 2.1, respectively in (2.17), and (2.18) we obtain.

$$\mathcal{L}_{\mathbb{Q}_c}(\rho_o) = \frac{c-2}{2(c-1)} \text{ and } \mathcal{U}_{\mathbb{Q}_c}(\rho_o) = 1,$$

which completes the proof. □

Lemma 2.3. *If we assume for simplicity, without loss of generality that* $n_1 \leq n_2 \cdots \leq n_c$*, then the lower bound of* $\rho_a(Q_c)$ *can be expressed as*

$$\mathcal{L}_{\mathbb{Q}_c}(\rho_a) = \frac{1}{2c}\left(c - 1 - \frac{n_c}{n - n_{c-1}}\right) \qquad (2.19)$$

while the upper bound $\mathcal{U}_{\mathbb{Q}_c}(\rho_a) = 1$.

Proof. If the classifier misclassifies all of the test points then $q_{ii} = 0, \forall i \in C$. However, as evident from (2.13) the value of $\mathcal{L}_{\mathbb{Q}_c}(\rho_a)$ is also dependent on the actual predictions as the the cost of misclassification to all the classes are not equal. To elaborate, we take a c-class confusion matrix Q_c, and construct Q'_c, such that $Q_c + B = Q'_c$, where, $B = [b_{ij}]_{c\times c}$, $b_{ij} \in \mathbb{Z}$, $\sum_{i=1}^{c} b_{ji} = 0$, $q'_{ij} = q_{ij} + b_{ij} \geq 0$, and $\sum_{i=1}^{c} b_{ij} = \bar{r}_i$, $\forall i, j \in C$, while conserving $n_i, \forall i \in C$, and n. Now, $\rho_a(Q'_c) - \rho_a(Q_c)$ can be calculated as

$$\rho_a(Q'_c) - \rho_a(Q_c) = \frac{1}{2c}\sum_{i=1}^{c}\frac{b_{ii}}{n_i} - \frac{1}{2c}\sum_{i=1}^{c}\frac{\bar{r}_i - b_{ii}}{n - n_i}. \tag{2.20}$$

Now for simplicity without loss of generality if we assume that $n_1 \leq n_2 \cdots \leq n_c$, then for any $i > j, \forall i, j \in C$, from (2.20) we can conclude that, increase in $\bar{r}_i - b_{ii}$ (i.e. the misclassification to other classes) will have larger effect on the value of $\rho_a(Q'_c)$ than $\bar{r}_j - b_{jj}$. In other words, the cost of misclassifications is higher for the majority class. Hence the value of $\rho_a(Q_c)$ will be minimum when all the points from classes other than c are wrongly predicted as class c, while the points from class c are misclassified as $c-1$. Hence, from (2.13) we get,

$$\mathcal{L}_{\mathbb{Q}_c}(\rho_a) = \frac{1}{2c}\Big(c - 1 - \frac{n_c}{n - n_{c-1}}\Big). \tag{2.21}$$

If the classifier correctly classifies all the test points then $r_i - q_{ii} = 0$, while $q_{ii}/n_i = 1, \forall i \in C$. Plugging these values in (2.13) gives $\mathcal{U}_{\mathbb{Q}_c}(\rho_a) = 1$, finishing the proof. □

We can now state the following theorem which investigates the behavior of different indices under the effect of varying RRT and number of classes.

Theorem 2.3. *Given a c-class classification problem:*

1. *The index* γ_c*, satisfies both of Condition 2.1 and 2.2.*
2. *The index* α *satisfies both of Condition 2.1 and 2.2.*
3. *The index* ρ_o *satisfies Condition 2.1 but not Condition 2.2.*
4. *The index* ρ_a *fails to satisfy both of Condition 2.1 and 2.2.*

5. The index κ_a satisfies Condition 2.2 but not Condition 2.1.

Proof. Let us consider two c-class confusion matrices Q_c and Q'_c. If we define Q_c as per Definition 2.2 then we can construct a new confusion matrix Q'_c by multiplying all the elements in the i^{th} row by a b_i, where $b_i \in \mathbb{R}^+$, $b_i q_{ij} \in \mathbb{Z}^+$, and $b_i \neq b_j; \forall i, j, \in C$, such that $Q_c \sim Q'_c$.

1) Using (2.10) we find the value of $\mathcal{V}_{Q'_c}(\gamma_c)$ as follows:

$$\mathcal{V}_{Q'_c}(\gamma_c) = \left(\prod_{i=1}^{c} \frac{b_i q_{ii}}{b_i n_i} \right)^{\frac{1}{c}} = \left(\prod_{i=1}^{c} \frac{q_{ii}}{n_i} \right)^{\frac{1}{c}} = \mathcal{V}_{Q_c}(\gamma_c).$$

Hence, it is proved that γ_c satisfies Condition 2.1.

Given a c-class confusion matrix Q_c, the $\gamma_c(Q_c)$ is only dependent on the values of q_{ii}, and n_i, as can be inferred from its definition in (2.10). Now, from Definition 2.2 we know that the values of $n_i > 0$, and $q_{ii} \geq 0$ (non-zero positive when at least one point from the class is correctly classified, 0 otherwise) for all $i \in C$. Thus, from (2.10), it is evident that $\gamma_c(Q_c) \geq 0$ (non-zero only when $q_{ii} > 0; \forall i \in C$), which implies that $\mathcal{L}_{\mathbb{Q}_c}(\gamma_c) = 0$.

Similarly, from the fact that $q_{ii} <= n_i; \forall i \in C$, as $n_i = \sum_{j=1}^{c} q_{ij}$, we can conclude that $0 \leq q_{ii}/n_i \leq 1$. Therefore, from (2.10) the value of $\mathcal{U}_{\mathbb{Q}_c}(\gamma_c)$ can found to be 1. Now, given the family of $c+1$-class confusion matrices, by the similar argument it can be shown that $\mathcal{L}_{\mathbb{Q}_{c+1}}(\gamma_c) = 0$, and $\mathcal{U}_{\mathbb{Q}_{c+1}}(\gamma_c) = 1$, which satisfies the Condition 2.2. This completes the first part of the theorem.

2) We take a c-class confusion matrix Q_c and construct Q'_c, such that they belong to the same equivalence class. We then find the value of $\mathcal{V}_{Q'_c}(\alpha)$ as per (2.10):

$$\mathcal{V}_{Q'_c}(\alpha) = \frac{1}{c} \sum_{i=1}^{c} \frac{b_i q_{ii}}{b_i n_i} = \frac{1}{c} \sum_{i=1}^{c} \frac{q_{ii}}{n_i} = \mathcal{V}_{Q_c}(\alpha).$$

Therefore, α satisfies Condition 2.1.

It is evident from Lemma 2.1 that $\mathcal{L}_{\mathbb{Q}_c}(\alpha) = 0$, $\mathcal{U}_{\mathbb{Q}_c}(\alpha) = 1$. By the same logic it can be claimed that $\mathcal{L}_{\mathbb{Q}_{c+1}}(\alpha) = 0$, $\mathcal{U}_{\mathbb{Q}_{c+1}}(\alpha) = 1$. Therefore, the lower and upper bound of α does not change with the increase in the number of classes, proving the second part of the theorem.

3) We start by a c-class confusion matrix Q_c and construct Q'_c ensuring that $Q_c \sim Q'_c$.

Now to confirm if ρ_o satisfies Condition 2.1, we find $\mathcal{V}_{Q'_c}(\rho_o)$, using (2.12) as follows:

$$\begin{aligned}\mathcal{V}_{Q'_c}(\rho_o) &= \frac{1}{2c}\sum_{i=1}^{c}\left(1+\frac{b_i q_{ii}}{b_i n_i}-\sum_{\substack{j=1\\ j\neq i}}^{c}\frac{b_j q_{ji}}{(c-1)b_j n_j}\right),\\ &=\frac{1}{2c}\sum_{i=1}^{c}\left(1+\frac{q_{ii}}{n_i}-\sum_{\substack{j=1\\ j\neq i}}^{c}\frac{q_{ji}}{(c-1)n_j}\right)=\mathcal{V}_{Q_c}(\rho_o).\end{aligned}$$

Thus we show the invariance of ρ_o under Condition 2.1.

We first start with a $Q_c \in \mathbb{Q}_c$, then following from Lemma 2.2, we get:

$$\mathcal{L}_{\mathbb{Q}_c}(\rho_o) = \frac{c-2}{2(c-1)} \text{ and } \mathcal{U}_{\mathbb{Q}_c}(\rho_o) = 1.$$

Approaching similarly for a $c+1$-class confusion matrix, we see that:

$$\mathcal{L}_{\mathbb{Q}_{c+1}}(\rho_o) = \frac{c-1}{2c} \neq \mathcal{L}_{\mathbb{Q}_c}(\rho_o), \tag{2.22}$$

$$\mathcal{U}_{\mathbb{Q}_{c+1}}(\rho_o) = 1 = \mathcal{U}_{\mathbb{Q}_c}(\rho_o). \tag{2.23}$$

Therefore, from (2.22), and (2.23), we conclude that the lower bound of ρ_o is dependent on the number of classes while the upper bound is remained at 1, violating Condition 2.2 and completing the third part of the theorem.

4) Similar to the previous approaches given a c-class confusion matrix Q_c, we construct Q'_c and express $\mathcal{V}_{Q'_c}(\rho_a)$ as follows:

$$\begin{aligned}\mathcal{V}_{Q'_c}(\rho_a) &= \frac{1}{2c}\sum_{i=1}^{c}\left(1+\frac{b_i q_{ii}}{b_i n_i}-\frac{\sum_{j=1}^{c} b_j q_{ji}-b_i q_{ii}}{\sum_{j=1}^{c} b_j n_j - b_i n_i}\right),\\ &= \frac{1}{2c}\sum_{i=1}^{c}\left(1+\frac{q_{ii}}{n_i}-\frac{\sum_{j=1}^{c} b_j q_{ji}-b_i q_{ii}}{\sum_{j=1}^{c} b_j n_j - b_i n_i}\right) \neq \mathcal{V}_{Q_c}(\rho_a).\end{aligned}$$

Hence, ρ_a do not satisfy Condition 2.1.

It is evident from Lemma 2.3 that for a $c+1$-class problem $\mathcal{U}_{\mathbb{Q}_{c+1}}(\rho_a) = \mathcal{U}_{\mathbb{Q}_c}(\rho_a) = 1$.

Moreover, similar to (2.21), we can calculate:

$$\mathcal{L}_{\mathbb{Q}_{c+1}}(\rho_a) = \frac{1}{2c+2}\Big(c - \frac{n_{c+1}}{n-n_c}\Big). \tag{2.24}$$

From, (2.21) and (2.24) we can show $\mathcal{L}_{\mathbb{Q}_c}(\rho_a) \neq \mathcal{L}_{\mathbb{Q}_{c+1}}(\rho_a)$, indicating that ρ_a does not satisfy Condition 2.2, which completes the fourth part of the theorem.

5) As previous we take a c-class confusion matrix Q_c, and construct Q'_c satisfying the equivalence relation. Let us now find $\mathcal{V}_{Q'_c}(\kappa_a)$ by (2.14),

$$\begin{aligned}\mathcal{V}_{Q'_c}(\kappa_a) &= \frac{1}{2c}\sum_{i=1}^{c}\left(\frac{b_i q_{ii}}{\sum_{j=1}^{c} b_j q_{ji}} + \frac{b_i q_{ii}}{b_i n_i}\right), \\ &= \frac{1}{2c}\sum_{i=1}^{c}\left(\frac{b_i q_{ii}}{\sum_{j=1}^{c} b_j q_{ji}} + \frac{q_{ii}}{n_i}\right) \neq \mathcal{V}_{Q_c}(\kappa_a).\end{aligned}$$

Therefore, we conclude that κ_a violates Condition 2.1.

We know from Definition 2.2, if q_{ii} becomes n_i for all $i \in C$, i.e. when all the points in the test set are correctly classified in their respective classes, then $r_i = q_{ii}; \forall i C$. On the other hand if all the test points are misclassified then $q_{ii} = 0$ for all $i \in C$. Consequently, $0 \leq q_{ii}/r_i \leq 1$, and $0 \leq q_{ii}/n_i \leq 1$, where both reaches the lower bound of 0 when $q_{ii} = 0$ (at the worst performance of the classifier) and the upper bound 1 when $q_{ii} = n_i$ (i.e. the classifier has achieved the best performance). Following this observation we can calculate $\mathcal{L}_{\mathbb{Q}_c}(\kappa_a) = \frac{1}{2}(0+0) = 0$ and $\mathcal{U}_{\mathbb{Q}_c}(\kappa_a) = \frac{1}{2}(1+1) = 1$. We can similarly find $\mathcal{L}_{\mathbb{Q}_{c+1}}(\kappa_a)$ and $\mathcal{U}_{\mathbb{Q}_{c+1}}(\kappa_a)$, which will be equal to their respective values for the set of c-class confusion matrices. Hence, κ_a satisfies Condition 2.2. □

If we consider the case of AUROC-OVO then it is evident from Lemma 2.2, that the lower limit of ρ_o gradually increases with the number of classes thus becomes affected by the Type 2 distortion. A solution to mitigate this problem is to apply a normalization to ρ_o, such that its lower bound can be made independent of C. This can be done by first subtracting the bias from ρ_o and then dividing the result by the coefficient (both terms are dependent on the choice of c) found in (2.15), which necessarily reduces the index to ACSA.

In a similar fashion, we can discuss the nature of AUROC-OVA as well. As per Lemma 2.3 the lower bound of the ρ_a index is dependent on the number of classes as well as on the

number of test points from the top two majority classes. Therefore, normalizing such an index will require to make certain assumptions on the representatives of the majority classes in the test set. In a special situation where the test set only contains $\frac{n}{2}$ points each from the top two majority classes, then $\mathcal{L}_{\mathbb{Q}_c}(\rho_a)$ reduces down to $\bar{\rho}_a(c) = \frac{c-2}{2c}$, which is a weak lower limit for ρ_a. We call the normalized AUROC-OVA, as nAUROC-OVA, and calculate it by first subtracting the reduced lower limit and then dividing the result by the difference between the reduced lower limit and unity. In other words nAUROC-OVA can be expressed as $\frac{\rho_a(Q_c)-\bar{\rho}_a(c)}{1-\bar{\rho}_a(c)}$.

The reason for violating Condition 2.1 by AURPC-OVA is the direct consideration of r_i (which involve the true as well as false predictions in the i^{th} class) in the precision counterpart. Therefore, we propose a modified AURPC-OVA such that while calculating the precision the q_{ji} values are properly scaled by their corresponding n_js, for all $j, i \in C$. We describe the modified AURPC-OVA called as mAURPC-OVA, in the following Definition 2.6.

Definition 2.6. *For a c-class confusion matrix Q_c the mAURPC-OVA index, denoted by $\hat{\kappa}_a$ is defined as:*

$$\hat{\kappa}_a(Q_c) = \frac{1}{2c}\sum_{i=1}^{c}\left(\frac{q_{ii}/n_i}{\sum_{j=1}^{c} q_{ji}/n_j} + \frac{q_{ii}}{n_i}\right). \tag{2.25}$$

We now proceed to confirm Condition 2.1, and 2.2 for mAURPC-OVA, in the following theorem.

Theorem 2.4. *The index $\hat{\kappa}_a$ satisfies both of the Conditions.*

Proof. Proceeding in a manner similar to the one taken for κ_a in Theorem 2.3, if we consider the two c-class confusion matrices Q_c and Q'_c, then $\mathcal{V}_{Q'_c}(\hat{\kappa}_a)$ can be expressed as follows:

$$\begin{aligned}\mathcal{V}_{Q'_c}(\hat{\kappa}_a) &= \frac{1}{2c}\sum_{i=1}^{c}\left(\frac{b_i q_{ii}/b_i n_i}{\sum_{j=1}^{c} b_j q_{ji}/b_j n_j} + \frac{b_i q_{ii}}{b_i n_i}\right), \\ &= \frac{1}{2c}\sum_{i=1}^{c}\left(\frac{q_{ii}/n_i}{\sum_{j=1}^{c} q_{ji}/n_j} + \frac{q_{ii}}{n_i}\right) = \mathcal{V}_{Q_c}(\hat{\kappa}_a),\end{aligned}$$

which indicates that $\hat{\kappa}_a$, satisfies Condition 2.1.

Similar to Theorem 2.3, we can see that $0 \leq q_{ii}/n_i \leq 1$, and $0 \leq \sum_{j=1}^{c} q_{ji}/n_j \leq 1$, for all $i, j, \in C$. Both of these terms reach their corresponding lower bound when classifiers

performs the worst and upper bound at the accurate classification as previously described in the fifth part of Theorem 2.3. Therefore, following Theorem 2.3, we can conclude that both of $\mathcal{L}_{\mathbb{Q}_c}(\hat{\kappa}_a) = 0 = \mathcal{L}_{\mathbb{Q}_{c+1}}(\hat{\kappa}_a)$ and $\mathcal{U}_{\mathbb{Q}_c}(\hat{\kappa}_a) = \mathcal{U}_{\mathbb{Q}_{c+1}}(\hat{\kappa}_a)$, hold implying that Condition 2.2 is satisfied by $\hat{\kappa}_a$. □

From Theorem 2.3 and 2.4 we can conclude that among all only GMean, ACSA, and AURPC-OVA satisfy both of Condition 2.1 and 2.2 and thus can be applicable to evaluate multi-class imbalanced classifiers in presence of varying RRT or number of classes. However, the question about the quality of the information provided by these indices under extremely poor classification performance on a single class is still required to be answered. Therefore, we proceed to the following Theorem 2.5 which evaluates the indices under the light of Condition 2.3.

Theorem 2.5. *Among the three indices which are immune to the two types of distortions, except γ_c, both of α and $\hat{\kappa}_a$ also satisfy Condition 2.3.*

Proof. To prove that γ_c fails to satisfy the third condition we start by finding $\mathcal{W}_{\mathcal{Q}_c(i)}(\gamma_c)$, which by (2.10) can be expressed as follows:

$$\mathcal{W}_{\mathcal{Q}_c(i)}(\gamma_c) = \lim_{\epsilon \to 0^+} \left(\epsilon \prod_{j=1, j\neq i}^{c} (1-\epsilon) \right)^{\frac{1}{c}} = \lim_{\epsilon \to 0^+} \epsilon^{\frac{1}{c}} (1-\epsilon)^{\frac{c-1}{c}} = 0. \tag{2.26}$$

From (2.26) and Theorem 2.3, we can see that for γ_c index $\mathcal{W}_{\mathcal{Q}_c(i)}(\gamma_c) = \mathcal{L}_{\mathbb{Q}_c}(\gamma_c)$, thus violating Condition 2.3.

We begin by calculating $\mathcal{W}_{\mathcal{Q}_c(i)}(\alpha)$, which according to (2.10) is as follows:

$$\begin{aligned} \mathcal{W}_{\mathcal{Q}_c(i)}(\alpha) &= \lim_{\epsilon \to 0^+} \frac{1}{c} \Big(\epsilon + \sum_{j=1, j\neq i}^{c} (1-\epsilon) \Big), \\ &= \lim_{\epsilon \to 0^+} \frac{\epsilon + (c-1)(1-\epsilon)}{c} = \frac{c-1}{c}. \end{aligned} \tag{2.27}$$

From (2.27) and Theorem 2.3, it is evident that α index $\mathcal{W}_{\mathcal{Q}_c(i)}(\alpha) > \mathcal{L}_{\mathbb{Q}_c}(\alpha)$, therefore, α satisfies Condition 2.3.

As per (2.6) the value of $\mathcal{W}_{\mathcal{Q}_c(i)}(\hat{\kappa}_a)$ can be calculate as:

$$\begin{aligned}
\mathcal{W}_{\mathcal{Q}_c(i)}(\hat{\kappa}_a) &= \lim_{\epsilon\to 0^+} \frac{1}{2c} \sum_{\substack{j=1 \\ j\neq i}}^{c} \left(1-\epsilon+\frac{1-\epsilon}{1-\epsilon+\frac{r_i-q_{jj}}{n-n_j}}\right) \\
&\quad + \lim_{\epsilon\to 0^+} \frac{1}{2c}\left(\epsilon+\frac{\epsilon}{\epsilon+\frac{r_i-q_{ii}}{n-n_i}}\right), \\
\Rightarrow \mathcal{W}_{\mathcal{Q}_c(i)}(\hat{\kappa}_a) &> \lim_{\epsilon\to 0^+} \frac{1}{2c} \sum_{\substack{j=1 \\ j\neq i}}^{c} \left(1-\epsilon+\frac{1-\epsilon}{1-\epsilon+\frac{n-n_j+n_j\epsilon}{n-n_j}}\right) \\
&\quad + \lim_{\epsilon\to 0^+} \frac{1}{2c}\left(\epsilon+\frac{\epsilon}{\epsilon+\frac{n-n_i\epsilon}{n-n_i}}\right) \\
\Rightarrow \mathcal{W}_{\mathcal{Q}_c(i)}(\hat{\kappa}_a) &> \frac{c-1}{2c}(1+\frac{1}{2})+0 = \frac{3(c-1)}{4c}.
\end{aligned} \tag{2.28}$$

From (2.28) and Theorem 2.4, it is evident that $\mathcal{W}_{\mathcal{Q}_c(i)}(\hat{\kappa}_a) > \mathcal{L}_{\mathbb{Q}_c}(\hat{\kappa}_a)$, confirming that the index $\hat{\kappa}_a$ satisfies Condition 2.3, which completes the proof. □

2.5 Experiments

This section first provides a brief description of the used dataset, followed by details of the experiment protocol, and finally illustrates the different results alongside appropriate discussion.

2.5.1 Description of datasets

We have used a subset of the widely popular ImageNet (Deng et al., 2009) classification dataset for all our experiments. The ImageNet dataset provides a vast collection of natural images categorized into a large number of structured classes (1000 leaf classes alongside 860 higher-level concepts following a predefined tree). For our experiments following the standard practice (Razavian et al., 2014; Nanni et al., 2017; Mahajan et al., 2018; Datta et al., 2019) we have taken a subset of the ImageNet training set by sampling images from 12 higher-level classes (formed by combining 1-5 leaf concepts and containing a total of 1300-6500 data instances). Since our chosen classifiers are only applicable to real-valued data, given the images, we need to extract quality features. Therefore, for the purpose of feature

extraction, we have used the state-of-the-art Inception V3 (Szegedy et al., 2016), an end-to-end deep neural network, which learns the map from the image space to a set of classes through a 2048-dimensional real-valued distributed representation space. The Inception V3 used by us is a standard implementation pre-trained on the complete ImageNet training set, publicly available from Keras deep learning API at `https://keras.io/applications`. Thus, in our case, the selected subset of images can be mapped to useful feature vectors by a simple forward pass through the pre-trained network. We have then created 12 two-class classification problems, having IR between 5 and 40 for experimentally validating the effect of Type 1 distortion over the various indices. Moreover, we have formed a total of 10 multi-class imbalanced datasets (4 sets for 3-class, while 3 sets each for the 5-class and 10-class classification problems) having IR between 20-30 for the purpose of empirically evaluating the effect of Type 2 distortion. A detailed description of the datasets used in our experiments can be found in Section A.2 of Appendix A.

2.5.2 Experiment protocol

We perform two sets of experiments respectively over two-class and multi-class imbalanced subsets of ImageNet to inspect the behavior of different indices in light of Condition 2.1 and Condition 2.2. We conduct our experiments using four state-of-the-art, classifiers of diverse nature, all specifically tailored for handling class-imbalance, namely Dual-LexiBoost with k-Nearest Neighbor as the base classifier (Datta et al., 2019), Near Bayesian SVM (NBSVM) (Datta and Das, 2015), RUSBoost (Seiffert et al., 2010; Japkowicz, 2000) with decision trees (Breiman, 2017) as the base classifier, and MLP (Rumelhart et al., 1986) combined with SMOTE (Chawla et al., 2002). The parameter settings of these methods can be found in Section A.3 of Appendix A.

For both experiments, the classifier is first trained with the training set and then tested by multiple test sets having different RRT values. This is done in an attempt to mimic the two primary causes of the first type of distortion as described in Section 2.1.3. In case of two-class datasets the RRT is varied between 1 (balanced), 0.5 (more number of minority class points are taken compared to the majority), half of the original IR (reduced effect of imbalance), the original IR of the training set, and twice of the original IR. Similarly, in case of multi-class datasets, we have used 5 different test sets with varying RRT (The first is balanced, in

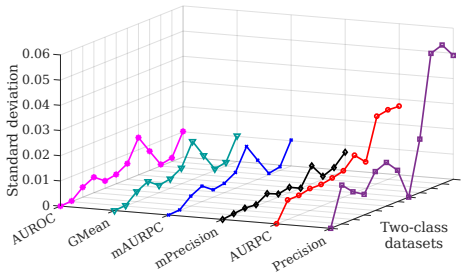


Figure 2.3: Effect of RRT on different indices over two-class imbalanced subsets of ImageNet.

the second the IR between the classes are reversed, in the third the original IR between the minority class and all others are halved, the fourth maintains the original IR, while the last doubles the test points from all classes except the minority). Such an experimental setting helps us to understand the effect of the varying number of test points from different classes on the values of the indices. Additionally, the experiments on multi-class datasets provide us with a way to inspect the effect of increasing c on the indices. Average results over 5 independent runs of 5-fold stratified cross-validation (which also aids in parameter tuning for the classifiers) are reported to ensure the reliability of our findings.

2.5.3 Validating the two-class classification performance evaluation indices in light of Condition 2.1

For each of the datasets, we have found the standard deviation of the mean performance in terms of an index over the five test sets and four classifiers. We plot the findings in Figure 2.3, which shows that the Precision index achieves the highest variability over the test sets for a given training set. However, the low standard deviation of GMean and AUROC suggests that the classifiers retain an almost similar performance over the various test sets. Therefore, the high standard deviation of Precision must be due to the changes in the actual numbers of the respective test points from the two classes, which vary significantly due to the diverse choice of RRT. These observations reflect the theoretical analysis which shows Precision to be sensitive over RRT even when the class-specific classification performance is retained, thus failing to satisfy Condition 2.1. Due to having Precision as a component AURPC also suffers from the same issue, though the additional consideration of Recall helps to mitigate the effect of altering RRT to some extent. Interestingly, mPrecision and mAURPC closely follow the

GMean and AUROC indices indicating their immunity against the effect of RRT.

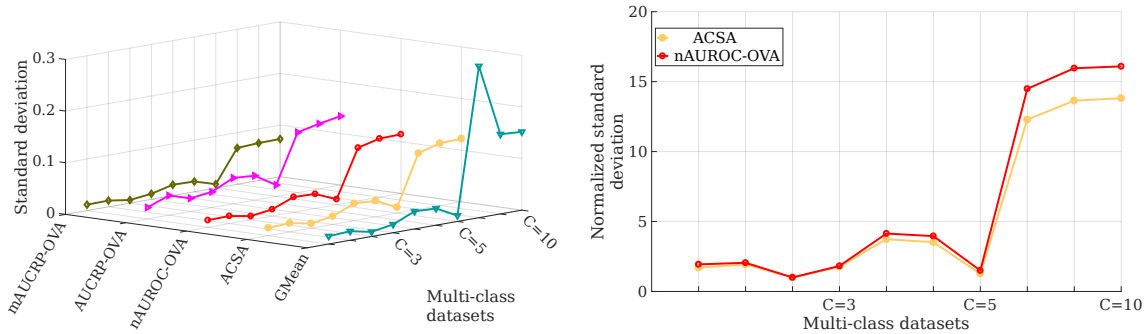


Figure 2.4: Analysis of index behavior over multi-class imbalanced subsets of ImageNet under Condition 2.1. (a) Effect of RRT on different indices over multi-class imbalanced subsets of ImageNet. (b) Stability of ACSA compared to nAUROC-OVA.

2.5.4 Validating the multi-class classification performance evaluation indices in light of Condition 2.1

We use an approach similar to the two-class case for validating Condition 1 for the multi-class performance evaluation indices. However, the indices which are susceptible to Condition 2 are expected to have a smaller range with an increasing value of c, and may result in a lower standard deviation over the test sets for a higher number of classes. Thus, comparing these indices with those indices satisfying Condition 2 may lead to a bias against the later and will not help to reach a conclusive remark. Hence, in Figure 2.4a, we only compare the standard deviations of indices satisfying Condition 2.2, viz. GMean, ACSA, nAUROC-OVA, AURPC-OVA, and mAURPC-OVA, over the various test sets for each of the datasets. A close inspection reveals that the minimum variability (especially improving from AURPC-OVA) is achieved by mAURPC-OVA establishing it as the better choice among the five contenders. Interestingly, ACSA has shown slightly higher variability compared to nAUROC-OVA, which is unlikely as the later violates Condition 2.1. This leads us to investigate further, by normalizing the standard deviation of the ACSA and nAUROC-OVA indices for each of the datasets by the respective minimum standard deviation achieved overall the multi-class datasets. This kind of normalized standard deviation can be considered as a measure of stability as it quantifies the variability of an index from its best stable performance (a lower

value signifies that the index can equivalently evaluate similar performing classifiers). We plot the results in Figure 2.4b, which shows the normalized standard deviation to be slightly greater for nAUROC-OVA than that of ACSA. This indicates nAUROC-OVA to be less stable compared to ACSA, and the lower standard deviation of the former in Figure 2.4a may be due to the fact that AUROC-OVA is normalized using a weak lower bound.

2.5.5 The effect of the number of classes (Condition 2.2) over the different indices

We consider AUROC-OVA and AUROC-OVO for this experiment as they are seen to have a higher lower bound with an increasing number of classes. Their respective normalized version, i.e. nAUROC-OVA and ACSA are also considered to establish the improvement achieved through normalization, alongside GMean as a reference. We plot the minimum value achieved by these indices for each of the datasets in Figure 2.5. The results show that on three and five class datasets the AUROC-OVA, and AUROC-OVO performs almost equivalently to their normalized counterparts. However, on the ten class datasets the minimum index value achieved by AUROC-OVA and AUROC-OVA are significantly higher than ACSA, and nAUROC-OVA. This validates the bias of AUROC-OVA and AUROC-OVO towards a higher value with an increasing number of classes, and also demonstrated the ability of the respective normalized versions to counter this bias.

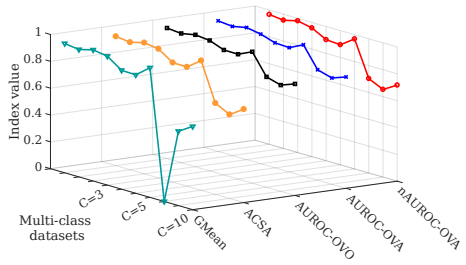


Figure 2.5: Effect of the number of classes on different indices over multi-class imbalanced subsets of ImageNet.

2.5.6 The effect of Condition 2.3 on multi-class indices

From Figure 2.5 the minimum values of GMean are consistent with the other indices over the three, and five class datasets. However, for the ten-class datasets, the index produced

significantly lower values compared to the others. Moreover, on a ten-class dataset GMean produced its lowest possible value of 0, indicating the worst possible classification performance. However, the values of the other indices over the same dataset clearly indicate that the classifier managed to successfully classify many of the test points. Therefore, despite satisfying Conditions 2.1 and 2.2, GMean fails to do the same for Condition 2.3 as poor performance on a single class results in the loss of all information about the performance on every other class.

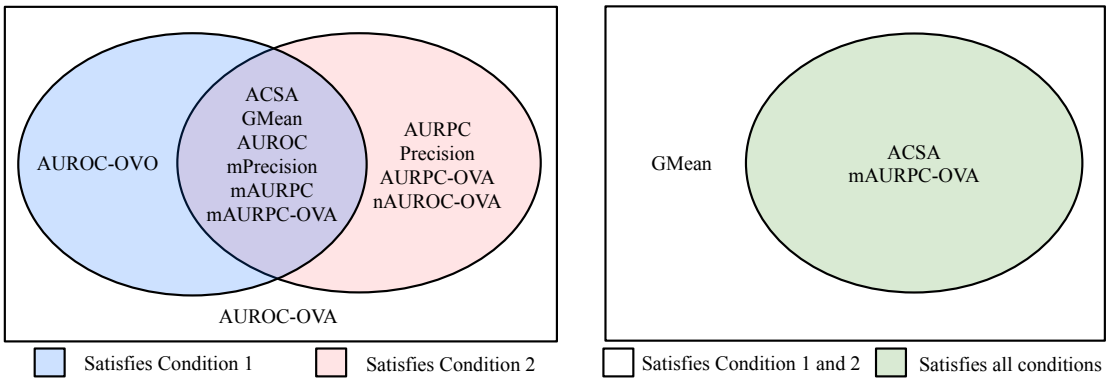


Figure 2.6: A summary of the different conditions satisfied by each of the indices under concern. (a) Summary of findings documented by Theorem 2.1, 2.2, 2.3, and 2.4, i.e. validation of indices under Condition 2.1, and 2.2. (b) Summary of findings in Theorem 2.5, i.e. validation of the indices under the light of Condition 2.3, which satisfy the fundamental properties and applicable to multi-class classification problems.

2.6 Applicability of indices

Based on the satisfaction of the two fundamental conditions the indices can be grouped as shown in Figure 2.6a. Moreover, the multi-class indices which satisfy Condition 2.1, and 2.2, are further classified by Condition 2.3 in Figure 2.6b. Therefore, using Figure 2.6 we can proceed to recommend an appropriate choice of indices for different applications.

In the case of two-class classification, all four of GMean, AUROC, mPrecision, and mAURPC satisfy Condition 1, thus any one of these can be a good index of choice. However, GMean is biased towards the accuracy of that class which is poorly classified compared to the other. In a two-class scenario, this property of GMean may prove useful as it will identify

the high bias of a classifier towards a particular class. AUROC, on the other hand, accords equal weight to the performance in both classes. Therefore, we recommend GMean for a general evaluation of the performance of a two-class classifier.

Recall and Precision (consequently AURPC, mPrecision, and mAURPC) both depend on the choice of the positive class. Recall is focused on the classification performance over the minority class, thus can be used in applications where false positives do not lead to severe consequences. For example, we may consider the case of benign and malignant tumor classification in medical diagnostic systems, where wrongly classifying a sample from the minority class of malignant tumors may result in a fatal outcome. On the other hand, Precision (and AURPC) can be effectively used when the application attempts to limit the number of false positives while the class priors do not significantly vary over time. One can think of the spam filtering problem where even though the non-spam emails are considerably high in numbers, labeling one of them as spam may lead to loss of important information. Evidently, mPrecision and mAURPC indices can act as the respective replacement of Precision and AURPC if the application under concern can cause Type 1 distortion.

In the case of multi-class classification, even though GMean satisfies both of Condition 2.1 and 2.2 it may still be biased in case of extremely poor performance over a single class, as indicated by its violation of Condition 2.3. GMean can however still prove beneficial if the target is to achieve non-zero classification accuracy on each class. On the contrary, ACSA and mAURPC satisfy all three of the conditions, and thus any of the two can be an appropriate choice of index. Finally, despite their violation of Condition 2.1, nAUROC-OVA and AURPC-OVA can be used in those applications where the misclassification from different classes are associated with different costs. For example, in a multi-class medical diagnostic application a somewhat similar set of syndromes may correspond to different diseases of varying severity and rarity.

2.7 Discussion

In this chapter, we showed that the common indices used for evaluating the performance of a classifier in presence of class imbalance may suffer from different forms of distortions, depending on the character of the data, especially the test set. We formally defined two

conditions that an index needs to satisfy to be resilient to such distortions. We presented theoretical analyses detailing the traits of the indices in light of these conditions and proposed necessary remedies as per need. We further defined a third condition to evaluate the quality of the information provided by an index, especially under adverse conditions such as exceptionally poor accuracy over a single class. We also undertook empirical analysis to support our theoretical findings. Finally, we discussed on the applicability of different indices and make recommendations.

Chapter 3

Convergence of the Class Membership Estimator in Fuzzy k-Nearest Neighbor Classifier on Balanced and Imbalanced Datasets

Summary

The question of validating the quality of the class memberships estimated by Fuzzy k-nearest neighbor (FkNN) for a regular multi-class classification problem still remains mostly unanswered. In this chapter, we attempt to address this issue by first proposing a different direction of evaluating a fuzzy classifier which shifts the focus from the misclassification error of FkNN to the class memberships estimated by it. This leads us to find novel theoretical upper bounds respectively on the bias and the mean squared error of the class memberships estimated by FkNN, which attest to the convergence of the estimated class memberships towards their corresponding ideals with the increasing availability of labeled data points, under some elementary assumptions on the class distribution and membership function. We also demonstrate that the proposed analysis can provide an explanation for the vulnerability of FkNN to class imbalance. A detailed simulation study on artificial and real datasets, as well as a performance comparison with the state-of-the-art, are presented to empirically support our claims. In this chapter, we also investigate the usefulness of coupling a points-specific locally adaptive class weights with the weighted variant of FkNN in an attempt to improve the performance of the classifier in the presence of class imbalance. Comparison of performance with the FkNN variants tailored to handle class imbalance on real-world benchmark imbalanced datasets validates

the efficacy of the proposed.

3.1 Introduction

3.1.1 Overview

From its introduction by Fix and Hodges Jr in 1951 the kNN classifier has always enjoyed popularity among the machine learning community for its methodical simplicity, non-parametric working principle (Duda et al., 2000), and ease of implementation. Further, kNN being a lazy learner does not engage in any learning activity before a query has been made to the classifier, making it readily integrable to an application.

Given a training set X and a test instance $\mathbf{y}$, the kNN classifier first finds the set $\mathbb{V}_k^X(\mathbf{y}) = \{\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_k\}$ of the k nearest neighbors (defined by some distance metric) of $\mathbf{y}$ in X. For simplicity we assume that the elements of $\mathbb{V}_k^X(\mathbf{y})$ are ordered by their respective distances from $\mathbf{y}$, i.e. $\mathbf{v}_i$ (where $i = 1, 2, \cdots, k$) is the i-th nearest neighbor of $\mathbf{y}$ in X. The classifier then labels the test point $\mathbf{y}$ with that class which has the majority of representatives in $\mathbb{V}_k^X(\mathbf{y})$. To elaborate, let us first consider an indicator function $\mathcal{I}$ which given an input condition returns 1 if the condition is found to be true and 0 otherwise. Then the kNN classification rule can be expressed as follows:

$$\hat{h}^X(\mathbf{y}) = \arg\max_{j \in C} \sum_{\mathbf{v} \in \mathbb{V}_k^X(\mathbf{y})} \mathcal{I}(h(\mathbf{v}) = j). \tag{3.1}$$

Even though the kNN classifier provides a commendable performance on numerous applications it still comes with its fair share of limitations. First, in kNN each of the k neighbors plays an equal role in deciding the class label for $\mathbf{y}$ irrespective of their distance from $\mathbf{y}$. Second, kNN does not account for the implicit uncertainty in the class distributions estimated from a finite number of available training points. The FkNN classifier attempts to address these issues of regular kNN by providing less importance to the distant neighbors and employing the theory of fuzzy sets (Zadeh, 1965). Similar to kNN (Muja and Lowe, 2014; Garcia-Pedrajas et al., 2015; Anava and Levy, 2016; Gallego et al., 2018), the FkNN classifier also gained popularity and interest among the research community (Derrac et al., 2014). In one direction of research, attempts were made to improve the classification per-

formance of FkNN (Yang and Chen, 1998; Sarkar, 2007; Derrac et al., 2016; Ezghari et al., 2017); while the other tailored FkNN for various real-world applications (Hu et al., 1998; Frigui and Gader, 2009) and data specific problems (Maillo et al., 2017).

In a c-class classification problem, FkNN considers each class as a fuzzy set and given a test point $\mathbf{y}$, it estimates a membership $\hat{u}_j(\mathbf{y})$ for the j-th class (where $j \in C$), such that $\sum_{j=1}^{c} \hat{u}_j(\mathbf{y}) = 1$. In other words, the class membership estimator (hereafter called just estimator) employed by FkNN estimates the degree of membership of $\mathbf{y}$ for each of the c classes instead of crisply assigning it to only one of them. The FkNN estimator is defined as:

$$\hat{u}_j(\mathbf{y}) = \frac{\sum_{\mathbf{v} \in \mathbb{V}_k^X(\mathbf{y})} u_j(\mathbf{v}) ||\mathbf{y} - \mathbf{v}||^{\frac{2}{1-m}}}{\sum_{\mathbf{v} \in \mathbb{V}_k^X(\mathbf{y})} ||\mathbf{y} - \mathbf{v}||^{\frac{2}{1-m}}}, \tag{3.2}$$

where, m is a newly introduced parameter which scales down the contributions of the distant neighbors in making the decision for $\mathbf{y}$. According to the study of Keller et al. (1985), the performance of FkNN does not alter significantly with varying m. Therefore, one can treat m as a constant and use its conventional choice of $m = 2$ to obtain a good performance on a wide variety of datasets. Furthermore, FkNN assumes that the class memberships for the training points i.e. $u_j(\mathbf{x})$, for all $\mathbf{x} \in X$ and $j \in C$, are known in advance. In practice one may find $u_j(\mathbf{x})$ by either utilizing domain knowledge to make certain assumptions on the corresponding class distribution, or using non-parametric kernel density estimation techniques (Silverman, 1988). Moreover, Keller et al. (1985) proposed a widely popular and efficient (Villar et al., 2016; Maillo et al., 2017) heuristic method which estimates $u_j(\mathbf{x})$ by exploiting the information extracted from the locality of $\mathbf{x}$. If we assume that among the k' training set neighbors of $\mathbf{x}$, the number of points belonging to the j-th class is k'_j, then the Keller's heuristic estimates $u_j(\mathbf{x})$ as follows:

$$u_j(\mathbf{x}) = \begin{cases} 0.51 + \left(\frac{k'_j}{k'}\right) \times 0.49 & \text{if } \mathbf{x} \in X_j, \\ \left(\frac{k'_j}{k'}\right) \times 0.49 & \text{otherwise.} \end{cases} \tag{3.3}$$

Both of the regular kNN and FkNN classifiers give equal importance to each of the classes while taking the decision for a test instance. However, depending on the specific nature of the problem an equal weighting of classes may not be proven beneficial. For example, one

may choose to compensate for the dearth of minority class representatives in an imbalanced training set by giving minority neighbors higher weights during the decision making (Tan, 2005). This can be achieved by a weighted variant of the kNN classifier where every class is associated with a weight and the number of points belonging to a certain class is multiplied by the weight associated with that class. The rest is similar to the conventional kNN decision rule, resulting in the assignment of the new point to the class having the maximum weighted number of members in the $\mathbb{V}_k^X(\mathbf{y})$. If the weight for the j-th class is denoted by ω_j, then the weighted kNN classification rule can be expressed as:

$$\hat{h}^X(\mathbf{y}) = \underset{j\in 1,2,\cdots,c}{\arg\max} \sum_{\mathbf{v}\in\mathbb{V}_k^X(\mathbf{y})} \omega_j \mathcal{I}(h(\mathbf{v}) = j), \tag{3.4}$$

where the weights ω_j for all $j \in C$ can be any positive real number. The weighted kNN classifier is actually a more general form of the kNN classifier, which can be reduced to the basic kNN classifier by making all the weights equal. The concept of weighted kNN can be directly extended to FkNN as well (WFkNN), by simply modifying the estimated class membership $\hat{u}_j(\mathbf{y})$ of a test point $\mathbf{y}$ in the j-th class as follows:

$$\hat{u}_j^{\Omega}(\mathbf{y}) = \frac{\omega_j \hat{u}_j(\mathbf{y})}{\sum_{j=1}^{c} \omega_j \hat{u}_j(\mathbf{y})} \tag{3.5}$$

3.1.2 Background

In this section, we first discuss the previous performance analysis of kNN and FkNN, following which we briefly describe the different studies which investigated the effect of class imbalance on the two classifiers under concern.

3.1.2.1 Performance analysis of kNN and FkNN

The first milestone research on the theoretical analysis of kNN was conducted by Cover and Hart (1967) who showed that the rate of misclassification of 1NN can be at most twice that of the Bayes decision rule in the Euclidean space. Furthermore, they commented on the preferable value of parameter k for achieving a good performance by stating that the choice of k should be varied with n ensuring $k \to \infty$ and $\frac{k}{n} \to 0$ as $n \to \infty$. In a subsequent work, Cover (1968) extended the result in (Cover and Hart, 1967) for an infinite number of

classes, which was further improved by Wagner (1971). Recently, Bax (2000) investigated the generalized error of 1NN by using the concept of validation and later extended it by proposing probably approximately correct error bounds (Bax, 2012).

Yang and Chen (1991) made a milestone contribution by proposing a theoretical analysis for the FkNN classifier. They demonstrated that the bound established by Cover (1968) holds in the case of F1NN as well. However, this significant attempt still has its limitations, as the authors restricted themselves to a more elementary form of the original problem. Firstly, instead of considering the consistency of a general FkNN the authors only focused on the simpler case of $k = 1$. Secondly, the analysis was performed for a bounded loss function. Thirdly, the approach employed by the authors involved a more complicated Lebesgue dominated convergence (Yang and Chen, 1991). Finally, the authors did not perform any simulation study to empirically validate their theoretical conclusions. In this chapter, we aim to address these issues and develop a simple yet commendable validation of the performance of FkNN using two unbounded loss functions.

3.1.2.2 Addressing class imbalance in kNN and FkNN

After the formal introduction of class imbalance problem by Kubat et al. in 1997 a couple of primary works (Zhang and Mani, 2003; Tan, 2005) investigated its effect on kNN classifiers. On one hand, Zhang and Mani (2003) attempted to effectively compensate for the dearth of minority training instances by undersampling the majority class. On the other hand, the idea of weighted kNN using class-specific weights to counter the adversarial effect of class imbalance was first explored by Tan (2005) and later studied in further detail by Wang et al. (2008), Liu and Chawla (2011), and Dubey and Pudi (2013). In other directions of research, Li and Zhang (2011) considered extending some of the minority points to Gaussian balls to improve their abundance. Further, (Kriminger et al., 2012; Zhang and Li, 2013; Zhang et al., 2017) investigated the importance of the neighborhood information in better estimation of the posterior probability for each of the classes. However, all of these methods involve an exhaustive search and/or introduce new tunable parameters, which hinder the scalability and easy implementation of kNN by burdening the user with a significant computational overhead.

Even though a collection of notable variants was proposed to improve the classification

accuracy of the canonical kNN in presence of class imbalance, the same cannot be said about FkNN. In fact, to the best of our knowledge, there is no notable work that investigated the effect of class imbalance on FkNN and propose remedial measures. In this chapter, we attempt to fill this gap by designing a WFkNN which is likely to improve the resilience of the classifier against class imbalance. Moreover, multiple studies (Wettschereck and Dietterich, 1994; Garcia-Pedrajas et al., 2015; Anava and Levy, 2016) highlighted the benefits of accounting locality information in the decision process of kNN, which is usually performed by adapting the value of the parameter k. In this chapter, we instead attempt to explore the effectiveness of incorporating the locality information through point specific adaptive class weights.

3.1.3 Motivation

The approach of validating the performance of a classifier by comparing its generalization error with that of the Bayes decision rule may fail for FkNN. This is due to the fact that the primary goal of FkNN is to estimate the class memberships of a given test point, from which consequently its class label can be decided. Hence, if the error between the original class membership and that estimated by the FkNN reduces on average, then the misclassification risk will decrease. On the other hand, multiple fuzzy classifiers can still achieve similar classification accuracy while the estimated class membership may lie far away from the corresponding original. We illustrate this idea by the following Example 3.1.

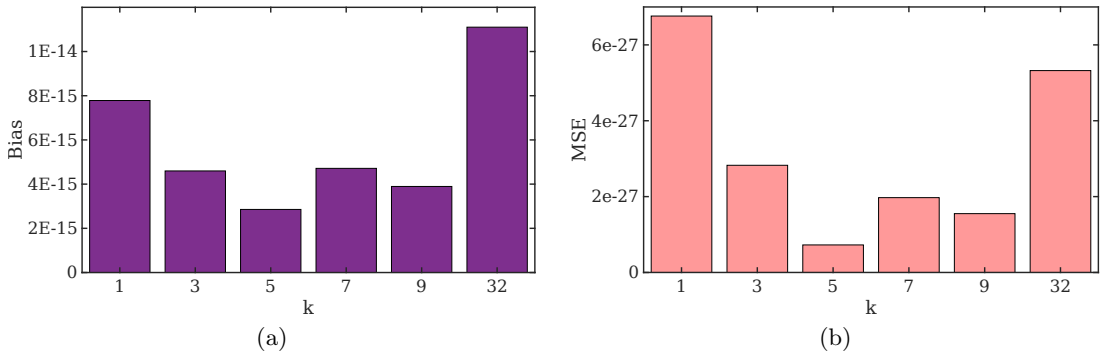


Figure 3.1: Similar classification performance does not imply similar convergence of bias and MSE. We are given a set of classifiers all of which achieve a 100% classification accuracy over the test set X'. (a) Even when the classifiers performed equivalently the respective bias did not converge to the same extent. (b) Similar can be observed for MSE as well which converged differently for distinct classifiers.

Example 3.1. *We take a 2-dimensional, 2-class classification problem, where the class distributions are defined as bivariate Gaussian. Both of the classes are sampled from normal distributions having an identity covariance matrix, while the mean for the first class is* $[-5, 0]$*, and that for the second class is* $[5, 0]$*. We generate a training set* X *and a test set* X'*, by respectively collecting* 10^3 *and 100 data points from each of the two classes. The initial membership for the training points as well as the ground truth for the test points can be easily calculated as the underlying class distributions are known. Thus, given a data point, its membership in a class is actually the probability for that point to be sampled from that class. Using this experimental setup, we classify* X' *by FkNN using* $k = 1, 3, 5, 7, 9, 11$*, and* 32 *(the conventional maximum value of* k *is taken as* $\lceil\sqrt{N}\rceil = \lceil\sqrt{10^3}\rceil$*). We present the bias and Mean Squared Error (MSE) of FkNN estimator for the positive class for each of the choices of* k *respectively in Figure 3.1a and Figure 3.1b, when all the FkNN classifiers achieved 0 misclassification error i.e. they are 100% accurate on* X'*. We can observe that all the classifiers though performed equally in terms of accuracy, the minimum bias and MSE are only achieved by FkNN when* $k = 5$*. This indicates that even when the estimated memberships do not converge to the corresponding true values, a fuzzy classifier may still be able to perform well.*

From the observation made in Example 3.1, a stronger validation of the performance of FkNN can be achieved by focusing on the loss functions such as bias (measuring the quality of the estimated class membership) and MSE (indicating the consistency of FkNN) of the estimator. As an added advantage, both of the loss functions have a positive and unbounded range. We specifically show that both of the bias and MSE of the FkNN estimator are upper bounded when a certain class of membership functions is used and the choice of k follows N as per the suggestion of Cover and Hart (1967).

3.1.4 Contributions of Chapter 3

Over the years, only a handful of studies shed light on the properties of FkNN using theoretical analysis. This is mostly due to the late development of FkNN as well as its additional complexity. This chapter aims to bridge this gap by introducing a novel direction for analyzing the behavior of FkNN.

Instead of following the traditional path of validating a fuzzy classifier's performance in terms of misclassification error, this chapter proposes a new direction to evaluate the accuracy of the estimated class memberships for a test point. As illustrated in Section 3.1.3, such an approach may be more informative, as it directly evaluates the ability of a fuzzy classifier to achieve its primary goal. Therefore, we focus on the bias and Mean Squared Error (MSE) of the FkNN estimator and attempt to establish their convergence under some elementary assumptions on the class distributions. As an additional advantage, both of the chosen loss functions are positive and unbounded, and their convergence towards zero implies a correct estimation of the class memberships and consequently an accurate classification with high probability.

In this chapter, we propose two fairly straightforward theorems (and deduce two corresponding corollaries) the first of which describes the convergence of the loss functions in two-class cases, while the other directly extends the result for multi-class scenarios. Additionally, the proposed theorems can be considered as reasonably robust as they are applicable to a wide class of membership functions and probability distributions. Moreover, as a major improvement over the previous works the current study is not bound to any particular value of k or m, and only requires satisfying some elementary assumptions on the choice of k corresponding to N.

This chapter further discusses on the implications of our proposed analysis in explaining the susceptibility of FkNN in the presence of class imbalance. As a remedy, we suggest employing a point specific Locally Adaptive Class Weighting (LACW) which can aid FkNN to offer better immunity against the effects of class imbalance. Contrary to a global class weighting strategy that assigns the same class weights to all test instances, the LACW can consider the locality information of a test point during adaptively calculating the class weights using a simple heuristic. This is expected to be beneficial as the useful locality information (Anava and Levy, 2016) of a $\mathbf{y} \in X'$ is not otherwise supplied to a FkNN classifier.

The rest of the chapter contains some preliminaries in Section 3.2, and the proposed analysis in Section 3.3. We present a detailed simulation study using both artificial and real datasets in Section 3.4, which also contains a performance evaluation of the LACW in conjunction with FkNN compared to FkNN variants tailored to handle class imbalance.

3.2 Preliminaries

In this section, we first discuss on the basic assumptions which are to be ensured for the proposed theoretical bounds to be applicable. Further, we list the previously found mathematical results which are necessary to proceed with our analysis.

3.2.1 Assumptions

The proposed upper bounds respectively for the bias and MSE of the estimator of FkNN make three elementary assumptions about the nature of the two associated parameters, the membership function, and the data distribution. The first assumption deals with the choices of parameters m and k. Whereas, the second and third assumptions respectively ensure some desirable properties for the membership function and data distribution.

Assumption 3.1. *The value of m should be greater than 1, while k should be varied as per the suggestion of Cover and Hart (1967) i.e. $k \to \infty$ and $\frac{k}{N} \to 0$ as $N \to \infty$.*

Assumption 3.2. *For a c-class classification problem, given a dataset $\mathbb{S}$, let the class-specific gradients of the membership function with respect to $\mathbf{s} \in \mathbb{S}$ be denoted as $\nabla u_j(\mathbf{s})$. Then $\nabla u_j(\mathbf{s})$ should exist for all $j = 1, 2, \cdots, c$ while their corresponding norm defined as $||\nabla u_j(\mathbf{s}))||$ should be upper bounded by some $A_0 > 0$.*

Assumption 3.3. *Let us assume that the data points belonging to $\mathbb{S}$ are identically and independently sampled from some arbitrary distribution defined by a probability density function p. Then the first-order differential of p should exist.*

3.2.2 Necessary result

Our analysis will require the following result found by Mack and Rosenblatt (1979) which given a d-dimensional dataset $\mathbb{S}$ containing S number of data instances comments on the expected distance between a data instance $\mathbf{s}$ and its k-th neighbour $\mathbf{v}_k$ in the set $\mathbb{S} \setminus \{\mathbf{s}\}$.

Result 3.1 (Mack and Rosenblatt (1979))**.** *If we assume that the dataset $\mathbb{S}$ is sampled from some probability distribution having density function p then the expected distance between the*

neighbors $\mathbf{s}$ *and* $\mathbf{v}_k$ *for all* $A_1 \neq 0$ *can be found as:*

$$E(||\mathbf{s}-\mathbf{v}_k||^{A_1}) = \left(\frac{k/S}{v_d p(\mathbf{s})}\right)^{\frac{A_1}{d}} + o\left((k/S)^{\frac{A_1}{d}}\right),$$

where v_d *is the volume of a d-dimensional sphere of unit radius.*

3.3 The convergence of FkNN class membership estimator

We prove the convergence of FkNN in terms of bias and MSE with the help of a couple of theorems. The first theorem establishes the proposed bounds for two-class classification problems while the second extends the theory for general multi-class situations.

3.3.1 Convergence for two-class classification problems

The convergence of FkNN for the two-class classification problem can be illustrated by the following Theorem 3.1.

Theorem 3.1. *Let us consider a two-class classification problem, where given a test point* $\mathbf{y}$*, its true membership to the* j*-th class and that estimated by FkNN, are denoted by* $u_j(\mathbf{y})$ *and* $\hat{u}_j(\mathbf{y})$*, respectively. Then under the aforementioned assumptions, as* $N \to \infty$*, the following can be shown to be true,*

$$\begin{aligned} &1) \quad E(|\hat{u}_j(\mathbf{y}) - u_j(\mathbf{y})|) \to 0, \forall j \in \{1,2\}, \\ &2) \quad E(|\hat{u}_j(\mathbf{y}) - u_j(\mathbf{y})|^2) \to 0 \forall j \in \{1,2\}. \end{aligned}$$

Proof. In case of a two-class classification problem, $u_2(\mathbf{s}) = 1 - u_1(\mathbf{s})$, $\forall \mathbf{s} \in \mathbb{S}$. Such a relationship would also be maintained while estimating the membership by FkNN for some $\mathbf{s} \in \mathbb{S}$. Therefore, $\hat{u}_2(\mathbf{s}) = 1 - \hat{u}_1(\mathbf{s})$. This enables us to safely assume that any bound established on the bias and MSE of the estimator for the first class will also hold true for the second class as well. Hence, for simplicity and without loss of generality we can prove the theorem only considering the first class.

Let us first define $m' = \frac{2}{1-m}$ and calculate the value of $\hat{u}_1(\mathbf{y})$ by expression (3.2). We can

now proceed to finding the bias of the estimator of FkNN for the first class.

$$|\hat{u}_1(\mathbf{y}) - u_1(\mathbf{y})| = \left| \frac{\sum_{\mathbf{v}\in\mathbb{V}_k^X(\mathbf{y})} u_1(\mathbf{v})||\mathbf{y}-\mathbf{v}||^{m'}}{\sum_{\mathbf{v}\in\mathbb{V}_k^X(\mathbf{y})} ||\mathbf{y}-\mathbf{v}||^{m'}} - u_1(\mathbf{y}) \right|,$$
$$= \left| \frac{b_1}{b_2} - u_1(\mathbf{y}) \right|. \tag{3.6}$$

Expanding b_1 in expression (3.6) by the Mean Value Theorem (MVT) (Apostol, 1964), we obtain,

$$b_1 = \sum_{\mathbf{v}\in\mathbb{V}_k^X(\mathbf{y})} \left(u_1(\mathbf{y}) + (\mathbf{y}-\mathbf{v})^T \nabla u_1(\boldsymbol{\delta}_\mathbf{v})\right) ||\mathbf{y}-\mathbf{v}||^{m'}, \tag{3.7}$$

where $\boldsymbol{\delta}_\mathbf{v} \in \mathcal{B}(\mathbf{y}, ||\mathbf{y}-\mathbf{v}||)$ for all $\mathbf{v} \in \mathbb{V}_k^X(\mathbf{y})$ while $\mathcal{B}(\mathbf{y}, ||\mathbf{y}-\mathbf{v}||)$ denotes a d-dimensional ball centered at $\mathbf{y}$ with a radius of $||\mathbf{y}-\mathbf{v}||$. Replacing the value of b_1 from expression (3.7) into expression (3.6) we obtain,

$$\begin{aligned}
|\hat{u}_1(\mathbf{y}) - u_1(\mathbf{y})| &= \left| u_1(\mathbf{y}) - u_1(\mathbf{y}) + \frac{1}{b_2} \sum_{\mathbf{v}\in\mathbb{V}_k^X(\mathbf{y})} (\mathbf{y}-\mathbf{v})^T \nabla u_1(\boldsymbol{\delta}_\mathbf{v}) ||\mathbf{y}-\mathbf{v}||^{m'} \right| \\
&= \left| \frac{1}{b_2} \sum_{\mathbf{v}\in\mathbb{V}_k^X(\mathbf{y})} (\mathbf{y}-\mathbf{v})^T \nabla u_1(\boldsymbol{\delta}_\mathbf{v}) ||\mathbf{y}-\mathbf{v}||^{m'} \right|, \\
&= \left| \frac{1}{b_2} \sum_{\mathbf{v}\in\mathbb{V}_k^X(\mathbf{y})} \frac{(\mathbf{y}-\mathbf{v})^T}{||\mathbf{y}-\mathbf{v}||} \nabla u_1(\boldsymbol{\delta}_\mathbf{v}) ||\mathbf{y}-\mathbf{v}||^{1+m'} \right|, \\
&\leq \frac{1}{b_2} \sum_{\mathbf{v}\in\mathbb{V}_k^X(\mathbf{y})} ||\nabla u_1(\boldsymbol{\delta}_\mathbf{v})|| \times ||\mathbf{y}-\mathbf{v}||^{1+m'}, \\
&\leq \frac{\sum_{\mathbf{v}\in\mathbb{V}_k^X(\mathbf{y})} ||\nabla u_1(\boldsymbol{\delta}_\mathbf{v})|| \times ||\mathbf{y}-\mathbf{v}||^{1+m'}}{k||\mathbf{y}-\mathbf{v}_1||^{m'}}.
\end{aligned} \tag{3.8}$$

Now according to Assumption 3.2 the norm of the gradient $||u_1(\boldsymbol{\delta}_\mathbf{v})|| \leq A_0$ for all $\mathbf{v} \in \mathbb{V}_k^X(\mathbf{y})$. Placing this value back in expression (3.8) we get,

$$|\hat{u}_1(\mathbf{y}) - u_1(\mathbf{y})| \leq \frac{A_0 \sum_{\mathbf{v}\in\mathbb{V}_k^X(\mathbf{y})} ||\mathbf{y}-\mathbf{v}||^{1+m'}}{k||\mathbf{y}-\mathbf{v}_1||^{m'}}. \tag{3.9}$$

Now taking the expectation for both sides of expression (3.9),

$$\begin{aligned} E\left(|\hat{u}_1(\mathbf{y}) - u_1(\mathbf{y})|\right) \leq & E\left(\frac{A_0 \sum_{\mathbf{v}\in\mathbb{V}_k^X(\mathbf{y})} ||\mathbf{y}-\mathbf{v}||^{1+m'}}{k||\mathbf{y}-\mathbf{v}_1||^{m'}}\right) \\ \leq & \frac{A_0}{k} \sum_{\mathbf{v}\in\mathbb{V}_k^X(\mathbf{y})} E\left(\frac{||\mathbf{y}-\mathbf{v}||^{1+m'}}{||\mathbf{y}-\mathbf{v}_1||^{m'}}\right). \end{aligned} \tag{3.10}$$

Applying the Cauchy Schwarz inequality (Gut, 2005) on the Right Hand Side (RHS) of expression (3.10) we get,

$$\begin{aligned} E\left(|\hat{u}_1(\mathbf{y}) - u_1(\mathbf{y})|\right) \leq & \frac{A_0}{k} \sum_{\mathbf{v}\in\mathbb{V}_k^X(\mathbf{y})} \left[E\left(||\mathbf{y}-\mathbf{v}||^{\frac{2m-6}{m-1}}\right) E\left(||\mathbf{y}-\mathbf{v}_1||^{\frac{4}{m-1}}\right)\right]^{\frac{1}{2}}, \\ \leq & A_0 \left[E\left(||\mathbf{y}-\mathbf{v}_k||^{\frac{2m-6}{m-1}}\right) E\left(||\mathbf{y}-\mathbf{v}_1||^{\frac{4}{m-1}}\right)\right]^{\frac{1}{2}}. \end{aligned} \tag{3.11}$$

We can now reduce the RHS of (3.11) using Result 3.1,

$$\begin{aligned} E\left(|\hat{u}_1(\mathbf{y}) - u_1(\mathbf{y})|\right) \leq & A_0 \left(\left(\frac{k/N}{v_d p(\mathbf{y})}\right)^{\frac{m-3}{d(m-1)}} + o\left((k/N)^{\frac{m-3}{d(m-1)}}\right)\right) \\ & \times \left(\left(\frac{1/N}{v_d p(\mathbf{y})}\right)^{\frac{2}{d(m-1)}} + o\left((1/N)^{\frac{2}{d(m-1)}}\right)\right), \\ \leq & A_0(b_3 + b_4 + b_5 + b_6), \end{aligned} \tag{3.12}$$

$$\text{where, } b_3 = \left(\frac{k^{\frac{m-3}{m-1}}/N}{v_d p(\mathbf{y})}\right)^{\frac{1}{d}}, \tag{3.12a}$$

$$b_4 = \left(\frac{k}{v_d p(\mathbf{y})}\right)^{\frac{m-3}{d(m-1)}} o\left((1/N)^{\frac{1}{d}}\right), \tag{3.12b}$$

$$b_5 = \left(\frac{1}{v_d p(\mathbf{y})}\right)^{\frac{2}{d(m-1)}} o\left(\left(k^{\frac{m-3}{m-1}}/N\right)^{\frac{1}{d}}\right), \tag{3.12c}$$

$$b_6 = o\left(\left(k^{\frac{m-3}{m-1}}/N\right)^{\frac{1}{d}}\right). \tag{3.12d}$$

Now, according to Assumption 3.1, if $m > 1$, then $\frac{m-3}{m-1} < 1$. Thus, $k^{\frac{m-3}{m-1}} < k$, for all $k \geq 1$, which implies that b_3 in the expression (3.12a) goes to 0 as $k/N \to 0$ with $N \to \infty$. By a similar logic, b_4, b_5, and b_6 all go to 0 as well when $N \to \infty$. Therefore, from (3.12), we

get

$$E\left(|\hat{u}_1(\mathbf{y}) - u_1(\mathbf{y})|\right) \to 0 \text{ as } N \to \infty. \tag{3.13}$$

This concludes the proof for the first part of the theorem.

For the second part of the theorem we follow a path similar to the one used for proving the bound on bias. Thus, using the MVT on MSE of the estimator we get:

$$(|\hat{u}_1(\mathbf{y}) - u_1(\mathbf{y})|^2) \leq \left(\frac{\sum_{\mathbf{v}\in\mathbb{V}_k^X(\mathbf{y})} ||\nabla u_1(\boldsymbol{\delta}_\mathbf{v})|| \times ||\mathbf{y}-\mathbf{v}||^{1+m'}}{k||\mathbf{y}-\mathbf{v}_1||^{m'}} \right)^2. \tag{3.14}$$

Now finding the expectation of (3.14) and proceeding similarly as the previous,

$$\begin{aligned}
E(|\hat{u}_1(\mathbf{y}) - u_1(\mathbf{y})|^2) \leq & E\left(\frac{A_0 \sum_{\mathbf{v}\in\mathbb{V}_k^X(\mathbf{y})} ||\mathbf{y}-\mathbf{v}||^{1+m'}}{k||\mathbf{y}-\mathbf{v}_1||^{m'}} \right)^2, \\
\leq & \frac{A_0^2}{k^2} E\left(\sum_{\mathbf{v}\in\mathbb{V}_k^X(\mathbf{y})} \frac{||\mathbf{y}-\mathbf{v}||^{1+m'}}{||\mathbf{y}-\mathbf{v}_1||^{m'}} \right)^2, \\
\leq & \frac{A_0^2}{k^2} E\left(\frac{k||\mathbf{y}-\mathbf{v}_k||^{1+m'}}{||\mathbf{y}-\mathbf{v}_1||^{m'}} \right)^2, \\
\leq & A_0^2 E\left(\frac{||\mathbf{y}-\mathbf{v}_k||^{1+m'}}{||\mathbf{y}-\mathbf{v}_1||^{m'}} \right)^2.
\end{aligned} \tag{3.15}$$

Applying Cauchy Schwartz inequality on the RHS of expression (3.15),

$$E(|\hat{u}_1(\mathbf{y}) - u_1(\mathbf{y})|^2) \leq A_0^2 \left[E\left(||\mathbf{y}-\mathbf{v}_k||^{\frac{4m-12}{m-1}} \right) E\left(||\mathbf{y}-\mathbf{v}_1||^{\frac{8}{(m-1)}} \right) \right]^{\frac{1}{2}}. \tag{3.16}$$

Applying Result 3.1 on the RHS of expression (3.16) we get,

$$\begin{aligned}
E(|\hat{u}_1(\mathbf{y}) - u_1(\mathbf{y})|^2) \leq A_0^2 & \left(\left(\frac{k/N}{v_d p(\mathbf{y})} \right)^{\frac{2m-6}{d(m-1)}} + o\left((k/N)^{\frac{2m-6}{d(m-1)}} \right) \right) \\
& \times \left(\left(\frac{1/N}{v_d p(\mathbf{y})} \right)^{\frac{4}{d(m-1)}} + o\left((1/N)^{\frac{4}{d(m-1)}} \right) \right).
\end{aligned} \tag{3.17}$$

Similar to the case of bias, it can be shown that under Assumption 3.1, as $N \to \infty$ the expression (3.17) goes to 0, thus completing the proof. □

Using Theorem 3.1, a stronger convergence of FkNN can be shown in the form of the following Corollary 3.1.

Corollary 3.1. *For a two-class classification problem the class memberships estimated by FkNN for a test point* $\mathbf{y}$ *converge in probability to the corresponding true memberships, as* $N \to \infty$.

Proof. In Theorem 3.1 we have shown that as $N \to \infty$, the following holds true for a $j \in \{1, 2\}$,

$$E(|\hat{u}_j(\mathbf{y}) - u_j(\mathbf{y})|) \to 0. \tag{3.18}$$

Now for some small $A_2 > 0$, Markov inequality (Gut, 2005) states that,

$$Pr(|\hat{u}_j(\mathbf{y}) - u_j(\mathbf{y})| > A_2) \leq \frac{E|\hat{u}_j(\mathbf{y}) - u_j(\mathbf{y})|}{A_2}. \tag{3.19}$$

From expression (3.18) and (3.19) it is evident that,

$$\hat{u}_j(\mathbf{y}) \xrightarrow{p} u_j(\mathbf{y}).$$

This completes the proof. □

3.3.2 Convergence for multi-class classification problems

The convergence of FkNN for the multi-class classification problem can be established by the following theorem.

Theorem 3.2. *Let us consider a c-class classification problem, where given a test point* $\mathbf{y}$*, its true membership to the j-th class and that estimated by FkNN, are denoted by* $u_j(\mathbf{y})$ *and* $\hat{u}_j(\mathbf{y})$*, respectively. Then under the aforementioned assumptions as* $N \to \infty$*, the following hold true,*

$$1) \quad E(|\hat{u}_j(\mathbf{y}) - u_j(\mathbf{y})|) \to 0, \textit{where } j = 1, 2, \cdots c,$$
$$2) \quad E(|\hat{u}_j(\mathbf{y}) - u_j(\mathbf{y})|^2) \to 0, \textit{where } j = 1, 2, \cdots c.$$

Proof. Both of bias and MSE of an estimator can only have positive range, which bound their

respective minimum value to 0. Therefore, it is sufficient to show that the respective sum of biases and MSEs over all the classes converge to 0, which necessarily indicate that bias and MSE of the estimator for the individual classes have converged to 0 as well. Thus, we reformulate the problem to prove that the following two statements hold true, when $N \to \infty$,

$$\mu_e = \sum_{j=1}^{c} E(|\hat{u}_j(\mathbf{y}) - u_j(\mathbf{y})|) \to 0, \tag{3.20}$$

$$\sigma_e = \sum_{j=1}^{c} E(|\hat{u}_j(\mathbf{y}) - u_j(\mathbf{y})|^2) \to 0. \tag{3.21}$$

We first start with the Left Hand Side (LHS) of expression (3.20) and proceed in a manner similar to Theorem 3.1. Thus, after applying MVT and Assumption 3.2 we reach to the following,

$$\begin{aligned}
\mu_e =& \sum_{j=1}^{c} E(|\hat{u}_j(\mathbf{y}) - u_j(\mathbf{y})|) \\
\leq& A_0 \sum_{j=1}^{c} E\left(\frac{\sum_{\mathbf{v} \in \mathbb{V}_k^X(\mathbf{y})} ||\mathbf{y} - \mathbf{v}||^{1+m'}}{k||\mathbf{y} - \mathbf{v}_1||^{m'}} \right), \\
\leq& \frac{cA_0}{k} E\left(\frac{\sum_{\mathbf{v} \in \mathbb{V}_k^X(\mathbf{y})} ||\mathbf{y} - \mathbf{v}||^{1+m'}}{||\mathbf{y} - \mathbf{v}_1||^{m'}} \right).
\end{aligned} \tag{3.22}$$

Applying Cauchy Schwartz inequality followed by Result 3.1 on the RHS of expression (3.22) as in Theorem 3.1 we obtain,

$$\begin{aligned}
\mu_e = \sum_{j=1}^{c} E(|\hat{u}_j(\mathbf{y}) - u_j(\mathbf{y})|) \leq cA_0 & \left(\left(\frac{k/N}{v_d p(\mathbf{y})} \right)^{\frac{m-3}{d(m-1)}} + o\left((k/N)^{\frac{m-3}{d(m-1)}} \right) \right) \\
& \times \left(\left(\frac{1/N}{v_d p(\mathbf{y})} \right)^{\frac{2}{d(1-m)}} + o\left((1/N)^{\frac{2}{d(m-1)}} \right) \right).
\end{aligned} \tag{3.23}$$

Applying Assumption 3.1 on the RHS of expression (3.23) we reach to expression (3.20), from which the first part of the theorem can be concluded.

The second part of the theorem can be shown to be true as well by starting from expression

(3.21) and following a path similar to Theorem 3.1. Thus,

$$\begin{aligned}\sigma_e = \sum_{j=1}^{c} E(|\hat{u}_j(\mathbf{y}) - u_j(\mathbf{y})|^2) &\leq cA_0^2 E\left(\frac{||\mathbf{y}-\mathbf{v}_k||^{1+m'}}{||\mathbf{y}-\mathbf{v}_1||^{m'}}\right)^2, \\ &\leq cA_0^2\left(\left(\frac{k/N}{v_d p(\mathbf{y})}\right)^{\frac{2m-6}{d(m-1)}} + o\left((k/N)^{\frac{2m-6}{d(m-1)}}\right)\right), \\ &\times\left(\left(\frac{1/N}{v_d p(\mathbf{y})}\right)^{\frac{4}{d(m-1)}} + o\left((1/N)^{\frac{4}{d(m-1)}}\right)\right) \end{aligned} \tag{3.24}$$

Applying Assumption 3.1 on expression (3.24) we can show (3.21) to be true, which completes the proof. □

Similar to the case of two-class classification, here also we can deduce a corollary which proves a stronger convergence of the FkNN estimator.

Corollary 3.2. *In a c-class classification problem as $N \to \infty$, given a test point $\mathbf{y}$,*

$$\hat{u}_j(\mathbf{y}) \xrightarrow{p} u_j(\mathbf{y});\ \textit{where, } j = 1, 2, \cdots c.$$

Proof. This can directly be shown with the help of Markov inequality in a manner similar to the one presented in Corollary 3.1. Therefore, using the convergence of bias for individual classes from Theorem 3.2, we can infer that the class memberships estimated for $\mathbf{y}$ by FkNN converges with probability to the corresponding true memberships. □

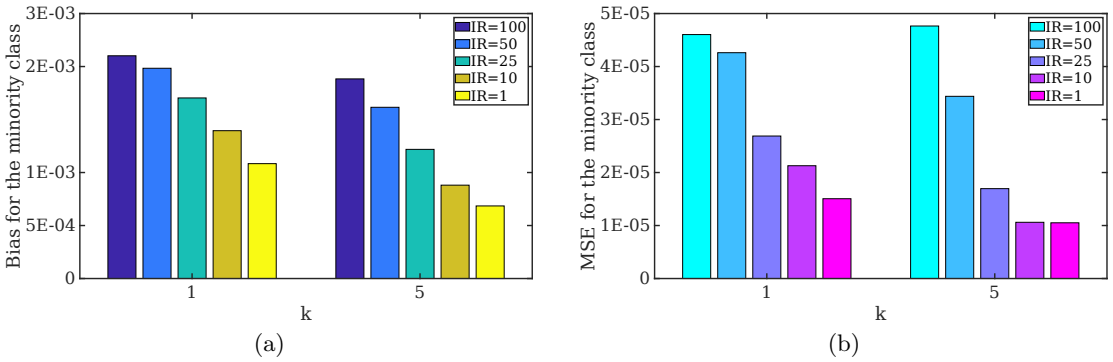


Figure 3.2: The effect of increasing class imbalance on the bias and MSE of the minority class.(a) The effect of class imbalance on bias of the minority class. (b) The effect of class imbalance on MSE of the minority class.

3.3.3 FkNN in presence of class imbalance

3.3.3.1 Discussion on the effect of class imbalance on FkNN under the light of Theorem 3.1, and Theorem 3.2

The two presented theorems and their associated corollaries indicate that the FkNN classifier can achieve better performance with the increasing availability of training samples. Moreover, if we convert the fuzzy class memberships to crisp class labeling by assigning a test point to the class with maximum membership, then the corollaries imply that a better classification accuracy may be reached with a high probability as the loss functions converge.

Furthermore, it is well known that kNN fails to achieve commendable accuracy in presence of class imbalance (Branco et al., 2016; Das et al., 2018), which is expected to be reflected in FkNN as well given the two share a similar philosophy. This shortcoming of FkNN can be explained under the light of Theorem 3.1 (and Theorem 3.2 for c-class classification), which can be considered as a significant practical implication of the theory. To better illustrate the susceptibility of FkNN in the presence of class imbalance we present the following Example 3.2.

Example 3.2. *We take a two-class classification problem, where both of the classes follow a normal distribution with the identity covariance matrix. However, for the distribution of the first class, the mean is* $[-2\ 0]$*, while that for the second class is* $[2\ 0]$*. Moreover, we construct five training sets by varying the IR, as 100, 50, 25, 10, and 1 (i.e. we combine 10000 points selected from the first class, and 100, 200, 400, 1000, and 10000 instances respectively sampled from the second class to achieve the desired IR in the corresponding training set). The test set is built by sampling 100 data points from each of the two classes. The membership of a point in a class is calculated by finding the probability of that point to belong in that class. With this setup we plot the bias and MSE for the minority class against the varying IR, while choosing the value of k as 1 and 5, respectively in Figure 3.2a and Figure 3.2a. Interestingly, both of the bias and MSE for the minority class improve with the decreasing IR and become minimum when the two classes are balanced. This can be directly explained by Theorem 3.1, as the number of minority class (and consequently the size of the training set) representatives increase with improving IR, it facilitates the loss functions to further converge.*

To elaborate, a class imbalanced dataset will always have a lesser number of training

points than a corresponding balanced one or in other words, N will decrease with increasing IR. Thus, the membership estimated by FkNN according to Theorem 3.1 and 3.2 will be more distant from their true values with the increasing imbalance between the classes. This in consequence is likely to lead to a higher misclassification by FkNN especially for the minority class as indicated by Corollary 3.1 and 3.2. As a remedy, similar to kNN one may use a class-specific weighting (Tan, 2005) to rectify the estimated class memberships, or introduce synthetic minority data instances (Chawla et al., 2002) to mitigate the effect of class imbalance by increasing the value of N. However, as the synthetic sampling contains the risk of contaminating the training set here we choose the alternative of combating class imbalance through class weighting. However, a global weighting strategy may not be fully effective in the absence of locality information of a test point (which is not readily available to the regular FkNN). Thus, in the following Section 3.3.3.2 we describe LACW, which can provide a set of adaptive class weights while taking the locality information of the test instance into account.

3.3.3.2 Point-specific locally adaptive class weights

A conventional Global Class Weighting (GCW) strategy which can be considered as folklore these days is to use a normalized inverse of the class priors as the class weights; i.e. the weight ω_j for the j-th class for all $j \in C$ is defined as $\frac{N}{cN_j}$. The intuition behind such a weighting scheme is to offer each of the classes a fair chance by providing them an opportunity to reach the ideal class prior $\frac{1}{c}$ of a balanced dataset from their corresponding imbalanced one of $\frac{N_j}{N}$ for all $j \in C$. Effectively GCW strategy when coupled with WFkNN will increase the membership to the minority classes while dragging down the same for the majority classes for all $\mathbf{y} \in X'$, compensating the disparity in the number of training points from different classes in an imbalanced X.

However, the conventional GCW strategy is incapable of providing a point specific set of weights while taking the locality information of a test point $\mathbf{y}$ into account. This limitation of the GCW strategy may lead the classifier to overcompensate in certain localities of the dataset resulting in a higher number of false positives. One may address this issue by calculating the weights not on the entire training set but on a neighborhood of the test point $\mathbf{y} \in X'$. We make a reasonable assumption that if a test point $\mathbf{y}$ lie in a dense region of the dataset then a larger number of training set neighbors should actively participate in calculating the

set of class weights for it. In contrast, if $\mathbf{y} \in X'$ resides in a sparse section of the dataset then a smaller number of neighboring training points should play a role in finding its suitable class weights. This allows us to further provide point specific information to the WFkNN classifier by adaptively determining the size of the neighborhood $\nu(\mathbf{y})$ using the local density around $\mathbf{y}$. However, instead of actually calculating or approximating the local density around a point one may choose to use an indicator that can be easily computed. According to Result 3.1 one such indicator can be the distance $\delta(\mathbf{y}, X)$ between the test point $\mathbf{y}$ and its nearest neighbor in the training set X. Thus, a small value of $\delta(\mathbf{y}, \mathbf{X})$ suggests that $\mathbf{y}$ lies in a dense neighborhood indicating that a high value of $\nu(\mathbf{y})$ should be chosen. On the other hand, if $\delta(\mathbf{y}, \mathbf{X})$ is found to be large then $\mathbf{y}$ is expected to reside in a sparse region which suggests selecting a lower value for $\nu(\mathbf{y})$.

We may now proceed to describe a simple heuristic technique to actually compute the value of $\nu(\mathbf{y})$. Given a training set X we start by defining a couple of constants:

$$\delta_{max} = \max_{\mathbf{x} \in X} \delta(\mathbf{x}, X \setminus \{\mathbf{x}\}),$$
$$\delta_{min} = \min_{\mathbf{x} \in X} \delta(\mathbf{x}, X \setminus \{\mathbf{x}\}).$$

Now given a test point $\mathbf{y} \in X'$ one of the three following situations can arise.

1. $\delta(\mathbf{y}, X) \leq \delta_{min}$ which indicates that $\mathbf{y}$ lies in a significantly dense region and $\nu(\mathbf{y})$ should be set to the maximum allowable neighborhood size ν_{max}.

2. $\delta(\mathbf{y}, X) \geq \delta_{max}$ which suggests that $\mathbf{y}$ resides in a sufficiently sparse location and $\nu(\mathbf{y})$ should be taken as the minimum allowable neighborhood size ν_{min}.

3. For any other value of $\delta(\mathbf{y}, X)$ between δ_{min} and δ_{max} the corresponding $\nu(\mathbf{y})$ should lie between ν_{min} and ν_{max}, while ensuring that if $\delta(\mathbf{y}, X) \leq \delta(\mathbf{y}', X)$ then $\nu(\mathbf{y}) \geq \nu(\mathbf{y}')$ for all $\mathbf{y}, \mathbf{y}' \in X'$.

The ν_{max} and ν_{min} are data dependent parameters the value of which should be large enough to offer a fair chance to all the classes for appearing as a neighbor. We respectively set ν_{max} and ν_{min} to N and $\sqrt{N}$ as these value are empirically found to provide a consistently superior performance over a wide variety of datasets. We calculate $\nu(\mathbf{y})$ for a $\delta(\mathbf{y}, X)$ lying between δ_{min} and δ_{max} as follows:

$$\nu(\mathbf{y}) = \left\lfloor \nu_{lin}(\mathbf{y})\nu_{exp}(\mathbf{y})^{\frac{1}{2}} \right\rfloor, \tag{3.25}$$

$$\text{where, } \nu_{lin}(\mathbf{y}) = \delta(\mathbf{y}, X)\frac{\nu_{min} - \nu_{max}}{\delta_{max} - \delta_{min}} + \frac{\nu_{max}\delta_{max} - \nu_{min}\delta_{min}}{\delta_{max} - \delta_{min}}, \tag{3.26}$$

$$\text{and } \nu_{exp}(\mathbf{y}) = \nu_{max} + (\nu_{max} - \nu_{min})\frac{e^{\delta_{min}} - e^{\delta(\mathbf{y},X)}}{e^{\delta_{max}} - e^{\delta_{min}}}. \tag{3.27}$$

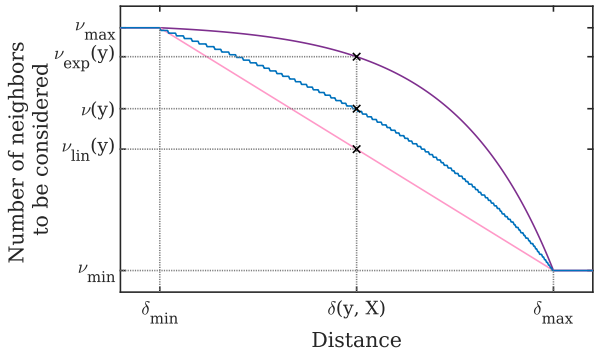


Figure 3.3: The curves for the linear estimate (in pink), concave exponential estimate (in magenta), and final $\nu(\mathbf{y})$ as the geometric mean of the two estimates (in blue), when $\delta(\mathbf{y}, X)$ is varied between δ_{min} and δ_{max}. We can observe that the linear model provides a restrictive estimate compared to the exponential, while the geometric mean of the two can be used to strike a balance.

To elaborate, we first perform a linear as well as a concave exponential estimate of $\nu(\mathbf{y})$ and then calculate the actual value of $\nu(\mathbf{y})$ by taking the geometric mean of the two estimates (floor value is chosen to ensure that the neighborhood size is an integer). The linear estimate models $\nu(\mathbf{y})$ as a linear function of $\delta(\mathbf{y}, X)$ between the two extremities $(\delta_{min}, \nu_{max})$ and $(\delta_{max}, \nu_{min})$. Such an estimate is useful to restrict the size of the neighborhood when N becomes large enough. On the other hand, the concave exponential model encourages a larger neighborhood as it estimates a higher value of $\nu(\mathbf{y})$ compared to its linear counterpart. This estimation is useful for moderate and small scaled dataset where the linear estimate may excessively shrink the neighborhood size. The final value of $\nu(\mathbf{y})$ is calculated as the geometric mean of the linear and exponential estimates striking a balance between the two. We empirically found that geometric mean which by construction produces a value closer to the linear estimate achieves a better performance compared to the arithmetic mean over datasets of diverse scale. Once the size of the neighborhood is estimated one can calculate the

weight ω_j for the j-th class for all $j \in C$ as $\frac{\nu(\mathbf{y})}{c\nu(\mathbf{y})_j}$, where $\nu(\mathbf{y})_j$ is the number of training points belonging to the j-th class in the $\nu(\mathbf{y})$ neighborhood of $\mathbf{y}$. Thereafter, one may estimate the class memberships using the WFkNN as described in (3.5).

3.4 Experiments

In this section, we first detail the simulation study to empirically validate our proposed analysis. Further, we provide a comparative study on real-world datasets to evaluate the efficacy of LACW in aiding WFlkNN to improve the classification accuracy in the presence of class imbalance.

3.4.1 Simulation study for validating Theorems 3.1 and 3.2

We start by describing the datasets used in this simulation study. We then detail the experimental protocol before proceeding to the results and subsequent discussions.

3.4.1.1 Description of the datasets used in the simulation study

We have used a collection of 9 artificial as well as 12 real-world datasets in this study.

Artificial Datasets: We design 9 artificial classification datasets each of which is d-dimensional and can have c classes, where $d \in \{2, 5, 10\}$ and $c \in \{2, 5, 9, 10, 15, 20\}$. An artificial dataset is formed by combining 200,000 d-dimensional data instances randomly sampled from each of the c classes, where the corresponding class distribution p is considered as multivariate Gaussian with a predefined mean and covariance matrix. We briefly highlight the key properties of the artificial datasets in Table 3.1, while a detailed description along with the respective generating technique can be found in Section B.1 of Appendix B. Moreover, for better visualization, we graphically illustrate the 2-dimensional datasets in Figure 3.4, where each of the classes is differently colored.

In case of an artificial dataset, it is trivial to label a data instance by the class from which it was originally generated. Furthermore, designing a suitable membership function which will satisfy Assumption 3.2, while reflecting the ideal membership to the original class, is also not difficult considering our freedom of choosing the appropriate distributions for individual classes. Therefore, given a data point $\mathbf{s}$ from a d-dimensional c-class artificial dataset $\mathbb{S}$, the

Table 3.1: Properties of the artificial datasets.

Dataset Name	Number of Dimensions	Number of Classes	Remark (if any)
AD-2-2-NO	2	2	No overlap
AD-2-2-SO	2	2	Small overlap
AD-2-2-LO	2	2	Large overlap
AD-9-2	2	9	No overlap
AD-15-2	2	15	No overlap
AD-5-5	5	5	Uncontrolled overlap
AD-10-5	5	10	Uncontrolled overlap
AD-10-10	10	10	Uncontrolled overlap
AD-20-10	10	20	Uncontrolled overlap

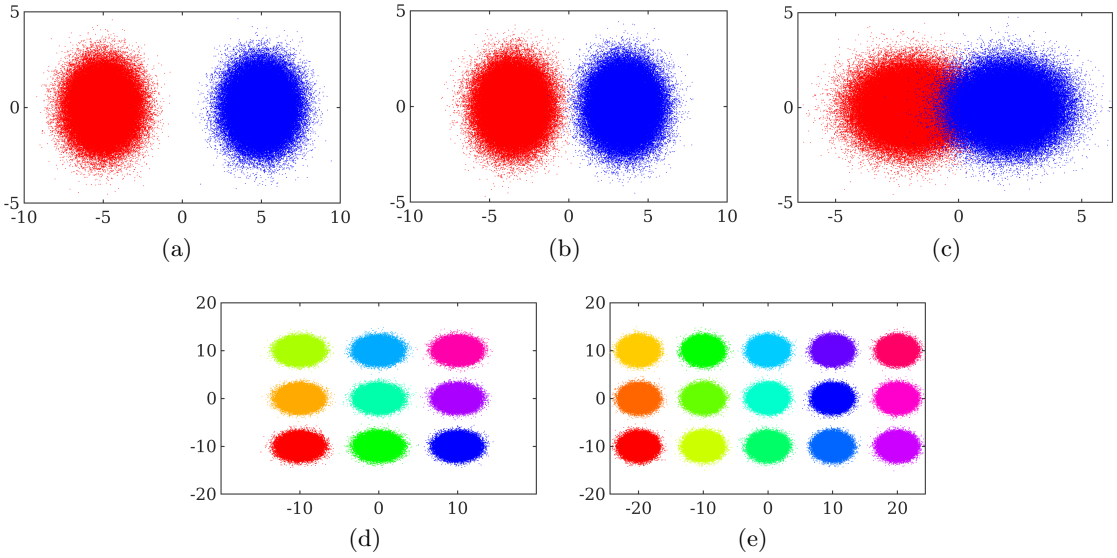


Figure 3.4: Two-dimensional artificial datasets. (a) AD-2-2-NO: 2-dimensional 2 class dataset with no overlap. (b) AD-2-2-SO: 2-dimensional 2 class dataset with small overlap. (c) AD-2-2-LO: 2-dimensional 2 class dataset with large overlap. (d) AD-9-2: 2-dimensional 9 class with no overlap. (e) AD-15-2: 2-dimensional 15 class dataset with no overlap.

value of $u_j(\mathbf{s})$ for the j-th class, can be defined as follows:

$$u_j(s) = \frac{p_j(\mathbf{s})}{\sum_{i=1}^{c} p_i(\mathbf{s})},$$

where the distribution of the j-th class is p_j (here multivariate Gaussian), while $p_j(\mathbf{s})$ is the probability of $\mathbf{s}$ being a member of the that class.

Real world datasets: We collect 12 real-world classification datasets of diverse properties from the KEEL (Triguero et al., 2011), and University of California, Irvine (Dua and

Graff, 2017) machine learning data repositories. In Section B.2 in Appendix B we provide a brief description of the various important attributes (namely the number of points, the number of dimensions, and the number of classes) of the real-world datasets used in the simulation study.

3.4.1.2 Experimental protocol

Our primary goal in this simulation study over artificial datasets is to validate Theorem 3.1 and 3.2. Therefore, for each of the artificial datasets, we gradually increase the size of the training set by respectively combining 10,000, 50,000, 100,000, and 150,000 randomly selected data instances from each of the c classes. Subsequently, we construct a test set corresponding to each of these previously assembled training sets, by picking 100 data instances from the remaining data points in each of the c classes. This entire process is repeated for 10 times (i.e. generating 40 training-test pairs from a single dataset) such that the effect of randomization on the results can be mitigated. Moreover, to evaluate the impact of the choice of k for a given N, we vary its value among 1, $\lceil N^{0.1} \rceil$, $\lceil N^{0.25} \rceil$, and $\lceil N^{0.5} \rceil$ for each of the training-test pairs.

Similarly, in case of the real datasets we perform a 10-fold cross-validation, and vary the value of k as 1, $\lceil N^{0.1} \rceil$, $\lceil N^{0.25} \rceil$, and $\lceil N^{0.5} \rceil$ for each of the training-test pairs. However, as we have pointed out in Section 3.1 the initial fuzzy memberships for the data points in a real dataset can be found by using various techniques. Therefore, in our experiments we choose the popular Keller's Heuristic (KH) (Keller et al., 1985) with the value of k' equals to k. Moreover, in an attempt to validate the efficacy of KH we compare its performance with the Kernel Density Estimation (KDE) using the Gaussian kernel while setting the bandwidth according to the rule of thumb as suggested by Silverman (1988). We use Wilcoxon Signed Rank (WSR) test (García et al., 2013) to evaluate if KH can perform significantly (with a significance level of 5%) different than KDE as an initialization technique, on average over all the 12 real-world datasets.

Among the available indices which are capable to quantize the performance of a classifier, one of the simplest yet effective indexes is the Accuracy measure. The Accuracy index given a c class confusion matrix Q_c is defined as $\beta = \sum_{i=1}^{c} q_{ii}/n$, where $\sum_{i=1}^{c} q_{ii}$ calculates the number of test points correctly classified by a classifier, and n is the size of the test set. Evidently,

$1 \geq \beta \geq 0$, and the higher value of Accuracy indicates a better classification. However, to calculate accuracy for FkNN, one needs to retrieve a class label from the estimated fuzzy memberships. This can be done by assigning a test point to that class which achieves the maximum membership among the c possible classes. Moreover, one is not required to consider the individual class-specific bias and MSE, as according to Theorem 3.2, only observing μ_e, and σ_e should suffice, especially when our experiments involved datasets with a varying number of classes. Therefore, we report all our results in terms of β, μ_e, and σ_e, which also enables us to investigate the importance of the latter two measures in case of fuzzy classifiers.

3.4.1.3 Results on artificial datasets

We plot the average μ_e respectively over the 2-dimensional, 5-dimensional, and 10-dimensional artificial datasets, with varying values of N, and corresponding choices of k in Figure 3.5a. The similar plots for σ_e and β, are respectively illustrated in Figure 3.5b and Figure 3.5c. Following the legends described in Figure 3.5d, we can observe that irrespective of data dimension, and choices of k, the value of μ_e, σ_e, and β, generally (the small fluctuations are mostly due to the effect of randomization) improve with increasing N. Furthermore, the best value of an index for all the cases is usually achieved by $k = \lceil N^{0.25} \rceil$, which is a moderate data-dependent choice. Interestingly, the best μ_e, σ_e, and β on average are achieved over the 2-dimensional datasets. Moreover, the deterioration in the case of μ_e and σ_e with increasing data dimensions are more apparent than that in β. This phenomenon may be explained by the presence of the uncontrolled overlap in the higher dimensional datasets.

To investigate further we plot the variation of μ_e, σ_e, and β averaged over the choices of k, with increasing value of N, for the three datasets, namely AD-2-2-NO, AD-2-2-SO, and AD-2-2-LO, respectively in Figure 3.6a, Figure 3.6b, and Figure 3.6c. As described in Table 3.1 and in Figure 3.4a, Figure 3.4b, and Figure 3.4c, the three chosen datasets have an increasing amount of overlap between the two classes, thus aiding us to assess the impact of overlapping classes on the performance of FkNN. From Figure 3.6a, Figure 3.6b, and Figure 3.6c, it is evident that the values of μ_e, σ_e, and β degrade with increasing overlap. This can be considered natural as given a test point FkNN will have neighboring training points from multiple classes in the overlapped region. All of these training points will have somewhat similar class memberships for the overlapped classes, thus using their information, FkNN will

strive to achieve an accurate membership estimate for the test points. However, the accuracy may still persist as the class corresponding to the maximum estimated membership is likely to be conserved in most of the regular situations. Moreover, with the increasing number of training samples, the effect of overlap may be reduced further. The detailed results on the artificial datasets can be found in Table B.6, B.4, and B.5 of Appendix B.

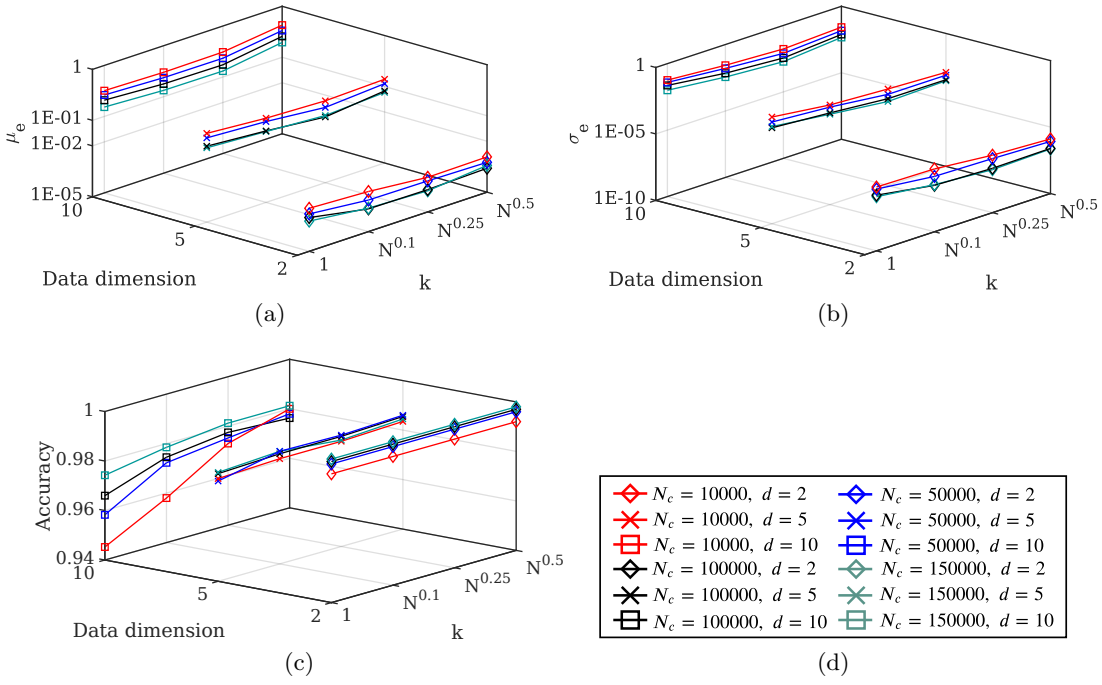


Figure 3.5: Summary of simulation on artificial datasets. (a) Comparison of μ_e over the artificial datasets. (b) Comparison of σ_e over the artificial datasets. (c) Comparison of Accuracy over the artificial datasets. (d) Legends. Note that in all cases a ceiling value is taken to ensure that k is an integer.

3.4.1.4 Results on real-world datasets

To compare the correctness of initial memberships calculated by KH and KDE, we plot the μ_e, σ_e, and β (averaged over the 12 real datasets) with varying choice of k, respectively in Figure 3.7a, Figure 3.7b, and Figure 3.7c. It is evident from Figure 3.7c that the initial membership estimated by KH testifies for a better classification than by KDE, or in other words, the former can better approximate the local data distribution which is helpful for a better performance of FkNN. This is further validated by the lesser average μ_e and σ_e achieved by FkNN using the memberships initialized by KH. Furthermore, the result of the

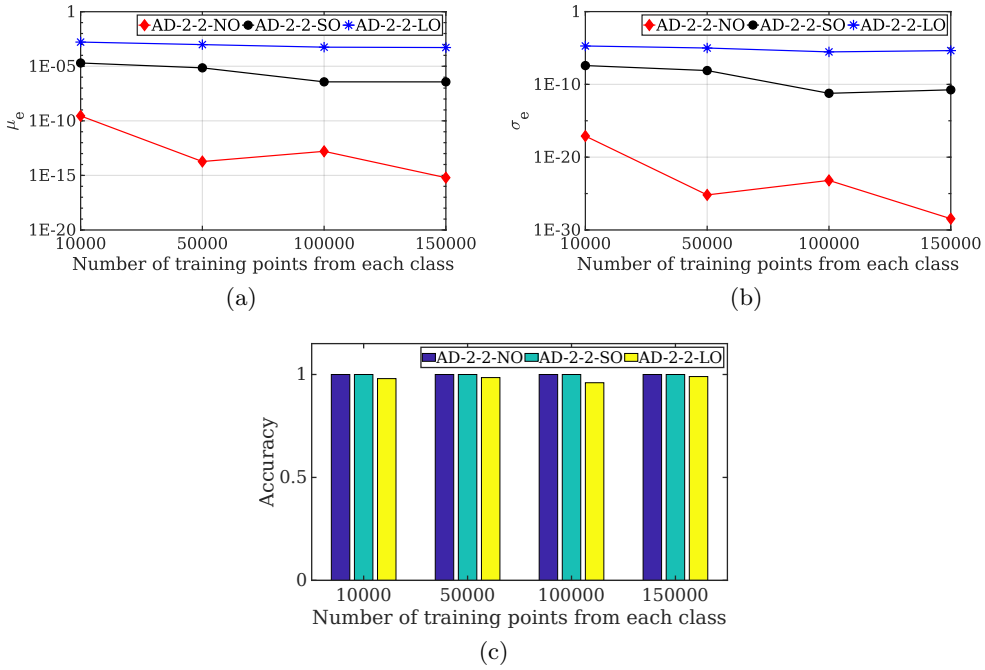


Figure 3.6: The effect of overlapping classes on bias, MSE, and Accuracy. (a) μ_e on the overlapped datasets. (b) σ_e on the overlapped datasets. (c) Accuracy on the overlapped datasets.

Wilcoxon signed rank test, as detailed in Table 3.2, highlights the statistically significant difference of average performance of FkNN in all the cases when KDE is used instead of KH for initialization. Further we observe that on average $k = \lceil N^{0.1} \rceil$ achieves better performance in terms of μ_e and β while $k = \lceil N^{0.25} \rceil$ gives a close competition especially on σ_e. This may be considered as an evidence in support of Assumption 3.1 given the fact that the chosen datasets are of diverse scale (N ranges from 150 to as large as 20,000) suggesting the better performing choice of k to be distinct as well. Thus, compared to the smallest possible value of 1 or the largest conventional choice of $\lceil N^{0.5} \rceil$ the moderate values of the parameter k such as $\lceil N^{0.1} \rceil$ and $\lceil N^{0.25} \rceil$ achieve a better performance on average. The results on the 12 real-world datasets are detailed in Table B.7 of Appendix B.

3.4.2 Comparative study to evaluate the efficiency of LACW

As Section 3.4.1 we start by describing the real-world datasets used in this study. Subsequently, we detail the experimental protocol, describe the competing techniques, and finally provide the results in an attempt to validate the efficacy of LACW in aiding WFkNN in the

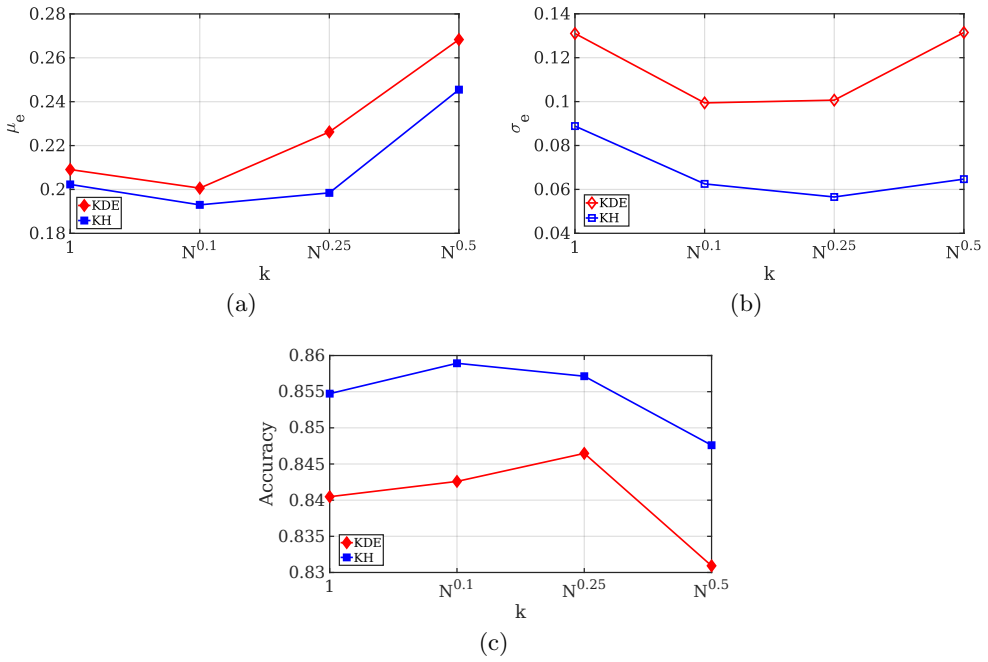


Figure 3.7: Comparison of the performance of KDE and KH in terms of μ_e, σ_e, and β over the real world datasets. (a) Comparison in terms of μ_e averaged over different k. (b) Comparison in terms of σ_e averaged over different k. (c) Comparison in terms of β averaged over different k. Note that in all cases a ceiling value is taken to ensure that k is an integer.

Table 3.2: Summary of Wilcoxon Signed Rank Test for the comparison of performance of KDE and KH over the real world datasets.

	$k=1$			$k=\lceil N^{0.1} \rceil$			$k=\lceil N^{0.25} \rceil$			$k=\lceil N^{0.5} \rceil$		
	μ_e	σ_e	β	μ_e	σ_e	β	μ_e	σ_e	β	μ_e	σ_e	β
KDE	$\mathcal{H}_1$	$\mathcal{H}_1$	$\mathcal{H}_1$	$\mathcal{H}_1$	$\mathcal{H}_1$	$\mathcal{H}_1$	$\mathcal{H}_1$	$\mathcal{H}_1$	$\mathcal{H}_1$	$\mathcal{H}_1$	$\mathcal{H}_1$	$\mathcal{H}_1$
KH	CM	CM	CM	CM	CM	CM	CM	CM	CM	CM	CM	CM

$\mathcal{H}_1$: The difference of performance on the 12 real world datasets are statistically significant as indicated by WSR. $\mathcal{H}_0$: The difference of performance on the 37 real world imbalanced datasets are statistically comparable as indicated by WSR.
CM: Control method.

presence of class imbalance.

3.4.2.1 Real world class imbalanced benchmark datasets used in this comparative study

We select 37 real-world class imbalanced benchmark datasets from six different machine learning data repositories and collections namely KEEL (Triguero et al., 2011), University of California, Irvine (Dua and Graff, 2017), IDA benchmark (Rätsch, 2001), Akbani et al. (2004),

Agnostic Learning vs. Prior Knowledge Challenge (Guyon, 2006), and LibSVM (Chang and Lin, 2011). The datasets are chosen to ensure diversity in terms of scale, dimensionality, number of classes, and IR. A detail of these datasets highlighting their key properties can be found in Table B.3 of Appendix B.

3.4.2.2 Experimental protocol

All our experiments are performed using a 10-fold stratified cross-validation while the value of k is varied between 1, $\lceil N^{0.1} \rceil$, $\lceil N^{0.25} \rceil$, and $\lceil N^{0.5} \rceil$. In this experiment, we are focused on validating the average performance of LACW coupled with WFkNN (hereafter called WFkNN+LACW) in comparison to other FkNN variants. Therefore, in all cases, we report the mean result overall folds and all choices of k. However, classifiers tailored for handling class imbalance may perform slightly poorer on the majority classes at the expense of a significant improvement over the minority ones due to overcompensation. Thus, μ_e and σ_e may not be able to reflect the true performance of a classifier in presence of class imbalance as the significantly lower bias and MSE achieved over less number of minority points may be overwhelmed by the minutely increased bias and MSE over a large number of majority instances. This leads us to use the bias and MSE (respectively denoted by μ_+ and σ_+) only over the minority class instead of μ_e and σ_e. Further, we recall from Chapter 2 that Accuracy cannot be considered as an ideal performance measure in the presence of class imbalance (Sokolova et al., 2006; Japkowicz, 2006). Hence, along with μ_+ and σ_+ we also evaluate the classifiers in terms of ACSA and GMean.

To evaluate the efficacy of WFkNN+LACW we compare its performance with five contending FkNN variants capable of handling class imbalance. First, the regular FkNN classifier as a benchmark. Second, the WFkNN coupled with GCW. Third, regular FkNN trained on a training set artificially balanced by SMOTE (Chawla et al., 2002). The parameter determining the number of neighbors in SMOTE is set to 5 following the original article. Fourth, regular FkNN trained on a training set artificially (almost) balanced by RUS (Japkowicz, 2000). To prevent serious loss of information we restrict RUS to retain at least 20% of data instances from any majority class even if an attempt of balancing requires a lesser number of data points to be sampled. Fifth, the WFkNN coupled with the class weighting approach proposed by Tan (2005) (hereafter called Neighbor Weighted FkNN or NWFkNN). The data-

dependent exponent parameter in NWFkNN is set to 4 following recommendations made in the corresponding research article. In the case of WFkNN+LACW we use the KH initial membership estimation where k' is set to k.

To validate if the performance of WFkNN+LACW is statistically different from its contenders we undertake a WSR test (with a significance level of 5%) over all datasets. Further, to better highlight the efficacy of WFkNN+LACW we perform a Wilcoxon Rank-Sum (WRS) test[1] (Hollander et al., 2013) with 5% level of significance on each of the datasets. We express the outcome of WRS test in the form of a Win-Tie-Loss count for each of the contenders. A Win (W) on a dataset takes place if a contender achieves a significantly worse performance from the control method WFkNN+LACW according to WRS. A Tie (T) is considered when the WRS finds the performance of the contender is statistically comparable with WFkNN+LACW. Finally, A Loss (L) is defined as the case when WFkNN+LACW performs worse than a contender while WRS indicates that the difference in performance is statistically significant.

Table 3.3: Performance comparison of WFkNN+LACW with other FkNN variants on real world class imbalanced benchmark datasets

	FkNN	WFkNN +GCW	FkNN +SMOTE	FkNN +RUS	NWFkNN	WFkNN +LACW
Average ACSA	0.70	0.77	0.60	0.77	0.73	**0.79**
Average GMean	0.55	0.73	0.33	0.70	0.63	**0.75**
Average μ_+	0.28	0.19	0.46	0.20	0.24	**0.18**
Average σ_+	0.12	0.07	0.24	0.08	0.09	**0.06**
W-T-L for ACSA	32-5-0	22-15-0	30-7-0	20-15-2	27-10-0	CM
W-T-L for GMean	31-6-0	22-15-0	30-6-1	21-14-2	28-9-0	CM
W-T-L for μ_+	36-1-0	19-18-0	36-1-0	17-14-6	31-6-0	CM
W-T-L for σ_+	34-3-0	20-17-0	37-0-0	20-12-5	25-12-0	CM
WSR for ACSA	$\mathcal{H}_1$	$\mathcal{H}_1$	$\mathcal{H}_1$	$\mathcal{H}_1$	$\mathcal{H}_1$	CM
WSR for GMean	$\mathcal{H}_1$	$\mathcal{H}_1$	$\mathcal{H}_1$	$\mathcal{H}_1$	$\mathcal{H}_1$	CM
WSR for μ_+	$\mathcal{H}_1$	$\mathcal{H}_1$	$\mathcal{H}_1$	$\mathcal{H}_1$	$\mathcal{H}_1$	CM
WSR for σ_+	$\mathcal{H}_1$	$\mathcal{H}_1$	$\mathcal{H}_1$	$\mathcal{H}_1$	$\mathcal{H}_1$	CM

The best result is boldfaced.
$\mathcal{H}_1$: The difference of performance on the 37 real world imbalanced datasets are statistically significant as indicated by WSR.
$\mathcal{H}_0$: The difference of performance on the 37 real world imbalanced datasets are statistically comparable as indicated by WSR.
CM: Control method.

[1] Also known as Mann-Whitney U test.

3.4.2.3 Results on real-world class imbalanced benchmark datasets

We summarize the results on 37 real-world class imbalanced benchmark datasets in the following Table 3.3. A closer inspection of Table 3.3 reveals that the regular FkNN fails to achieve a commendable performance suggesting the need of specially tailored FkNN variants capable of countering the adverse effects of class imbalance. Moreover, WFkNN+LACW outperforms others in all four of the performance indices while the WSR test verifies its performance to be significantly different from all its contenders over the 37 datasets. Therefore, in view of the empirical findings, it is safe to say that WFkNN+LACW is indeed capable of alleviating the susceptibility of WFkNN in the presence of class imbalance. Further, the improvement achieved over WFkNN+GCW highlights the importance of incorporating local information through the point specific adaptive class weighting strategy in WFkNN+LACW. Moreover, the performance of WFkNN+LACW is only followed by FkNN+SMOTE which manages to achieve 6 and 5 wins when respectively compared in terms of μ_+ and σ_+. This is an expected behavior as SMOTE increases the number of training points which according to Theorem 3.1 and 3.2 should facilitate a better convergence of FkNN. However, as noted earlier in Section 3.4.1.3 an improvement in bias and MSE may not always alter the final labeling of a test point. Therefore, the competitive performance of FkNN+SMOTE becomes less apparent when compared in terms of ACSA or GMean. We detail the results on the real world class imbalanced datasets in Table B.8, B.9, B.10, and B.11 on Appendix B.

3.5 Discussion

In this chapter, we presented an analytical view of the performance of FkNN classifier. Contrary to the previous attempts our analysis followed a simpler approach that utilizes unbounded loss functions, relies only on three elementary assumptions, and remains independent of the choice of k, m, and the number of classes. Moreover, we showed that proving the convergence of bias and MSE of FkNN estimator can indicate a more interpretable performance assurance than comparing the risk of FkNN with that of the Bayes classifier. In particular, we proved that the bias and MSE of the FkNN estimator for each of the classes are bounded, and go to 0 as $n \to \infty$. Additionally, we presented a detailed simulation study verifying our theoretical claims and evaluating the performance of FkNN in practical applica-

tions. The simulation also highlights the need for tuning k for achieving a better performance and validates the resilience of FkNN to the presence of overlapping classes. Further, we described LACW, a point specific locally adaptive class weighting strategy that when coupled with weighted FkNN can help in improving the performance of the classifier in the presence of class imbalance. Finally, we detailed a comparison study with FkNN variants tailored for handling class imbalance to establish the efficacy of the LACW strategy and validated the usefulness of incorporating the local information of a test point through adaptive class-specific weights.

Chapter 4

Parameter Independent Fuzzy k-Nearest Neighbor Classifier for Balanced and Imbalanced Data Classification

Summary

Unlike the canonical kNN which treats the neighbors equally, the FkNN classifier imposes a weight on each of the k nearest neighbors based on their distances from the query point, by using a fuzzy membership function. FkNN though improves the performance of kNN, requires optimizing additional data-dependent parameters other than k. Furthermore, FkNN does not consider the effect of those representative features of a data point which may be noisy, redundant, and may not contain useful information to distinctly identify a specific class. We attempt to address both of these issues in the current chapter by proposing a Parameter Independent Fuzzy class-specific Feature Weighted k-Nearest Neighbor (PIFWkNN) classifier. PIFWkNN formulates the issues of choosing a suitable value of k and a set of class-dependent optimum weights for the features as a single-objective continuous non-convex optimization problem. We solve this problem by using a very competitive variant of Differential Evolution (DE), called Success-History based Adaptive DE (SHADE). We perform extensive experiments to demonstrate the improved accuracy of PIFWkNN compared to the other state-of-the-art classifiers. Furthermore, FkNN classifiers are known to be susceptible to the presence of a class imbalance in the training set. In this chapter, we attempt to address this issue as well by aiding FkNN to combat the detrimental effects of class imbalance through a set of global class-

specific weights. The formulation of PIFWkNN allows a direct optimization of such a set of global weights which are likely to be effective in offering the classifier additional immunity against class imbalance. We call our approach PIFW^{2}kNN and validate its efficacy through a comparative study of performance.

4.1 Introduction

4.1.1 Overview

As mentioned earlier in Chapter 3 the kNN classifier has always been a popular and widely applied choice (Angiulli and Fassetti, 2013; Liu and Liu, 2016) for its good performance as well as conceptual and computational simplicity. Furthermore, kNN, being a non-parametric classifier (Duda et al., 2000), does not depend on any prior assumptions made about the data distribution. However, kNN is reliant on three important factors for performing a successful classification; namely the size of the neighborhood determined by the value of k, the distance function used to identify the neighbors, and the importance of a neighbor, class, or feature.

Over the years a vast amount of research concentrated on one or more of these three directions to improve the accuracy of kNN. Cover and Hart (1967) first showed that the probability of inaccurate classification by kNN can be at most twice the risk of the Bayes classifier when $k = 1$ and the number of training samples $N \to \infty$. However, in reality, one always has access to a finite number of training instances. Thus, choices of k other than 1 may be more useful (Hall et al., 2008). Hence for practical implementation, several techniques like cross-validation and probabilistic modeling (Ghosh, 2006) are suggested to find a single global optimum choice of k. It is also a common practice to use a conventional constant choice of k limited among 1, 3, 5, 7, 9 and $\lceil\sqrt{N}\rceil$ (Bhattacharya et al., 2012; Domeniconi et al., 2002).

Optimizing only the number of neighbors may not enough as the definition of a set of neighboring training points is reliant on the choice of the distance metric. The issue of finding a suitable distance measure for the kNN classifier influenced many researchers to design adaptive Euclidean, City-block (Wang et al., 2007), and Mahalanobis (Weinberger et al., 2005) distances. Previous researches also pointed out that the kNN classifier may be benefited by incorporating one or more of the class, feature, and neighbor specific weighting

schemes. An intuitive idea suggests that a distant neighbor should play a less significant role than a close one in deciding for a test instance, which is theoretically validated recently by (Samworth, 2012). Class-specific weighting is useful for aiding kNN with additional knowledge about the class properties, especially when all the classes are not equally represented in the training set (Tan, 2005). Moreover, to remove the effect of noisy and redundant features, attribute specific weighting has been shown to be beneficial (Mateos-García et al., 2012).

As discussed in Section 3 the traditional kNN provides a hard decision for a test point which may be restricted due to the limited access to the information about the class distributions learned from a finite set of training samples. Contrary to this, the Fuzzy-kNN (FkNN) classifier (Keller et al., 1985; Derrac et al., 2014) not only provides a soft decision for a query point $\mathbf{y}$ (in the form of probabilistic membership for $\mathbf{y}$ to belong in all of the possible classes) but also assigns a weight to a neighbor inversely proportionate to its distance from $\mathbf{y}$. The flexibility offered by FkNN is further extended by utilizing the interval type-2 fuzzy logic (Mendal and John, 2002). FkNN and its variants though improve the accuracy of kNN, in return increase the complexity by introducing some new parameters (for example m, which controls the weight decay of a neighbor in proportion to its distance from the test instance or k' which is used for calculating an initial membership of the training points) alongside k, all of which are needed to be properly optimized.

4.1.2 Background

4.1.2.1 Class specific feature selection

A data instance is traditionally represented through a set of handcrafted features (Fisher, 1936). However, in most cases, the set of features is not designed while keeping a particular application in mind. Thus, if a data instance is expressed in the form of a d-dimensional vector then all of the d features may not effectively contribute to a certain application. Further, a feature can be noisy by design and may provide redundant information (Hanchuan Peng et al., 2005). Thus, a primary concern of the machine learning community is to select a subset of useful features that will offer relevant information for efficiently performing a given task such as classification (Chandrashekar and Sahin, 2014). One may further extend the notion of feature selection in case of classification by arguing that not all selected features

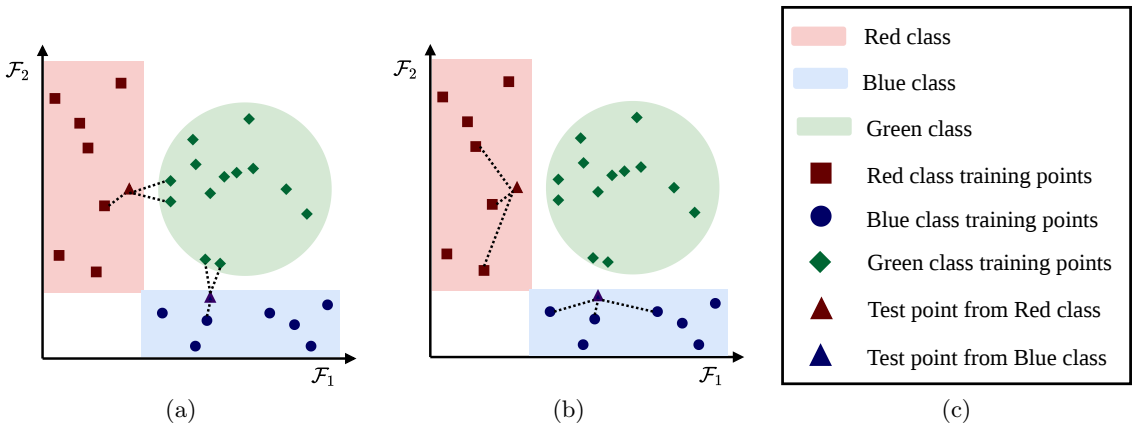
play an equal role in distinguishing all the classes. Thus, contrary to the selection of a global subset of features it may be more beneficial to look for class-specific subsets that are likely to improve the classification performance on the class under concern (Pineda-Bautista et al., 2011; Chen and Gu, 2015). However, a crisp selection either retains or discards a feature in its entirety which may hinder the efficient exploitation of information. Instead one may choose to assign a class-specific weight to each feature according to their relevance in the proper identification of a particular class.

4.1.2.2 Evolutionary optimization for parameter tuning

Employing Evolutionary Optimization (EvO) techniques for optimizing fuzzy classifiers is a well-explored direction of research (Munoz-Salinas et al., 2008; Trawiński et al., 2013, 2014; Fernández et al., 2015b). This is because in such systems it is often difficult to express the objective function in a form which will ensure the necessary conditions set by a mathematical optimizer. Following this route EvO techniques were also employed for tuning k (and in cases m and k') associated with kNN, FkNN, and their variants (Paredes and Vidal, 2006; Hu and Xie, 2005). The reason behind favoring an EvO technique over a mathematical optimization algorithm is the inadequate knowledge about the properties of the fitness function used to model the optimization problem. Moreover, depending on the encoding scheme (influenced by the domain of the fitness function), one has the freedom of choosing a particular algorithm from the vast and diverse collection of EvO methods such as Genetic Algorithm (GA) (Goldberg, 1989), Differential Evolution (DE) (Storn and Price, 1997), etc. The practice of using EvO algorithms for optimizing the performance of kNN was later modified to incorporate the calculation of a set of optimal global weights for the features (Raymer et al., 2000) and further extended to consider additional class information. However, EvO algorithms introduce a new set of performance affecting parameters which are to be adjusted. Furthermore, the common practice does not simultaneously optimize the feature weights and the value of k.

4.1.3 Motivation

We are motivated by the fact that the performance of FkNN can be improved by using the optimal choices of parameters k, k', and m. However, among the three parameters optimizing only k will suffice as according to the study by Keller et al. (1985); the value of



(a) (b) (c)

Figure 4.1: The effect of class specific feature weighting on a kNN classifier. (a) Classification without feature weighting. (b) Classification when class-specific feature weighting is used. (c) Legends.

m has little effect over the performance when restricted to its conventional choice, while k and k' can have similar value, thus requiring no explicit tuning. Furthermore, introducing important additional information through features and class weights may be helpful for better classification. This can be achieved by either a set of globally optimized weights for the attributes in addition to class weighting (AlSukker et al., 2010) or a class-specific optimal set of weights for the features (Paredes and Vidal, 2006). Among the two techniques, we prefer the use of class-specific feature weights for its simplicity. The properties of the class distributions are directly reflected through the sampled values of the features. Therefore, it is evident that each class can be better distinguished from the rest using a unique set of informative features. Hence, a weighting scheme should consider the importance of a feature not from the global perspective of the entire dataset but from that of the individual classes; which can be better realized by using a set of class-specific feature weights. The usefulness of such a weighting technique can be explained with the help of the following illustrative Example 4.1.

Example 4.1. *Let us a take a three-class classification problem, where the classes are named as Red, Blue, and Green as illustrated in Figure 4.1a (see Figure 4.1c for legends). The dataset is two-dimensional i.e. a data instance is represented through two features namely* $\mathcal{F}_1$ *and* $\mathcal{F}_2$*. We take two test instances, one from the Red class and the other from the Blue class. If we do not use any class specific feature weighting then from Figure 4.1a we can see that*

both of the test points will be misclassified as members of the Green class by a regular kNN classifier with $k = 3$ if Euclidean distance is used. However, a careful observation will reveal that $\mathcal{F}_2$ has little contribution for distinguishing a member of the Red class while $\mathcal{F}_1$ performs the same for the Blue class by remaining almost invariant. In case of the Green class, both of $\mathcal{F}_1$ and $\mathcal{F}_2$ are evenly spread between a similar range indicating an equal importance in distinguishing one of its member. Hence, using weighted Euclidean distance (see expression (4.2)) with a feature weight vectors $\omega^{\mathcal{F}}_{Red}$, $\omega^{\mathcal{F}}_{Blue}$, and $\omega^{\mathcal{F}}_{Green}$ respectively for the Red, Blue, and Green class may prove to be beneficial. We select such a set of weights as in matrix $\Omega^{\mathcal{F}}$ in expression (4.1).

$$\Omega^{\mathcal{F}} = \begin{bmatrix} \omega^{\mathcal{F}}_{Red} \\ \omega^{\mathcal{F}}_{Blue} \\ \omega^{\mathcal{F}}_{Green} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{bmatrix}. \tag{4.1}$$

We can verify that the use of such a set of weights can indeed correctly classify both of the test points in their respective classes as seen in Figure 4.1b.

Thus, in an attempt to improve the accuracy of an FkNN classifier, one may simultaneously optimize the parameters of FkNN as well as a set of class-specific feature weights using a suitable combination of a fitness function, encoding scheme, and an EvO algorithm. The class-specific feature weighting should also offer immunity against class imbalance to some extent as it attempts to adapt the distance measure and consequently the neighbors to facilitate a better classification. However, only class-specific feature weighting may not be enough if the number of training points from the minority class(es) is severely limited. The situation will only worsen with the increasing sparsity of the minority class(es). Thus, to offer further resilience against class imbalance one may attempt to replace FkNN with WFkNN and introduce an additional set of class-specific weights. Optimization of such class-specific weights alongside the parameters and feature weights of WFkNN by EvO techniques can be achieved by a simple modification of the fitness function and the encoding scheme of the candidate solutions.

4.1.4 Contribution of Chapter 4

In this chapter, we propose a new Parameter Independent Fuzzy class-specific feature Weighted kNN (PIFWkNN) classifier. The major contribution of PIFWkNN is incorporating a set of optimal class-specific weights for the features with the traditional FkNN. The PIFWkNN classifier further optimizes the choice of the global value of k, such that the performance of FkNN can be maximized. However, in contrary to the previous works, PIFWkNN does not express the optimization of the weights and the value of k as two separate problems. The PIFWkNN classifier uses a novel real-valued encoding scheme such that the two optimization problems can be conjugated into a single one and can be simultaneously solved by an evolutionary algorithm like DE. However, the need of preserving the parameter independent nature of the proposed classifier cannot be satisfied by the use of regular DE, as it introduces two new performance controlling parameters (namely the scale factor of the mutation and the crossover rate). Therefore, in this study, we use a state-of-the-art variant of DE, called Success-History based parameter Adaptive Differential Evolution (SHADE) (Tanabe and Fukunaga, 2013) which can optimally adapt the different associated parameters of DE on-the-fly alongside ensuring better performance than canonical DE.

To establish the effectiveness of the alterations made through the proposed technique, we perform an extensive experiment using 20 real-world datasets of varying properties. We compare the classification accuracy of PIFWkNN with that of kNN, FkNN, Extended Nearest Neighbor (ENN) (Tang and He, 2015), kLDEDW (Mateos-García et al., 2012), DE4 (Al-Sukker et al., 2010), CW (Paredes and Vidal, 2006), GA-Fuzzy kNN (Hu and Xie, 2005) and, EF-kNN IVFS (Derrac et al., 2016). Two non-parametric statistical testing methods namely Wilcoxon Rank Sum[1] (WRS) (Gibbson and Chakraborti, 2011) test and Wilcoxon Signed-Rank (WSR) (Derrac et al., 2011) test are used to check if the improvement of PIFWkNN is significantly better than the contending algorithms.

We further extend the PIFWkNN classifier for improving its performance of class imbalance data and propose a class weighted variant called as PIFW^2kNN. The proposed PIFW^2kNN replaces the FkNN in PIFWkNN with a WFkNN such that additional class-specific weights can be Incorporated. Instead of finding the class weights through some

[1]Alternatively called Mann-Whitney U test.

heuristic techniques PIFW^2kNN optimizes them simultaneously with the associated parameters and feature weights. We validate the efficacy of the PIFW^2kNN on 20 class imbalanced real-world benchmark datasets in comparison to the state-of-the-art variants of kNN and FkNN tailored to efficiently combat the effects of class imbalance.

The rest of the chapter is organized as follows. In Section 4.2, we discuss the PIFWkNN and PIFW^2kNN algorithms in detail. In Section 4.3, we present the results and evaluate the improvements achieved by the proposed classifier, and follow it with a discussion in Section 4.4.

4.2 Proposed method

In Chapter 3 we have already introduced the kNN, FkNN, and WFkNN classifiers. Thus, in this chapter, we can directly start with the discussion on the role of a distance function in such classifiers.

4.2.1 Feature weighted Euclidean distance

Both of the kNN and the FkNN algorithms can be easily modified by using a suitable distance measure Δ which is necessary for finding the neighbors and calculating the class memberships. Instead of the traditional Euclidean distance here we propose to use a class-based feature weighted Euclidean distance measure $\Delta_{\Omega^{\mathcal{F}}}$. Given a training point $\mathbf{x} \in X$ and a test point $\mathbf{y} \in X'$ weighted Euclidean distance can be defined as:

$$\Delta_{\Omega^{\mathcal{F}}}(\mathbf{x}, \mathbf{y}) = \left(\sum_{i=1}^{d} \omega^{\mathcal{F}}_{h(\mathbf{x})i} (x^{(i)} - y^{(i)})^2 \right)^{\frac{1}{2}}, \tag{4.2}$$

where, $\omega^{\mathcal{F}}_{ij}$ is the weight for the i-th dimension when the training point $\mathbf{x}$ belongs to the j-th class for all $i \in \{1, 2, \cdots, d\}$ and $j \in C$. Therefore, $\Omega^{\mathcal{F}} = [\omega^{\mathcal{F}}_{ij}]_{c \times d}$.

4.2.2 Optimization problems, DE, and SHADE

Let us take a function $\mathcal{J} : \Theta \to \mathbb{R}$, where Θ is a set defined as follows:

$$\Theta = \{\theta | \theta \in \mathbb{R}^d, A^{(1)}_3 \leq \theta^{(1)} \leq A^{(1)}_4, \cdots, A^{(d)}_3 \leq \theta^{(d)} \leq A^{(d)}_4\}. \tag{4.3}$$

Here, $A_3 = \{A_3^{(1)}, A_3^{(2)}, \cdots, A_3^{(d)}\}$ and $A_4 = [A_4^{(1)}, A_4^{(2)}, \cdots, A_4^{(d)}\}$. An optimization problem, whose solution is $\hat{\theta}$ can be expressed in the form of minimization problem without loss of generality as follows:

$$\hat{\theta} = \arg\min_{\theta \in \Theta} \mathcal{J}(\theta). \tag{4.4}$$

Here $\mathcal{J}$ is known as the objective function, which represents the optimization problem and is needed to be minimized. DE attempts to find $\hat{\theta}$ by starting with a random population $\Theta_{N_p} \subseteq \Theta$ containing N_p number of candidate solutions and over iterations evolving/improving them until a termination condition is met. The evolution is performed by sequentially applying three operations namely mutation (using a difference vector of two candidate solutions), crossover (exponential or binomial), and selection to the population, a detail of which can be found in Das et al. (2016). There exists a wide variety of mutation and crossover operations, designed for tackling different types of optimization problems.

The canonical DE technique though simple in nature greatly depends on the choices of the scale factor $\mathbb{F}$ and the crossover rate C_r to attain good performance. The value of N_p is conventionally kept fixed to a constant, while the maximum number of Fitness Evaluations (FEs) is dependent on the problem dimension (Liang et al., 2013; Awad et al., 2016). Thus, the use of DE though relieves the user from tuning the value of k and finding the optimal feature weights; still requires adjusting $\mathbb{F}$ and Cr for effectively improving the accuracy of FkNN. This issue can be solved by using a state-of-the-art variant of DE, called SHADE, which utilizes self-adaptation techniques to intelligently calculate the optimal value of $\mathbb{F}$ and C_r. Moreover, SHADE produces a better result than regular DE using less number of iterations which is confirmed by its performance in the CEC 2013 single-objective real-valued benchmark functions optimization challenge (Liang et al., 2013). The SHADE method is described in Algorithm 1, where `randc`(A_5, A_6) and `randn`(A_7, A_8) sample a real value at random respectively from a Cauchy distribution (with location A_5 and scale A_6), and a normal distribution (with mean A_7 and standard deviation A_8).

Algorithm 1 The SHADE algorithm

Input: The optimization problem $\mathcal{J}$, A_3, A_4, population size N_p, the size of memory pool $\mathbb{H}$.
Output: The solution $\hat{\mathbf{s}}$ of the optimization problem $\mathcal{J}$.

Initialize population $\Theta_{N_P} = \{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \cdots, \boldsymbol{\theta}_{N_p}\}$, a set of random d-dimensional real vectors sampled from an uniform distribution and scaled between A_3 and A_4. Also set $M(C_r) = \{0.5\}^{\mathbb{H}}$, $M(\mathbb{F}) = \{0.5\}^{\mathbb{H}}$, $A_r = \phi$, $b_0 = 1$.
while not converged **do**
  $\mathcal{M}(C_r) = \phi$, $\mathcal{M}(\mathbb{F}) = \phi$.
  for $j \leftarrow 1$ to N_p **do**
    Randomly select an integer ψ between 1 and $\mathbb{H}$.
    Set C_{r_j} =`randn`$(M(C_r)_\psi, 0.1)$, $\mathbb{F}_j$ =`randc`$(M(\mathbb{F})_\psi, 0.1)$.
    Generate the trial solution $\boldsymbol{\theta}'_j$ by current-to-pbest/1/bin method using crossover rate C_{r_j} and scale factor $\mathbb{F}_j$ (Zhang and Sanderson, 2009).
    if $\mathcal{J}(\boldsymbol{\theta}'_j) \leq \mathcal{J}(\boldsymbol{\theta}_j)$ **then**
      $\boldsymbol{\theta}_j = \boldsymbol{\theta}'_j$.
    end if
    if $\mathcal{J}(\boldsymbol{\theta}'_j) < \mathcal{J}(\boldsymbol{\theta}_j)$ **then**
      Update $A_r = A_r \cup \boldsymbol{\theta}_j$, $\mathcal{M}(C_r) = \mathcal{M}(C_r) \cup C_{r_j}$, and $\mathcal{M}(\mathbb{F}) = \mathcal{M}(\mathbb{F}) \cup \mathbb{F}_j$.
    end if
  end for
  Randomly discard solutions from A_r, if $|A_r| > N_p$.
  if $\mathcal{M}(C_r) \neq \phi$ and $\mathcal{M}(\mathbb{F}) \neq \phi$ **then**
    Update $M(C_r)_{b_0}$ and $M(\mathbb{F})_{b_0}$ respectively using $\mathcal{M}(C_r)$ and $\mathcal{M}(\mathbb{F})$ following Tanabe and Fukunaga (2013).
    Update $b_0 = (b_0 \; mod \; \mathbb{H}) + 1$.
  end if
end while

4.2.3 Addressing the compromise between the performance and complexity of SHADE compared to canonical DE

We describe a set of assumptions based on which the computational complexity of DE and SHADE can be derived and compared in the subsequent theorem.

Assumption 4.1. *Both DE and SHADE are optimizing the same problem using similar fitness function and equivalent d-dimensional encoding of the candidate solutions. This enables us to discard the complexity of evaluating the fitness function from our discussion.*

Assumption 4.2. *Let N_p be treated as a constant following the common practice as described in details by Das et al. (2016). This assumption can be further supported by the fact that in majority of cases $Np < A_9 d$ (where A_9 is a small usually $0 \leq A_9 \leq 10$) and almost certainly $N_p \ll d^2$.*

Assumption 4.3. *Both DE and SHADE are allowed to employ an equal number of FEs*

while attempting to optimize the fitness function. Following the conventional standards of evolutionary algorithm evaluation documented by various CEC competitions (Liang et al., 2013; Awad et al., 2016) the number of FEs can be considered to be bounded by $O(d)$. Hence, using Assumption 4.2 the number of generations (τ) can be bounded by $O(d)$ as well.

Under these elementary assumptions, we can now proceed to compare the computational complexities of SHADE and regular DE.

Theorem 4.1. *For a maximum number of generations bounded by $O(d)$, an attempt to optimize a d-dimensional optimization problem can be done by both of DE and SHADE in $O(d^2)$ time.*

Proof. The time complexity of DE following Assumption 4.1 was previously derived as $O(N_p d\tau)$ in the work of Zielinski et al. (2005). This complexity can directly be reduced to $O(d^2)$ if Assumption 4.2 and 4.3 are applied.

At the start of every generation, SHADE spends a constant (follows from Assumption 4.2) amount of time to sort the current N_p solutions based on their fitness such that a small fraction of N_p number of better solutions can be identified. Alongside, SHADE also finds the union of the current population and an external archive, both containing at most N_p number of d-dimensional candidate solutions, which requires $O(d)$ time due to Assumption 4.2. In every generation, each of the N_p solutions (called as target) respectively goes through the evolutionary steps of mutation, crossover, and selection. SHADE uses a current-to-pbest/1 mutation strategy, which can generate a mutated (donor) vector in $O(d)$ time (constant time is spent for randomly sampling the parameters from their respective distribution and finding two random solutions from the archive). The target and corresponding donor then participate in binomial crossover to form a trial vector in another $O(d)$ time. The selection between trial and target vector based on their fitness can be performed in constant time with the help of Assumption 4.1. Moreover, following Assumption 4.2 and the size of $\mathbb{H}$ being a small integer, the time needed for the archiving of successful parameters and maintaining $\mathbb{H}$ can be considered to be constant. Therefore, each generation of SHADE can be performed in a total of $O(N_p d) \approx O(d)$ time. Finally, the total time complexity of SHADE can be bounded by $O(d\tau)$, which following Assumption 4.3 is reduced to $O(d^2)$. □

From Theorem 4.1 we can observe that canonical DE and SHADE are both bounded by the same time complexity of $O(d^2)$; while the later is documented (Tanabe and Fukunaga, 2013) to outperform the former. Furthermore, a recent study by Ghosh et al. (2017) showed that in practice on 30-dimensional benchmark problems (Liang et al., 2013), SHADE takes slightly more execution time than DE/rand/1/bin algorithm, if the values of $\mathbb{F}$ and C_r are kept fixed for DE. However, in the surveys of Das and Suganthan (2011) and Das et al. (2016), the authors highlighted the importance of tuning $\mathbb{F}$ and C_r to optimize the performance of DE by citing several notable works on the topic. Therefore, the effective execution time of a single run of DE will be the average time reported by Ghosh et al. (2017) multiplied by the number of possible combinations of allowed values of $\mathbb{F}$ and C_r; making it far more disadvantageous than a single run of SHADE. In conclusion, SHADE can be considered more beneficial in terms of the performance-complexity trade-off compared to canonical DE, which leads us to use it as the evolutionary optimizer in this chapter.

4.2.4 Objective function and encoding scheme for PIFWkNN

We need to design an objective function, the minimization of which will optimize the performance of PIFWkNN. A choice of such a function can be the leave-one-out misclassification error over the training set. Such an error can be calculated by classifying each $\mathbf{x} \in X$ by PIFWkNN, while $X \setminus \{\mathbf{x}\}$ is used as the training set. As our aim is to optimize the parameters for PIFWkNN, the candidate solutions of the objective function should also incorporate the parameter space of the classifier. Let us denote the domain of the objective function as $\mathbb{T} \in \mathbb{R}^{d'}$. Thus, we formally define the objective function $\mathcal{E} : \mathbb{T} \to \mathbb{R}$ as shown below:

$$\mathcal{E}(\boldsymbol{\eta}) = \left(1 - \frac{1}{N}\sum_{\mathbf{x}\in X} \mathcal{I}(\hat{h}^{X\setminus\{\mathbf{x}\}}(\mathbf{x}) = h(\mathbf{x}))\right). \tag{4.5}$$

where $\boldsymbol{\eta} \in \mathbb{T}$ is a candidate solution of $\mathcal{E}$ from which the optimal choices of the parameters k and $\Omega^{\mathcal{F}}$ can be extracted. Now the only remaining task is to encode $\boldsymbol{\eta}$ in such a way that not only $\Omega^{\mathcal{F}}$ and k can be easily calculated from it but also it can be properly evolved by SHADE. We illustrate such an encoding in Figure 4.2.

One such formulation of $\boldsymbol{\eta}$ can be in the form of a d'-dimensional vector where each row of $\Omega^{\mathcal{F}}$ is horizontally concatenated while an extra dimension is added for k. However, such a

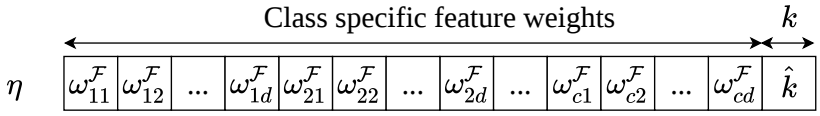


Figure 4.2: Encoding scheme for PIFWkNN. We represent a candidate solution $\boldsymbol{\eta} \in \mathbb{T}$ in the form of a d' dimensional real-valued vector, where $d' = (d \times c) + 1$. Given a d-dimensional c-class dataset the first $(d \times c)$ dimensions of η represent the class specific feature weights. Specifically the feature weights for the j-th class are stored between the $((d \times (j-1)+1)$-th and $(d \times j)$-th dimension of η for all $j \in C$. The last $((d \times c)+1)$-th dimension of $\boldsymbol{\eta}$ is used to optimize the parameter k.

formulation of $\boldsymbol{\eta}$ despite being capable to simultaneously encode $\Omega^{\mathcal{F}}$ and k, cannot be directly implemented for optimization in practice. This is because $\Omega^{\mathcal{F}}$ is a real-valued matrix, each row of which should be in the range between 0 and 1. On the other hand, k is an integer lying in the range between 1 and $\sqrt{N}$. Such an issue can be solved by extracting $\Omega^{\mathcal{F}}$ and k from an evolved solution $\boldsymbol{\eta}$ (instead of directly using it as the parameter vector) after applying some corrections.

4.2.5 Putting it all together: the PIFWkNN algorithm

The proposed PIFW-kNN classifier starts by initializing a random (sampled from a d'-dimensional uniform distribution) population $\mathbb{T}_{N_p}$ containing N_p number of candidate solutions $\boldsymbol{\eta}$ (where, $0 \leq \eta^{(i)} \leq 1, \forall \boldsymbol{\eta} \in \mathbb{T}_{N_p}$, for all $i = 1, 2, \cdots, d'$ and $\mathbb{T}_{N_p} \subset \mathbb{T}$ is the current population). Each of the initial solutions are corrected and after extracting corresponding $\Omega^{\mathcal{F}}$ and k evaluated by calculating $\mathcal{E}$. For $\Omega^{\mathcal{F}}$ the correction and extraction can be done by normalizing the set of weights for each of the classes ($[\eta^{(jd-d+1)}, \eta^{(jd-d+2)} \cdots, \eta^{(jd)}]$ for the j-th class) in the range 0 to 1. While, for k, $\eta^{(d')}$ is first bounded between ϵ and 1, where ϵ is a very small positive real number (for $\eta^{(d')} \leq 0$ set $\eta^{(d')} = \epsilon$ and for $\eta^{(d')} \geq 1$ set $\eta^{(d')} = 1$, otherwise the value of $\eta^{(d')}$ can be retained). The modified $\eta^{(d')}$ is multiplied by the maximum allowable value of k and rounded to the next integer as in expression (4.6).

$$k = \left\lceil \eta^{(d')} \times \sqrt{N} \right\rceil. \tag{4.6}$$

This population is fed to SHADE for evolution. In each iteration of SHADE new solutions are generated by applying evolutionary operations, and from these solutions after necessary

correction a value of $\Omega^{\mathcal{F}}$ and k can be extracted. Using these obtained parameter settings alongside the training set X, the classification error $\mathcal{E}$ of the modified FkNN is calculated and the new solution becomes a part of the population only if its error is less than that of the corresponding parent. These steps are repeated until a termination condition is met. Finally the best solution $\hat{\boldsymbol{\eta}}$ among the evolved population is obtained which can be used to construct the optimal $\Omega^{\mathcal{F}}$ and k. These parameters can then be utilized for classifying a test point. We describe the entire method in Algorithm 2.

Algorithm 2 The PIFWkNN algorithm

Input: The training set X, a test point $\mathbf{y} \in X'$.
Output: The class label $\hat{h}^X(\mathbf{y})$ of $\mathbf{y}$ as predicted by PIFWkNN.

1: Randomly initialize population $\mathbb{T}_{N_p} = [\eta_{tj}]_{N_p \times d'}$ with real values lying between 0 and 1.
2: Evolve $\mathbb{T}_{N_p}$ by SHADE to minimize objective function $\mathcal{E}$ as in (4.5). Apply necessary corrections to each of the newly obtained trial solution and extract $\Omega^{\mathcal{F}}$ and k as described in Section 4.2.5 for evaluating them by $\mathcal{E}$.
3: Find $\hat{\boldsymbol{\eta}} = \arg\min_{\boldsymbol{\eta} \in \mathbb{T}_{N_p}} \mathcal{E}(\boldsymbol{\eta})$ and construct optimal $\Omega^{\mathcal{F}}$ and k using $\hat{\boldsymbol{\eta}}$.
4: Calculate $\hat{u}_j(\mathbf{y})$ according to (3.2) using optimal k and $\Omega^{\mathcal{F}}$, by the distance as in (4.2) and training set X.
5: Find $\hat{h}^X(\mathbf{y}) = \arg\max_{j \in C} \hat{u}_j(\mathbf{y})$.

4.2.6 The PIFW^2kNN algorithm

The PIFWkNN due to the incorporation of class-specific feature weighting and an optimized choice of k is likely to offer immunity against class imbalance to a certain extent. However, if a certain class contains a significantly limited number of training points then it may not be possible for PIFWkNN to find an optimized set of feature weights for that class. Such a situation commonly arises for the minority class(es) where a dearth of sufficient training point is quite regular. Thus, to achieve good performance over the class imbalanced datasets we introduce the PIFW^2kNN classifier. The PIFW^2kNN classifier varies from its predecessor PIFWkNN in two basic aspects.

1. The optimization problem of PIFW^2kNN contains three elements, namely the class specific feature weights, the parameter k, and the class specific weights to handle class imbalance. Thus, we define a candidate solution $\boldsymbol{\eta}' \in \mathbb{T}'$ as a d''-dimensional real valued vector as shown in Figure 4.3. Similar to PIFWkNN such an encoding $\boldsymbol{\eta}'$ is capable to simultaneously optimize all three parameters while offering simple correction

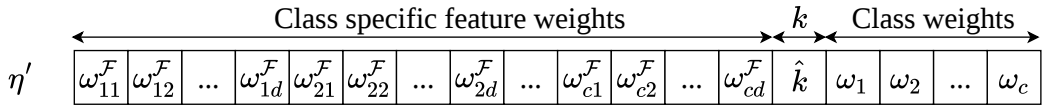


Figure 4.3: Encoding scheme for PIFW^2kNN. Similar to PIFWkNN here also we represent a candidate solution $\boldsymbol{\eta}' \in \mathbb{T}'$ in the form of a d'' dimensional real-valued vector, where $d'' = (d \times c) + c + 1$. Given a d-dimensional c-class dataset the first $(d \times c)$ dimensions of $\boldsymbol{\eta}'$ similar to PIFWkNN represent the class specific feature weights. The $((d \times c) + 1)$-th dimension of $\boldsymbol{\eta}'$ is used to store $\hat{k}$ from which the parameter k is estimated as in PIFWkNN. Whereas, the class weights to compensate for the class imbalance are stored between the $((d \times c) + 2)$-th and $((d \times c) + c + 1)$-th dimension of $\boldsymbol{\eta}'$.

and extraction. Given a solution $\boldsymbol{\eta}'$ the $\Omega^{\mathcal{F}}$ and k follows the similar correction and extraction routine from the PIFWkNN. The class specific weights Ω can be extracted as

$$\Omega = [\eta'^{((d\times c)+2))}, \eta'^{((d\times c)+3))}, \cdots, \eta'^{((d\times c)+c+1))}]. \tag{4.7}$$

To respect the bound constraint one needs ensure that each element of Ω is strictly positive (in case of a violation replacement with a random real number between 0 and 1 may be performed). The class specific weights can be initialized randomly. However, we have observed in our experiments that initializing Ω with GCW usually reaches to a better optima in less time.

2. The objective function in 4.5 is incapable of considering the bias which may be induced in presence of class imbalance. To elaborate, let us think of an example where the training set contains 90 points from the majority class and only 10 representatives from the minority class. In such a scenario even if all the minority training points gets misclassified the objective function in 4.5 will return a commendable error of 0.1. As a remedy we chose to redefine the objective function $\mathcal{E}' : \mathbb{T}' \rightarrow \mathbb{R}^{d''}$ as the geometric mean of the leave-one-out class specific accuracies for all $\mathbf{x} \in X$ as follows:

$$\mathcal{E}'(\boldsymbol{\eta}') = \left(1 - \left(\prod_{j \in C} \frac{1}{N_j} \sum_{\mathbf{x} \in X_j} \mathcal{I}(\hat{h}^{X \setminus \{\mathbf{x}\}}(\mathbf{x}) = j)\right)^{\frac{1}{c}}\right). \tag{4.8}$$

Such an objective function similar to the GMean index has a couple of benefits in our particular scenario. First, $\mathcal{E}'$ only returns a value less than the maximum (i.e. 1, when

there exist at least one class for which no training point has been correctly classified) if and only if all the c classes attain a non-zero class specific accuracy. Second, the error is more inclined to decrease with the increasing accuracy over the worst performing class. In other words $\mathcal{E}'$ attempts to ensure non-zero class specific accuracy for all the classes while stressing on the similar performance over all the classes including the difficult ones.

The complete PIFW^2kNN is described in the following Algorithm 3.

Algorithm 3 The PIFW^2kNN algorithm

Input: The training set X, a test point $\mathbf{y} \in X'$.
Output: The class label $\hat{h}^X(\mathbf{y})$ of $\mathbf{y}$ as predicted by PIFW^2kNN.

1: Randomly initialize population $\mathbb{T}'_{N_p} = [\eta'_{tj}]_{N_p \times d'}$ with real values lying between 0 and 1. The $[\eta'^{((d\times c)+2))}, \eta'^{((d\times c)+3))}, \cdots, \eta'^{((d\times c)+c+1))}]$ dimensions of $\boldsymbol{\eta}'$ which corresponds to Ω can also be initialized by GCW.
2: Evolve $\mathbb{T}'_{N_p}$ by SHADE to minimize objective function $\mathcal{E}'$ as in (4.8). Apply necessary corrections to each of the newly obtained trial solution and extract $\Omega^{\mathcal{F}}$, k, and Ω as described in Section 4.2.5 and 4.2.6 for evaluating them by $\mathcal{E}'$.
3: Find $\hat{\boldsymbol{\eta}}' = \arg\min_{\boldsymbol{\eta}' \in \mathbb{T}'_{N_p}} \mathcal{E}'(\boldsymbol{\eta}')$ and construct optimal $\Omega^{\mathcal{F}}$, k, and Ω using $\hat{\boldsymbol{\eta}}'$.
4: Calculate $\hat{u}_j^{\Omega}(\mathbf{y})$ according to (3.5) using optimal k, $\Omega^{\mathcal{F}}$, and Ω by the distance as in (4.2) and training set X.
5: Find $\hat{h}^X(\mathbf{y}) = \arg\max_{j \in C} \hat{u}_j^{\Omega}(\mathbf{y})$.

4.3 Experiments

In this section, we will undertake a couple of experiments in an attempt to validate the respective improved performances of PIFWkNN and PIFW^2kNN in comparison to a set of contending classifiers.

4.3.1 Datasets used for evaluating PIFWkNN and PIFW2kNN

We evaluate the performance of PIFWkNN on 20 real-world classification datasets having diverse scale, dimensionality, and the number of classes. All the datasets are collected from the UCI Machine Learning Repository (Dua and Graff, 2017). A detailed list of these datasets along with their key properties can be found in Table C.1 in Appendix C.

For validating the efficacy of PIFW^2kNN we use 20 real-world benchmark class imbalanced datasets available from UCI Machine Learning Repository (Dua and Graff, 2017),

KEEL (Triguero et al., 2011), IDA benchmark (Rätsch, 2001), Akbani et al. (2004), Agnostic Learning vs. Prior Knowledge Challenge (Guyon, 2006), and LibSVM (Chang and Lin, 2011). While choosing the datasets we attempt to ensure diversity in terms of scale, dimensionality, number of classes, and the extent of class imbalance. Table C.2 in Appendix C lists these benchmark class imbalanced datasets detailing their various properties.

Table 4.1: Brief description and parameter settings of contending algorithms of PIFWkNN.

Algorithm	Description	Parameter Settings
kNN	Described in Section 3.1	The value of k is optimized by cross-validation over the possible choices $1, 3, 5, 7, 9, 11$, and $\sqrt{N}$.
FkNN	Described in Section 3.1	The value of k is optimized by cross-validation over the possible choices $1, 3, 5, 7, 9, 11$, and $\sqrt{N}$. $k' = k$ and $m = 2$ following Keller et al. (1985).
ENN	A variant of the nearest neighbor rule which unlike kNN, considers not only the nearest neighbors of the test instance but also all the training points that consider that test instance as their nearest neighbor (Tang and He, 2015).	The value of k is optimized by cross-validation over the possible choices 1,3,5,7,9,11 and $\sqrt{N}$.
GA-Fuzzy kNN	First evolutionary approach for FkNN classifier (Hu and Xie, 2005). This optimizes k' and m using binary genetic algorithm.	The value of k is varied between 1,3,5,7,9,11 and $\sqrt{N}$. Crossover probability is varied between 0.7, 0.8 and 0.9. Mutation probability is varied between 0.01, 0.025 and 0.05.
EF-kNN IVFS	An Evolutionary FkNN approach using interval-valued fuzzy sets (Derrac et al., 2016).	The value of k is varied between 1,3,5,7,9,11 and $\sqrt{N}$. Interval-valued fuzzy set is used to calculate the membership of training instances of FkNN based on multiple choices of two parameters, k' and m. Initial divergence rate (Eshelman, 1991) is varied between 0.2, 0.35 and 0.5.
CW	This algorithm uses a class-dependent feature weighting method and the weight matrix is learned by gradient descent algorithm (Paredes and Vidal, 2006).	The results are quoted from (Mateos-García et al., 2012)
DE4	This algorithm uses a hybrid weighting approach, which combines the global feature weighting and class weighting and the weights are optimized using differential evolution (Al-Sukker et al., 2010).	The results are quoted from (Mateos-García et al., 2012)
kLDEDW	It is a nearest neighbor method that uses label dependent feature weighting technique and optimizes the weight matrix and the value of k using genetic algorithm (BLX-α crossover technique) (Mateos-García et al., 2012).	The results are quoted from (Mateos-García et al., 2012).

4.3.2 Experimental protocol

To validate the efficacy of PIFWkNN we compare its performance with eight kNN and FkNN variants often using EvO techniques for optimizing the different associated parameters. The

eight contending classifiers contenders are listed in Table 4.1 along with a brief description of their working strategy and parameter search space. Interestingly, from Table 4.1 it can be observed that many improved variants of kNN and FkNN actually ends up introducing additional tunable parameters while attempting to optimize the others. Whereas, PIFWkNN handles all its parameters in a self-adaptive manner. In all cases, the results are reported in terms of Accuracy averaged over 10-fold cross-validation. Additionally, in case of PIFWkNN, GA-Fuzzy-kNN, and EF-kNN IVFS i.e. methods using EvO techniques the classifier performance for each fold is averaged over five independent runs in an attempt to mitigate the effect of any randomization bias. For CW, DE4, and kLDEDW the mean performance on 10 fold cross-validation is quoted from the corresponding article. Moreover, in this experiment, our target is to evaluate the efficacy of PIFWkNN in optimizing an effective set of class-specific feature weights along with a global choice of k. Thus, we compare the performance of the proposed PIFWkNN against the best result obtained by each of the contending classifiers. The parameter setting for which the contenders achieve the best performance is either detailed in Table C.3 in Appendix C or can be found in the corresponding article.

To evaluate the effectiveness of PIFW^2kNN in retaining its performance in presence of class imbalance we compare it against six kNN and FkNN variants tailored to serve a similar purpose. The chosen competing classifiers are dyn$k$NN (Garcia-Pedrajas et al., 2015), CCNND (Kriminger et al., 2012), CWkNN (Dubey and Pudi, 2013), WkNN+GCW, FkNN+SMOTE, and FkNN+RUS. The parameter tuning strategy for dyn$k$NN is followed from the original article, CCNND is run with the nearest neighbor while the number of neighbors in CWkNN is set to $\sqrt{N}$. For FkNN+SMOTE and FkNN+RUS, the parameter settings are kept similar to those described in Section 3.4.2.2. In all cases the results are reported in terms of GMean and ACSA averaged over 10-fold cross-validation (PIFW^2kNN are additionally run for five times on each fold to reduce bias induced by randomization) and all considered parameter choices.

In both experiments, we report the average rank achieved by a classifier as an indication of its consistency. Further, to validate if the performance improvement achieved by the proposed techniques is statistically significant we perform WRS and WSR tests by setting the proposed classifier as control. Similar to Section 3.4.2.2 the result of the WRS test is reported in terms of Win (W), Tie (T), and Loss (L). To recall, win signify the number of

datasets on which the contending classifier performed significantly worse than the control method. Loss indicates the number of datasets on which the performance of the competing classifier is significantly better than the control. Finally, Tie denotes the number of cases when the contender demonstrates a statistically comparable performance with the control. The WSR test further verifies if the contender performs significantly different from the proposed methods. For both WSR and WRS the null hypothesis is only rejected with a 95% confidence level.

Table 4.2: Comparison of PIFWkNN with improved variants of kNN and FkNN on real world datasets.

Algorithm	Average Accuracy	Average Rank	WRS W-T-L	WSR
PIFWkNN (Ours)	**0.84**	**1.67**	CM	CM
kNN	0.81	4.82	13-7-0	$\mathcal{H}_1$
FkNN	0.79	5.57	14-6-0	$\mathcal{H}_1$
ENN	0.76	6.77	17-3-0	$\mathcal{H}_1$
GA-Fuzzy kNN	0.81	5.42	15-4-1	$\mathcal{H}_1$
EF-kNN IVFS	0.81	5.05	15-5-0	$\mathcal{H}_1$
kLDEDW	0.82	3.42	N/A	N/A
CW	0.80	5.25	N/A	N/A
DE4	0.75	7.00	N/A	N/A

CM: Denotes the control method PIFWkNN. N/A: Statistical tests such as WRS and WSR cannot be performed as the results are quoted from the corresponding original article. $\mathcal{H}_1$ is the alternative hypothesis denoting that the performance of the contender is significantly different from PIFW-kNN on 20 datasets as indicated by WSR. $\mathcal{H}_0$ is the null hypothesis denoting that the performance of the contender is statistically comparable with PIFW-kNN on 20 datasets as indicated by WSR.

4.3.3 Comparison of PIFWkNN with other classifiers on real-world datasets

We summarize the performance of PIFWkNN and eight other contending algorithms over 20 real-world datasets in terms of Accuracy in Table 4.2. A closer inspection of Table 4.2 reveals that the self-adaptive PIFWkNN achieves the highest average Accuracy and the lowest average rank over the 20 datasets. This finding attests to the better performance of PIFWkNN highlighting its consistency compared to its contenders. The performance of PIFWkNN is followed by kLDEDW another EvO based approach which also incorporates feature weighting while finding the neighborhood. This observation acts as evidence in support of the usefulness of optimizing class-specific feature weights and global choice of k by EvO techniques. The WRS result suggests that PIFWkNN can significantly outperform all of its contenders

in a majority of cases. In fact, PIFWkNN achieves either statistically better or at least comparable Accuracy with the contending algorithms on all the datasets except a single one. However, the loss suffered by PIFWkNN while competing with GA-Fuzzy kNN on only a single dataset cannot lead us to a general conclusion as such a performance decline of the proposed classifier is not observed on any other occasion. Moreover, the results of the WSR test suggest that the performance of PIFWkNN is significantly different from its contenders over all the 20 datasets further validating the improvement offered by the proposed. The complete results on individual datasets are detailed in Table C.4 in Appendix C.

4.3.4 Comparison of PIFW^2kNN with other tailored classifiers on real world-class imbalanced benchmark datasets

In Table 4.3 we compare the proposed PIFW^2kNN in terms of GMean and ACSA with six variants of kNN and FkNN tailored to handle class imbalance on 20 real-world imbalanced benchmark datasets. From Table 4.3 it is evident that PIFW^2kNN achieves the highest average GMean and ACSA among the seven competing classifiers indicating the improved performance of the proposed. Furthermore, PIFW^2kNN also achieves the best average rank for both GMean and ACSA attesting to its consistency. Interestingly, the performances of PIFW^2kNN in terms of GMean and ACSA do not vary by a large amount which indicates that the proposed classifier offers better resilience against class imbalance as it achieves almost similar class-specific accuracies. The performance of PIFW^2kNN is followed by WkNN+GWS and FkNN+SMOTE. This observation supports the efficiency of the popular GWS approach as well as attest to the efficacy of the simple SMOTE based oversampling. This observation is also backed up by the WRS result which shows that both WkNN+GWS and FkNN+SMOTE achieve a higher number of statistically comparable performances with PIFW^2kNN among the contenders. Even though CCNND manages to perform significantly better than PIFW^2kNN on a single dataset the classifier achieves lower GMean, ACSA, along with respective ranks on average indicating a poor consistency. Moreover, the WSR test suggests that the proposed PIFW^2kNN performs significantly different from the other classifiers on the 20 class imbalanced datasets in terms of both indices.

Table 4.3: Comparison of PIFWkNN with tailored variants of kNN and FkNN on real world class imbalanced benchmark datasets.

Algorithm	GMean				ACSA			
	Average Performance	Average Rank	WRS W-T-L	WSR	Average Performance	Average Rank	WRS W-T-L	WSR
PIFW^2kNN (Ours)	**0.80**	**1.45**	CM	CM	**0.81**	**1.78**	CM	CM
dyn$k$NN	0.66	4.65	18-2-0	$\mathcal{H}_1$	0.73	4.98	17-3-0	$\mathcal{H}_1$
CCNND	0.70	4.45	16-3-1	$\mathcal{H}_1$	0.75	4.73	15-4-1	$\mathcal{H}_1$
CWkNN	0.35	6.68	19-1-0	$\mathcal{H}_1$	0.60	6.23	19-1-0	$\mathcal{H}_1$
WkNN+GCW	0.73	2.85	11-8-1	$\mathcal{H}_1$	0.77	3.20	10-8-2	$\mathcal{H}_1$
FkNN+SMOTE	0.73	3.58	12-8-0	$\mathcal{H}_1$	0.78	3.38	10-9-1	$\mathcal{H}_1$
FkNN+RUS	0.67	4.35	17-3-0	$\mathcal{H}_1$	0.76	3.73	13-7-0	$\mathcal{H}_1$

CM: Denotes the control method PIFW^2kNN. $\mathcal{H}_1$ is the alternative hypothesis denoting that the performance of the contender is significantly different from PIFW-kNN on 20 datasets as indicated by WSR. $\mathcal{H}_0$ is the null hypothesis denoting that the performance of the contender is statistically comparable with PIFW-kNN on 20 class imbalanced datasets as indicated by WSR.

4.4 Discussion

In this chapter, we have proposed a parameter independent technique of feature weighted fuzzy kNN which uses a self-adaptive variant of DE to optimize the feature weights as well as the value of k. In our model, the weights of the features are dependent on the label of the training points. The optimizing of the choice of k and the class-specific feature weight matrix are formulated as a single problem that can be solved by SHADE using the proposed encoding scheme. The proposed model has been tested on 20 real-world datasets of varying properties and compared with 8 state-of-the-art and popular classifiers. From our experiments, it can be concluded that the proposed PIFW-kNN can not only perform independently (without requiring any additional parameter tuning) but can also show a significant edge over some of the parameter-free popular classifiers. Further, to efficiently handle class imbalance we have proposed PIFW^2kNN which incorporates additional class-specific weights in the PIFWkNN framework. This can be achieved by modifying the encoding scheme as well as the objective function to provide simultaneous optimization of the class weights. The efficacy of PIFW^2kNN has been validated by experimental comparison with six kNN and FkNN variants tailored for handling class imbalance on 20 real-world benchmark imbalanced datasets.

Chapter 5

Generative Adversarial Minority Oversampling

Summary

Class imbalance is a long-standing problem relevant to several real-world applications of deep learning. Oversampling techniques, which are effective for handling class imbalance in classical learning systems, can not be directly applied to end-to-end deep learning systems. In this chapter, we propose a three-player adversarial game between a convex generator, a multi-class classifier network, and a real/fake discriminator to perform oversampling in deep learning systems. The convex generator generates new samples from the minority classes as convex combinations of existing instances, aiming to fool both the discriminator as well as the classifier into misclassifying the generated samples. Consequently, the artificial samples are generated at critical locations near the peripheries of the classes. This, in turn, adjusts the classifier induced boundaries in a way that is more likely to reduce misclassification from the minority classes. Extensive experiments on multiple class imbalanced image datasets establish the efficacy of our proposal.

5.1 Introduction

5.1.1 Overview

Over the years, the machine learning community has devised many methods for tackling class imbalance (Krawczyk, 2016; Branco et al., 2016). However, only a few of these techniques have been extended to deep learning even though class imbalance is fairly persistent is such

networks, severely affecting both the feature extraction as well as the classification process (Xie and Tu, 2015; Huang et al., 2016; Xie and Tu, 2017; Buda et al., 2018; Khan et al., 2018; Johnson and Khoshgoftaar, 2019). The existing solutions (Huang et al., 2016; Chung et al., 2016; Wang et al., 2016; Lin et al., 2017a; Bulò et al., 2017) for handling class imbalance in deep neural networks mostly focus on cost tuning to assign suitably higher costs to minority instances. Another interesting class of approaches (Yan et al., 2015; Dong et al., 2018) focuses on constructing balanced subsamples of the dataset. Wang et al. (2017b) proposed a novel meta-learning scheme for imbalanced classification. It is interesting to note that oversampling techniques like SMOTE (Chawla et al., 2002) have not received much attention in the context of deep learning, despite being very effective for classical systems (Fernández et al., 2018). This is because deep feature extraction and classification are performed in an end-to-end fashion, making it hard to incorporate oversampling which is typically done subsequent to feature extraction. An attempt to bridge this gap was made by Ando and Huang (2017) in their proposed deep oversampling framework (DOS). However, DOS uniformly oversamples the entire minority class and is not capable of concentrating the artificial instances in difficult regions. Additionally, the performance of DOS depends on the choice of the class-wise neighborhood sizes, which must be determined by costly parameter tuning.

Generative adversarial networks (GANs) are a powerful subclass of generative models that have been successfully applied to image generation. This is due to their capability to learn a mapping between a low-dimensional latent space and a complex distribution of interest, such as natural images (Goodfellow et al., 2014; Mirza and Osindero, 2014; Radford et al., 2015; Odena et al., 2017). The approach is based on an adversarial game between a generator that tries to generate samples that are similar to real samples and a discriminator that tries to discriminate between real training samples and generated samples. The success of GANs as generative models has led Douzas and Bacao Douzas and Bacao (2018), Ali-Gombe and Elyan (2019), and Liu et al. (2019) to investigate their effectiveness in oversampling the minority class(es). However, attempting to oversample the minority class(es) using GANs can lead to boundary distortion (Santurkar et al., 2018), resulting in a worse performance on the majority class [1]. Moreover, the generated points are likely to lie near the mode(s) of the

[1] Boundary distortion is a form of co-variance shift observed in GANs where the samples generated near the periphery of a class may lack diversity. Such a phenomenon may lead to a significant diversion of the distribution of the generated samples from the corresponding original class distribution.

minority class(es) (Srivastava et al., 2017), while new points around the class boundaries are required for learning reliable discriminative (classification) models (Han et al., 2005; He et al., 2008). Mariani et al. (2018) also attempted to utilize GAN coupled with an autoencoder (Ballard, 1987) for artificially balancing a class imbalanced training set. Their generative model is expected to generate new images for the minority class by learning useful features from the abundant majority instances. However, their generative model may fail to serve its purpose if the majority and minority classes do not share enough common features, limiting the applicability of their technique.

5.1.2 Background

The success of SMOTE (Chawla et al., 2002, 2003) has inspired several improvements. For example, Han et al. (2005) and Bunkhumpornpat et al. (2009) attempt to selectively oversample minority class points lying close to the class boundaries. Works like (He et al., 2008; Lin et al., 2013; Barua et al., 2014), on the other hand, asymmetrically oversample the minority class such that more synthetic points are generated surrounding the instances which are difficult to classify. Although these methods achieved commendable improvement on traditional classifiers, they can neither be extended to deep learning techniques nor be applied to images, respectively due to the end-to-end structure of deep learning algorithms and a lack of proper notion of distance between images.

Extending GANs for semi-supervised learning works like (Kumar et al., 2017; Salimans et al., 2016) fused a c-class classifier with the discriminator by introducing an extra output line to identify the fake samples. On the other hand, (Springenberg, 2015) proposed a c-class discriminator which makes uncertain predictions for fake images. Additionally, (Odena et al., 2017) proposed a shared discriminator-cum-classifier network which makes two separate sets of predictions using two different output layers. These approaches can loosely be considered to be related to GAMO as these also incorporate a classifier into the adversarial learning scheme.

5.1.3 Motivation

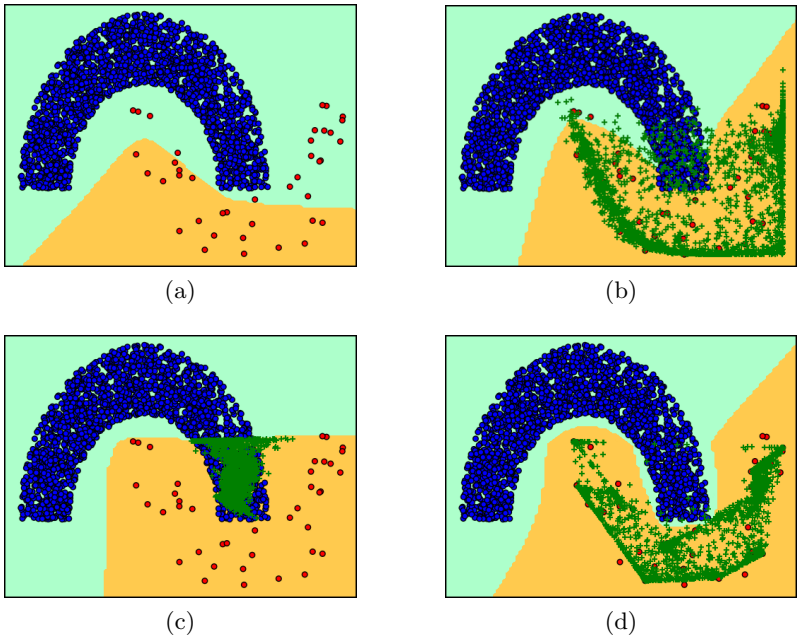
As mentioned earlier in Section 5.1.1 deep neural networks are susceptible to the class imbalance in the training set. A naive way to tackle this issue maybe balancing the training set by generating artificial data instances. However, in the case of images, the task becomes difficult as the traditional oversampling approaches are not capable to handle to such data in an end-to-end network. One may attempt to use a conditional GAN (cGAN) (Mirza and Osindero, 2014) to generate new instances for the minority class(es). However, the cGAN may fail to generate useful diverse new samples due to mode collapse[2] and boundary distortion, especially in the absence of enough complex image samples from the minority class. Further, the generator will not be able to help the classifier by generating points in the difficult to learn regions without a proper channel of communication. One may address these issues is to design a generator that will be adversarially connected with the classifier and will generate a new sample as a convex combination of the existing instances of a class. However, if the classes are not convex by nature then the samples generated by such convex generators may fail to adhere to the class distribution. This can be solved by introducing an additional discriminator which will guide the convex generator to properly follow the class distribution. We illustrate this in the following Example 5.1.

Example 5.1. *We take a traditional "two-moon" dataset, which contains 2 horseshoe-shaped classes arranged in an interleaved fashion. The reasons for choosing such a dataset are two-fold. First, the classes are non-convex. Second, the classes are only separable by a non-linear class boundary. We generate a class imbalanced problem having an IR of 40 by respectively sampling 2000 and 50 points from the first and the second class. In the sub-figures under Figure 5.1 the majority class is represented by blue dots, the minority class is denoted by red dots, while green cross (+) is used to illustrate the generated points for the minority class. Further, the sea-green and yellow regions denote the areas where the classifiers respectively label a point by the majority and the minority class. We illustrate four scenarios in Figure 5.1. First, the performance of a classifier network H on the class imbalanced training set in Figure 5.1a. Second, the performance of H on a training set balanced by cGAN in Figure*

[2]Mode collapse is a form of co-variance shift observed in GANs where the generated samples remain within a set of limited instances even when the input changes over a wide range. This may happen when the discriminator gets trapped to a local optima and the generator can easily fool it by perfecting the generation of only a few instances.

5.1b. Third, the performance of H on a training set balanced by a convex generator G in Figure 5.1c. Fourth, in Figure 5.1d we illustrate the performance of H on a training set balanced by a convex generator G additionally guided by a discriminator D.

We can see in Figure 5.1d that the classifier H achieves the best performance when D aids the generator G to generate the new samples in the critical locations while respecting the class distribution.



(a) (b) (c) (d)

Figure 5.1: Illustration using a "two-moon" dataset: (a) Imbalanced classification with an unaided classifier network H results in misclassification of the minority class instances. (b) Artificial minority points generated using conditional GAN help to improve the result on the minority class but bleed into the majority class, affecting the performance on the latter. (c) New points are generated by training a convex generator G alternatingly with H. This is a two player adversarial game where G attempts to generate samples which are hard for H to correctly classify. This results in ideal performance on the minority class, but at the cost of misclassifying the majority class as G does not adhere to the distribution of the minority class. (d) Ideal performance on both classes is achieved by further incorporating an additional discriminator D to induce fidelity to the minority class distribution and to limit bleeding into majority class territory.

5.1.4 Contributions of Chapter 5

Hence, in this chapter, we propose (in Section 5.2) a novel end-to-end feature-extraction-classification framework called Generative Adversarial Minority Oversampling (GAMO) which

employs adversarial oversampling of the minority class(es) to mitigate the effects of class imbalance[3]. The contributions made in this chapter differ from the existing literature in the following ways:

1. Unlike existing deep oversampling schemes (Ando and Huang, 2017; Douzas and Bacao, 2018), GAMO is characterized by a three-player adversarial game among a convex generator G, a classifier network H, and a discriminator D.

2. Our approach is fundamentally different from the existing adversarial classification schemes where the generator works in harmony with the classifier to fool the discriminator (Salimans et al., 2016; Kumar et al., 2017; Springenberg, 2015; Odena et al., 2017). In GAMO our convex generator G attempts to generate new samples which will fool the classifier H as well as the discriminator D.

3. Unlike the generator employed in GAN (Goodfellow et al., 2014), we constrain G to conjure points within the convex hull of the class of interest. Additionally, the discriminator D further ensures that G adheres to the class distribution for non-convex classes. Consequently, the adversarial contention with H pushes the conditional distribution(s) learned by G towards the periphery of the respective class(es), thus helping compensate for class imbalance effectively.

4. In contrast to methods like (Chawla et al., 2002; Douzas and Bacao, 2018), G can oversample different localities of the data distribution to different extents based on the gradients obtained from H.

5. For applications requiring a balanced training set of images, we also propose a technique called GAMO2pix (Section 5.4) that can generate realistic images from the synthetic instances generated by GAMO in the distributed representation space.

We undertake an ablation study as well as evaluate the performance of our method compared to the state-of-the-art in Section 5.3.

[3]Codes & data at: https://github.com/SankhaSubhra/GAMO.

5.2 Proposed Method

Let us consider a c-class classification problem with a training dataset $X \subset \mathbb{R}^d$ (of images vectorized either by flattening or by a convolutional feature extraction network F). Let the prior probability of the i-th class be P_i, where $i \in C = \{1, 2, \cdots c\}$. Without loss of generality, we consider the classes to be ordered such that $P_1 \leq P_2 \leq \cdots < P_c$. We intend to train a classifier H having c output lines, where the i-th output $H_i(\mathbf{x})$ predicts the probability of any $\mathbf{x} \in X$ to be a member of the i-th class.

5.2.1 Adversarial Oversampling

Our method plays an adversarial game between a classifier that aims to correctly classify the data points and a generator attempting to spawn artificial points which will be misclassified by the classifier. The idea is that generating such difficult points near the fringes of the minority class(es) will help the classifier to learn class boundaries which are more robust to class imbalance. In other words, the performance of the classifier will adversarially guide the generator to generate new points at those regions where the minority class under concern is prone to misclassification. Moreover, the classifier will aid the generator to adaptively determine the concentration of artificial instances required to improve the classification performance in a region, thus relieving the user from tuning the amount of oversampling. Instead, we only need to fix the number of points to be generated to the difference between the number of points in the majority class and that of the (respective) minority class(es).

5.2.2 Convex Generator

The generator tries to generate points which will be misclassified by the classifier. Hence, if left unchecked, the generator may eventually learn to generate points which do not coincide with the distribution of the intended minority class. This may help improve the performance on the concerned minority class but will lead to high misclassification from the other classes. To prevent this from happening, we generate the new points only as convex combinations of the existing points from the minority class in question. This will restrict the generated distribution within the convex hull of the real samples from the (respective) minority class(es). Since the generator attempts to conjure points that are difficult for the classifier, the points

are generated near the peripheries of the minority class(es).

Our convex generator G comprises of two modules: a Conditional Transient Mapping Unit ($cTMU$) and a set of class-specific Instance Generation Units (IGU), which we propose to limit the model complexity. The $cTMU$ network learns a mapping t, conditioned on class i, from a λ_l-dimensional latent space to an intermediate space. The IGU_i, on the other hand, learns a mapping g_i from the $cTMU$ output space to a vector $g_i(t(\mathbf{z}|i))$ of N_i convex weights using softmax activation, where $\mathbf{z}$ is a latent variable drawn from a standard normal distribution. Thus, $g_i(t(\mathbf{z}|i)) \geq \mathbf{0}$, and $g_i(t(\mathbf{z}|i))^T\mathbf{1} = 1$. Hence, G can generate a new d-dimensional sample for the i-th class as a convex combination of the data points in X_i,

$$G(\mathbf{z}|i) = g_i(t(\mathbf{z}|i))^T X_i. \tag{5.1}$$

Formally, the adversarial game played by the proposed classifier-convex generator duo poses the following optimization problem, when cross entropy loss is considered:

$$\min_G \max_H J(G,H) = \sum\nolimits_{i\in C} J_i, \tag{5.2}$$

$$\begin{aligned}
&\text{where} \quad J_i = (J_{i1} + J_{i2} + J_{i3} + J_{i4}),\\
&J_{i1} = P_i \mathbb{E}_{\mathbf{x}\sim p_i}[\log H_i(\mathbf{x})],\\
&J_{i2} = \sum\nolimits_{j\in C\setminus\{i\}} P_j \mathbb{E}_{\mathbf{x}\sim p_j}[\log(1 - H_i(\mathbf{x}))],\\
&J_{i3} = (P_c - P_i)\mathbb{E}_{G(\mathbf{z}|i)\sim p_i^{(g)}}[\log H_i(G(\mathbf{z}|i))], \text{and},\\
&J_{i4} = \sum_{j\in C\setminus\{i\}} (P_c - P_j)\mathbb{E}_{G(\mathbf{z}|j)\sim p_j^{(g)}}[\log(1 - H_i(G(\mathbf{z}|j)))],
\end{aligned}$$

while p_i and $p_i^{(g)}$ respectively denote the real and generated class conditional probability distributions of the i-th class.

The two-player minimax game formalized in (5.2) is played between a classifier H and a generator G. H attempts to correctly classify all real as well as generated points belonging to all the classes. Whereas, G strives to generate sample(s) which have a high probability of being classified by H into all other classes. To demonstrate how such an adversarial game can aid H to learn a better class boundary, we illustrate its chronological progression in a more explanatory manner in Figure 5.2. In Theorem 5.1, we show that the optimization problem

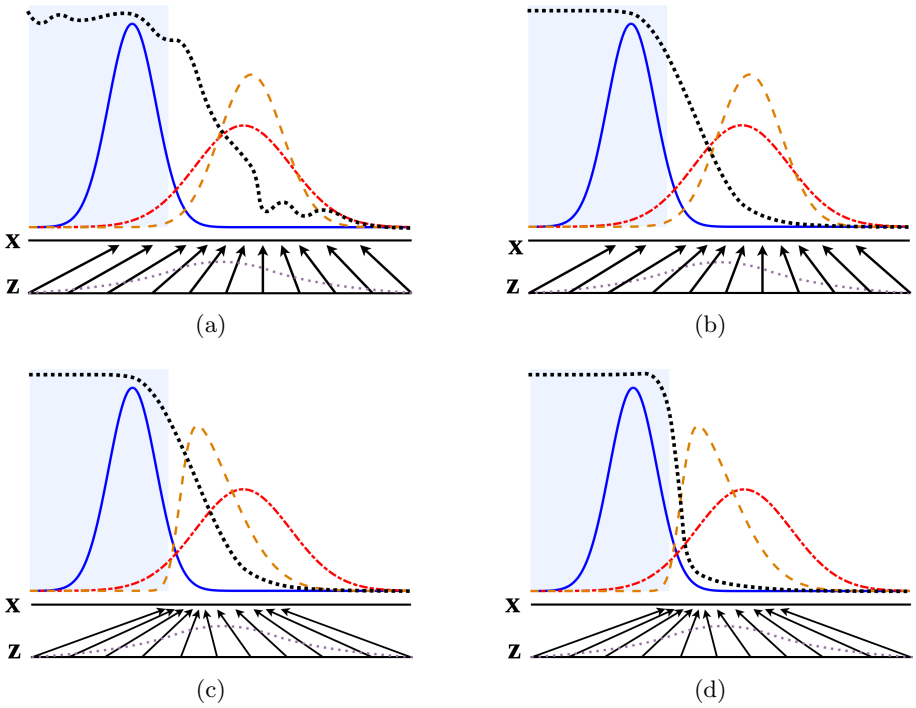


Figure 5.2: GAMO functions by simultaneously updating the classifier H and the generator G. The classification function (black, dotted line) is trained to correctly classify samples from the majority class distribution p_{maj} (blue, solid line), the real minority class distribution p_{min} (red, dots and dashes) as well as the generated minority distribution $p_{min}^{(g)}$ (brown, dashed line). The generator, on the other hand, is trained to generate minority samples which will be misclassified by H. The upward arrows show how the generator learns the mapping $\mathbf{x} = G(\mathbf{z})$ from a standard normal distribution (mauve, dotted line) in the latent space to convex combinations of the real minority instances from the minority class. The ideal classification function is shown as a blue highlight in the background. (a) Let us consider an initial adversarial pair: the generated distribution $p_{min}^{(g)}$ is similar to the real distribution of the minority class p_{min} and H is an inaccurate classifier. (b) H is trained to properly classify the samples from the three distributions p_{maj}, p_{min}, and $p_{min}^{(g)}$; resulting in a non-ideal trained classifier which is biased in favor of the majority class. (c) After an update to G, the gradient of H has guided $G(\mathbf{z})$ to flow to regions that are more likely to be misclassified by H. (d) Thereafter, retraining H results in a classifier much closer to the ideal classifier due to the increased number of minority samples near the boundary of the two classes.

in (5.2) is equivalent to minimizing a sum of the Jensen-Shannon divergences.

Theorem 5.1. *Optimizing the objective function J is equivalent to the problem of minimizing*

the following summation of Jensen-Shannon (JS) divergences:

$$\sum_{i=1}^{c} JS\Big(\big(P_i p_i + (P_c - P_i)p_i^{(g)}\big)\Big|\Big|\sum_{\substack{j\neq i\\ j=1}}^{c}\big(P_j p_j + (P_c - P_j)p_j^{(g)}\big)\Big)$$

Proof. For simplicity and without loss of generality we focus on a single minority class, say the i-th one. Then, we can start by finding that $H_i^*(\mathbf{x})$, which will maximize J_i (the component of J corresponding to the i-th class) for a given G. Therefore, we first find the partial differentiation of J_i, with respect to $H_i(\mathbf{x})$, as follows:

$$\frac{\partial J_i}{\partial H_i(\mathbf{x})} = \frac{P_i p_i}{H_i(\mathbf{x})} - \frac{\sum_{j\in C\setminus\{i\}} P_j p_j}{1 - H_i(\mathbf{x})} + \frac{(P_c - P_i)p_i^{(g)}}{H_i(\mathbf{x})} - \frac{\sum_{j\in C\setminus\{i\}}(P_c - P_j)p_j^{(g)}}{1 - H_i(\mathbf{x})} \tag{5.3}$$

Equating (5.3) to 0, and solving it for $H_i(\mathbf{x})$ gives,

$$H_i^*(\mathbf{x}) = \frac{P_i p_i + (P_c - P_i)p_i^{(g)}}{\sum_{l=1}^{c}\big(P_l p_l + (P_c - P_l)p_l^{(g)}\big)} \tag{5.4}$$

Plugging in the value of $H_i^*(\mathbf{x})$ from (5.4) back in J_i, we get,

$$
\begin{aligned}
J_i = & \int P_i p_i \log \frac{P_i p_i + (P_c - P_i) p_i^{(g)}}{\sum_{l=1}^{c} \left(P_l p_l + (P_c - P_l) p_l^{(g)}\right)} dx + \\
& \int \sum_{j \in C \setminus \{i\}} P_j p_j \log \frac{\sum_{j \in C \setminus \{i\}} \left(P_j p_j + (P_c - P_j) p_j^{(g)}\right)}{\sum_{l=1}^{c} \left(P_l p_l + (P_c - P_l) p_l^{(g)}\right)} dx + \\
& \int \left((P_c - P_i) p_i^{(g)}\right) \log \frac{P_i p_i + (P_c - P_i) p_i^{(g)}}{\sum_{l=1}^{c} \left(P_l p_l + (P_c - P_l) p_l^{(g)}\right)} dx + \\
& \int \sum_{j \in C \setminus \{i\}} \left((P_c - P_j) p_j^{(g)} + P_j p_j\right) \log \frac{\sum_{j \in C \setminus \{i\}} \left(P_j p_j + (P_c - P_j) p_j^{(g)}\right)}{\sum_{l=1}^{c} \left(P_l p_l + (P_c - P_l) p_l^{(g)}\right)} dx \\
J_i = & \int \left(P_i p_i + (P_c - P_i) p_i^{(g)}\right) \log \frac{P_i p_i + (P_c - P_i) p_i^{(g)}}{\frac{1}{2} \sum_{l=1}^{c} \left(P_l p_l + (P_c - P_l p_l^{(g)}\right)} dx - \\
& \log 2 \int \left(P_i p_i + (P_c - P_i) p_i^{(g)}\right) dx + \\
& \int \sum_{j \in C \setminus \{i\}} \left(P_j p_j + (P_c - P_j) p_j^{(g)}\right) \log \frac{\sum_{j \in C \setminus \{i\}} \left(P_j p_j + (P_c - P_j) p_j^{(g)}\right)}{\frac{1}{2} \sum_{l=1}^{c} \left(P_l p_l + (P_c - P_l) p_l^{(g)}\right)} dx - \\
& \log 2 \int \sum_{j \in C \setminus \{i\}} \left(P_j p_j + (P_c - P_j) p_j^{(g)}\right) dx \\
J_i = & 2JS\Big(\left(P_i p_i + (P_c - P_i) p_i^{(g)}\right) \Big|\Big| \sum_{j \in C \setminus \{i\}} \left(P_j p_j + (P_c - P_j) p_j^{(g)}\right)\Big) - c P_c \log 2 \qquad (5.5)
\end{aligned}
$$

From (5.5), ignoring the constant scalar multiplicative factor and the additive factor $-cP_c \log 2$ (also a constant for a given problem) we can conclude that

$$
\min_G \max_H J \sim \min_{p^{(g)}} \sum_{i=1}^{c} JS\Big(\left(P_i p_i + (P_c - P_i) p_i^{(g)}\right) \Big|\Big| \sum_{j \neq i} \left(P_j p_j + (P_c - P_j) p_j^{(g)}\right)\Big), \qquad (5.6)
$$

which completes the proof. □

The behavior of the proposed approach can be understood by interpreting Theorem 5.1. The optimization problem aims to bring the generated distribution, for a particular class, closer to the generated as well as real distributions for all other classes. Since the real distributions are static for a fixed dataset, the optimization problem in Theorem 5.1 essentially attempts to move the generated distributions for each class closer to the real distributions for all other classes. This is likely to result in the generation of ample points near the peripheries,

which are critical to combating class imbalance. While doing so, the generated distributions for all classes also strive to come closer to each other. However, the generated distributions for the different classes do not generally collapse upon each other, being constrained to remain within the convex hulls of the respective classes.

5.2.3 Additional Discriminator

While the generator only generates points within the convex hull of the samples from the minority class(es), the generated points may still be placed at locations within the convex hull which do not correspond to the distribution of the intended class (recall Figure 5.1c). This is likely to happen if the intended minority class(es) are non-convex in shape. Moreover, we know from Theorem 5.1 that the generated distributions for different minority classes may come close to each other if the respective convex hulls overlap. To solve this problem, we introduce an additional conditional discriminator which ensures that the generated points do not fall outside the actual distribution of the intended minority class(es). Thus, the final adversarial learning system proposed by us consists of three players, viz. a multi-class classifier H, a conditional discriminator D which given a class aims to distinguish between real and generated points, and a convex generator G that attempts to generate points which, in addition to being difficult for H to correctly classify, are also mistaken by D to be real points sampled from the given dataset. The resulting three-player minimax game is formally presented in (5.7).

$$\begin{aligned}
&\min_G \max_H \max_D \hat{J}(G,H,D) = \sum\nolimits_{i \in C} \hat{J}_i, \\
&\text{where, } \hat{J}_i = (J_{i1} + J_{i2} + J_{i3} + J_{i4} + \hat{J}_{i1} + \hat{J}_{i2}), \\
&\hat{J}_{i1} = P_i \mathbb{E}_{\mathbf{x} \sim p_i}[\log D(\mathbf{x}|i)], \text{and,} \\
&\hat{J}_{i2} = (P_c - P_i)\mathbb{E}_{G(\mathbf{z}|i) \sim p_i^{(g)}}[\log(1 - D(G(\mathbf{z}|i)|i))].
\end{aligned} \tag{5.7}$$

5.2.4 Least-Square Formulation

Mao et al. (2017) showed that replacing the popular cross entropy loss in GAN with least square loss can not only produce better quality images but also can prevent the vanishing gradient problem to a greater extent. Therefore, we also propose a variant of GAMO using

the least square loss, which poses the following optimization problem:

$$\min_H L_H = \sum\nolimits_{i \in C} (L_{i1} + L_{i2} + L_{i3} + L_{i4}), \tag{5.8}$$

$$\min_D L_D = \sum\nolimits_{i \in C} (L_{i5} + L_{i6}), \tag{5.9}$$

$$\min_G L_G = \sum\nolimits_{i \in C \setminus \{c\}} (L_{i7} + L_{i8} + L_{i9}), \tag{5.10}$$

where, $L_{i1} = P_i \mathbb{E}_{\mathbf{x} \sim p_i}[(1 - H_i(\mathbf{x}))^2]$,

$L_{i2} = \sum\nolimits_{j \in C \setminus \{i\}} P_j \mathbb{E}_{\mathbf{x} \sim p_j}[(H_i(\mathbf{x}))^2]$,

$L_{i3} = (P_c - P_i) \mathbb{E}_{G(\mathbf{z}|i) \sim p_i^{(g)}}[(1 - H_i(G(\mathbf{z}|i)))^2]$,

$L_{i4} = \sum\nolimits_{j \in C \setminus \{i\}} (P_c - P_j) \mathbb{E}_{G(\mathbf{z}|j) \sim p_j^{(g)}}[(H_i(G(\mathbf{z}|j)))^2]$,

$L_{i5} = P_i \mathbb{E}_{\mathbf{x} \sim p_i}[(1 - D(\mathbf{x}|i))^2]$,

$L_{i6} = (P_c - P_i) \mathbb{E}_{G(\mathbf{z}|i) \sim p_i^{(g)}}[(D(G(\mathbf{z}|i)|i))^2]$,

$L_{i7} = \mathbb{E}_{G(\mathbf{z}|i) \sim p_i^{(g)}}[(H_i(G(\mathbf{z}|i)))^2]$,

$L_{i8} = \sum\nolimits_{j \in C \setminus \{i,c\}} \mathbb{E}_{G(\mathbf{z}|j) \sim p_j^{(g)}}[(1 - H_i(G(\mathbf{z}|j)))^2]$,

$L_{i9} = \mathbb{E}_{G(\mathbf{z}|i) \sim p_i^{(g)}}[(1 - D(G(\mathbf{z}|i)|i))^2]$.

5.2.5 Putting it all together

The model for the GAMO framework is detailed in Figure 5.3, while the complete algorithm is described in Algorithm 4. To ensure an unbiased training for H and D we generate artificial points for the i-th class with probability $(P_c - P_i)$ to compensate for the effect of imbalance. On the other hand, to also ensure unbiased training for G we use samples from all classes with equal probability.

5.3 Experiments

We evaluate the performance of a classifier in terms of two indices that are not biased toward any particular class (Sokolova and Lapalme, 2009), namely ACSA (Huang et al., 2016; Wang et al., 2017b) and GMean (Kubat et al., 1997; Branco et al., 2016). All our experiments have

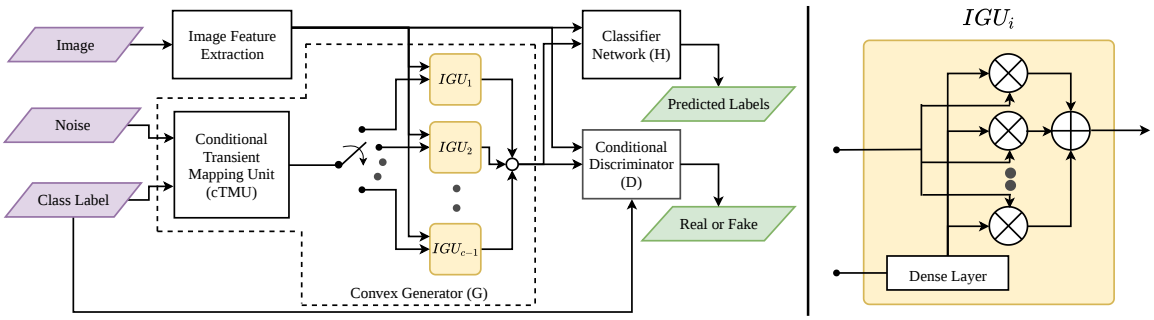


Figure 5.3: The GAMO model: (Left) Schematic of the GAMO framework; (Right) Illustration of an Instance Generation Unit (IGU). Given an image, the extracted feature vectors (either by a convolutional neural network F or by flattening) are fed to the classifier network H as well as the conditional discriminator D. H predicts the class label for the input data point while D distinguishes between real and fake data instances. The convex generator network G is composed of a $cTMU$, and IGUs corresponding to each of the $c-1$ minority classes. The IGU_i network takes an intermediate vector generated by $cTMU$ and maps it to a set of N_i convex weights. It then takes the set X_i as input and generates a new sample for the i-th class, as the convex combination of all the $\mathbf{x}_j \in X_i$.

Algorithm 4 Generative Adversarial Minority Oversampling (GAMO)

Input: X: training set, λ_l: latent dimension, λ_1: minibatch size, λ_2, λ_3: (hyperparameters, set to $\lceil \frac{N}{\lambda_b} \rceil$ in our implementation).
Output: A trained classification network H.
Note: For flattened images F need not be trained, i.e., $F(X)$ can be replaced by X.

while not converged **do**
  for λ_2 steps **do**
    Sample $\Lambda_d = \{\mathbf{x}_1, \mathbf{x}_2, \cdots \mathbf{x}_{\lambda_1}\}$ from X, with corresponding set of class labels Y_d.
    Update F by gradient descent on $(H(F(\Lambda_d)), Y_d)$ keeping H fixed.
  end for
  for λ_3 steps **do**
    Sample $\Lambda_d = \{\mathbf{x}_1, \mathbf{x}_2, \cdots \mathbf{x}_{\lambda_1}\}$ from X, with corresponding set of class labels Y_d.
    Sample $\Lambda_n = \{\mathbf{z}_1, \mathbf{z}_2, \cdots \mathbf{z}_{\lambda_1}\}$ from a l-dimensional standard normal distribution.
    Update H and D by respective gradient descent on $(H(F(\Lambda_d)), Y_d)$ and $(D(F(\Lambda_d)|Y_d), \mathbf{1})$, keeping F fixed.
    Generate set of labels Y_n by assigning each $\mathbf{z}_j \in \Lambda_n$ to one of the $c-1$ minority classes, with probability $\propto (P_c - P_i)$; $\forall i \in C \setminus \{c\}$.
    Update H and D by respective gradient descent on $(H(G(\Lambda_n|Y_n)), Y_n)$ and $(D(G(\Lambda_n|Y_n)|Y_n), \mathbf{0})$, keeping G fixed.
    Sample $\Lambda_g = \{\mathbf{z}_1, \mathbf{z}_2, \cdots \mathbf{z}_{\lambda_1}\}$ from a λ_l-dimensional standard normal distribution.
    Generate set of labels Y_g by assigning each $\mathbf{z}_j \in \Lambda_g$ to any of the $c-1$ minority classes with equal probability. Take ones' complement of Y_g as $\overline{Y}_g$.
    Update G by gradient descent on $(H(G(\Lambda_g|Y_g)), \overline{Y}_g)$ keeping H fixed.
    Update G by gradient descent on $(D(G(\Lambda_g|Y_g)|Y_g), \mathbf{1})$ keeping D fixed.
  end for
end while

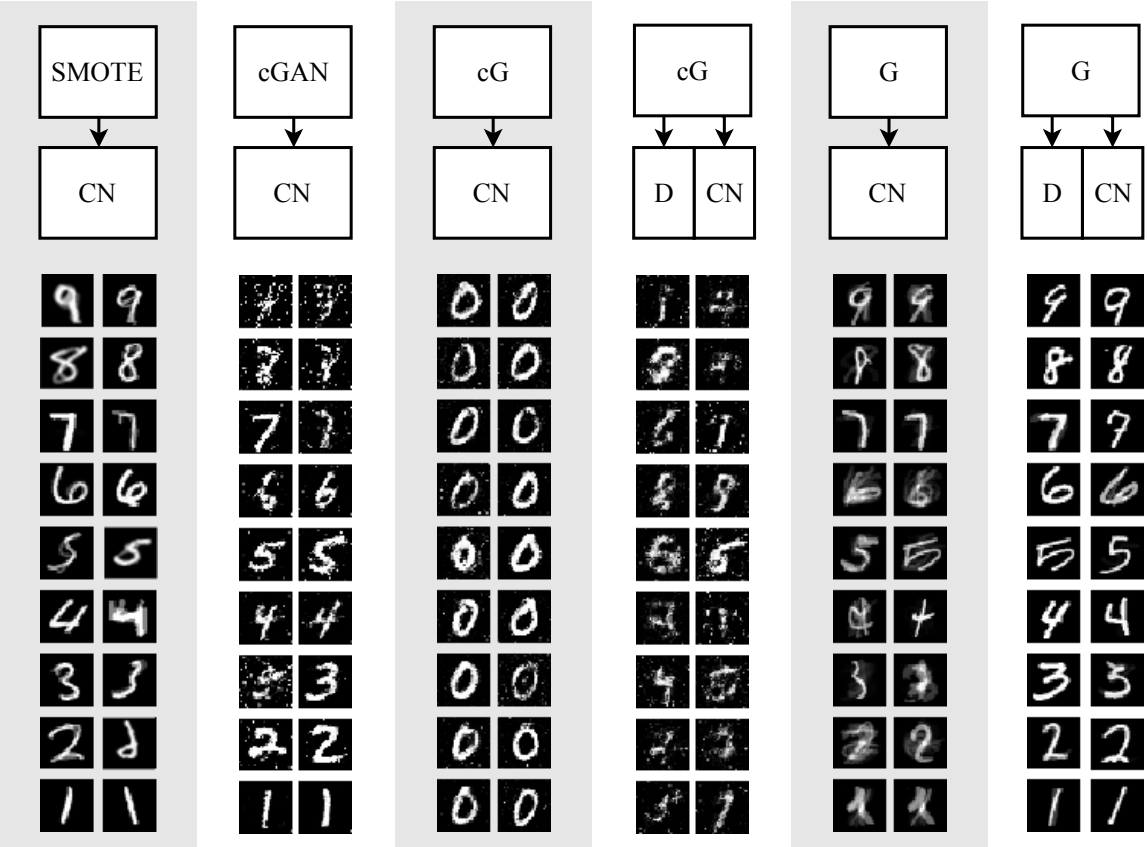


Figure 5.4: Ablation study on the class imbalanced MNIST dataset: SMOTE generates artificial samples from the minority class(es) as convex combinations of pairs of neighbors from the respective class(es). The oversampled dataset is then classified using a classifier network CN. SMOTE sometimes generates unrealistic "out-of-distribution" samples which are combinations of visually disparate images that happen to be Euclidean neighbors in the flattened image space. Using cGAN for generating new samples results in realistic images only from the more abundant minority classes. Training only a conditional Generator cG adversarially against CN, to generate images which will be misclassified by CN, results in new samples which all resemble the majority class '0'. Introducing a discriminator D (to ensure that cG adheres to class distributions) into the mix results in new samples which are somewhat in keeping with the class identities, but still unrealistic in appearance. Employing our proposed convex generator G to generate new samples by training it adversarially with CN (the GAMO\D formulation) results in samples which are in keeping with the class identities, but often "out-of-distribution" as the classes are non-convex. Finally, introducing D into this framework results in the complete GAMO model which can generate realistic samples which are also in keeping with the class identities.

been repeated 10 times to mitigate any bias generated due to randomization and the means and standard deviations of the index values are reported.

We have used a collection of 7 image datasets for our experiments, namely MNIST (LeCun

Table 5.1: Comparison of classification performance of CE and LS variants of classifiers on class imbalanced MNIST and Fashion-MNIST datasets.

Dataset	Algorithm	CE		LS	
		ACSA	GMean	ACSA	GMean
MNIST	Baseline CN	0.88±0.01	0.87±0.02	0.88±0.01	0.86±0.01
	SMOTE+CN	0.88±0.02	0.87±0.03	0.89±0.01	0.89±0.01
	cGAN+CN	0.88±0.01	0.87±0.01	0.89±0.01	0.88±0.01
	cG+CN	0.86±0.03	0.85±0.02	0.86±0.03	0.85±0.03
	cG+D+CN	0.85±0.02	0.83±0.01	0.85±0.02	0.82±0.02
	GAMO\D (Ours)	0.87±0.01	0.86±0.01	0.88±0.01	0.87±0.01
	GAMO (Ours)	0.89±0.01	0.88±0.01	**0.91±0.01**	**0.90±0.01**
Fashion-MNIST	Baseline CN	0.82±0.01	0.80±0.01	0.81±0.01	0.79±0.01
	SMOTE+CN	0.82±0.01	0.80±0.02	0.80±0.01	0.77±0.01
	Oversample+CN	0.81±0.01	0.79±0.01	0.81±0.01	0.79±0.01
	Augment+CN	0.82±0.01	0.78±0.01	0.82±0.01	0.78±0.01
	DOS	0.82±0.01	0.79±0.01	0.81±0.01	0.79±0.02
	cDCGAN+CN	0.81±0.02	0.78±0.01	0.82±0.01	0.80±0.01
	cG+CN	0.79±0.02	0.77±0.02	0.80±0.01	0.77±0.02
	cG+D+CN	0.79±0.02	0.78±0.01	0.79±0.01	0.78±0.02
	GAMO\D (Ours)	0.81±0.01	0.80±0.01	0.82±0.01	0.80±0.01
	GAMO (Ours)	0.82±0.01	0.80±0.01	**0.83±0.01**	**0.81±0.01**

The best result is boldfaced.

et al., 1998), Fashion-MNIST (Xiao et al., 2017), CIFAR10 (Krizhevsky, 2009), SVHN (Netzer et al., 2011), LSUN (Yu et al., 2015) and SUN397 (Xiao et al., 2010)[4]. All the chosen datasets except SUN397 are not significantly imbalanced in nature, therefore we have created their imbalanced variants by randomly selecting a disparate number of samples from the different classes[5]. Further, for all the datasets except SUN397, 100 points are selected from each class to form the test set. In the case of SUN397 (50 classes of which are used for our experiments) 20 points from each class are kept aside for testing.

We refrain from using pre-trained networks for our experiments as the pre-learned weights may not reflect the imbalance between the classes. We, instead, train the models from scratch to emulate real-world situations where the data is imbalanced and there is no pre-trained network available that can be used as an appropriate starting point. We have obtained the optimal architectures and hyperparameters for each contending method in Section 5.3-5.4 using a grid search (see in Section D.2 of Appendix D).

[4]Additional study on class imbalanced non-image benchmark datasets can be found in the Section D.3 in Appendix D. These results are omitted from the current chapter to preserve the focus on image classification as GAMO is primarily designed to address that problem.

[5]Additional notes on data creation can be found in Section D.1 in Appendix D

Table 5.2: Comparison of classification performance on class imbalanced CIFAR10 and SVHN datasets.

Algorithm	CIFAR10		SVHN	
	ACSA	GMean	ACSA	GMean
Baseline CN	0.45±0.01	0.37±0.01	0.74±0.01	0.73±0.01
SMOTE+CN	0.46±0.02	0.4±0.02	0.75±0.01	0.73±0.02
Oversample+CN	0.44±0.02	0.37±0.03	0.74±0.02	0.73±0.02
Augment+CN	0.47±0.01	0.39±0.02	0.69±0.01	0.63±0.01
cDCGAN+CN	0.42±0.02	0.32±0.03	0.69±0.01	0.66±0.02
DOS	0.46±0.02	0.37±0.01	0.71±0.02	0.68±0.01
GAMO\D (Ours)	0.47±0.01	0.40±0.01	0.75±0.01	0.75±0.02
GAMO (Ours)	**0.49±0.01**	**0.43±0.02**	**0.76±0.01**	**0.75±0.02**

The best result is boldfaced.

5.3.1 Class imbalanced MNIST and Fashion-MNIST

The experiments in this section are conducted using imbalanced subsets of the MNIST and Fashion-MNIST datasets. In the case of both the datasets, we have sampled 4000, 2000, 1000, 750, 500, 350, 200, 100, 60, and 40 points from classes in order of their index. Thus, the datasets have an IR of 100. We begin by establishing the effectiveness of our proposed framework. We also compare between the two variants of GAMO which use Cross Entropy (CE) and Least Square (LS) losses, respectively.

We undertake an ablation study on MNIST using flattened images to facilitate straightforward visualization of the oversampled instances. Convolutional features are used for Fashion-MNIST. For MNIST, we have compared GAMO, against baseline classifier network (CN), SMOTE+CN (training set is oversampled by SMOTE), cGAN+CN (training set oversampled using cGAN, which is then used to train CN), and also traced the evolution of the philosophy behind GAMO, through cG+CN (conditional generator cG adversarially trained against CN, in contrast to cGAN+CN where CN does not play any part in training cGAN), cG+D+CN (cG+CN network coupled with a discriminator D), and GAMO\D (GAMO without a discriminator) on the MNIST dataset. For Fashion-MNIST, SMOTE+CN is performed in the feature space learned by baseline CN. Oversample+CN (minority class images randomly sampled with replacement), Augment+CN (data augmentation to create new images and balance the training set), and DOS are also considered during comparison, while cGAN+CN is replaced by cDCGAN+CN (oversampled using conditional deep convolutional GAN).

The ablation study is shown visually in Figure 5.4 and the results for both datasets are tabulated in Table 5.1. Overall, GAMO is observed to perform better than all other methods

on both datasets. Interestingly, GAMO\D performs much worse than GAMO on MNIST but improves significantly on Fashion-MNIST. This may be due to the fact that the convolutional feature extraction for Fashion-MNIST results in distributed representations where the classes are almost convex with little overlap between classes, enabling the convex generator to always generate data points that reside inside the class distributions.

Since we observe from Table 5.1 that the LS variants of the classifiers mostly perform better than their CE based counterparts (which according to Mao et al. (2017) is contributed by the more stable and better decision boundary learned in LS), all the experiments in the subsequent sections are reported using the LS formulation for all the contending algorithms.

5.3.2 Class imbalanced CIFAR10 and SVHN

In the case of CIFAR10 and SVHN, the classes are subsampled (4500, 2000, 1000, 800, 600, 500, 400, 250, 150, and 80 points are selected in order of the class labels) to achieve an IR of 56.25. From Table 5.2 we can see that GAMO performs better than others on both of these datasets, closely followed by GAMO\D, further confirming the additional advantage of convolutional feature extraction in the GAMO framework. Interestingly, Augment+CN performs much worse than the other methods on the SVHN dataset. This may be due to the nature of the images in the SVHN dataset, which may contain multiple digits. In such cases, attempting to augment the images may result in a shift of focus from one digit to its adjacent digit, giving rise to a discrepancy with the class labels.

Table 5.3: Comparison of classification performance with increased number of training instances on class imbalanced CelebA and LSUN datasets.

Algorithm	CelebA-Small				CelebA-Large			
	During Training		During Testing		During Training		During Testing	
	ACSA	GMean	ACSA	GMean	ACSA	GMean	ACSA	GMean
Baseline CN	0.91±0.01	0.91±0.01	0.59±0.01	0.45±0.04	0.93±0.01	0.92±0.01	0.71±0.01	0.60±0.03
SMOTE+CN	0.99±0.01	0.99±0.01	0.62±0.02	0.48±0.03	0.99±0.01	0.99±0.01	0.71±0.02	0.66±0.03
Oversample+CN	0.99±0.01	0.99±0.01	0.59±0.03	0.39±0.04	0.98±0.01	0.98±0.02	0.68±0.02	0.62±0.02
Augment+CN	0.74±0.06	0.70±0.09	0.62±0.05	0.47±0.08	0.82±0.01	0.79±0.01	0.72±0.01	0.66±0.02
cDCGAN+CN	0.86±0.01	0.84±0.01	0.59±0.01	0.36±0.02	0.87±0.01	0.86±0.01	0.67±0.01	0.58±0.02
DOS	0.82±0.03	0.80±0.02	0.61±0.01	0.48±0.02	0.84±0.01	0.83±0.02	0.72±0.01	0.64±0.02
GAMO (Ours)	0.92±0.01	0.91±0.01	**0.66±0.01**	**0.54±0.02**	0.91±0.01	0.91±0.01	**0.75±0.01**	**0.70±0.02**
	LSUN-Small				LSUN-Large			
	During Training		During Testing		During Training		During Testing	
	ACSA	GMean	ACSA	GMean	ACSA	GMean	ACSA	GMean
Baseline CN	0.90±0.01	0.89±0.01	0.50±0.01	0.28±0.05	0.87±0.01	0.87±0.01	0.61±0.02	0.54±0.03
SMOTE+CN	0.99±0.02	0.99±0.02	0.50±0.01	0.40±0.02	0.98±0.01	0.98±0.01	0.66±0.03	0.62±0.03
Oversample+CN	0.99±0.01	0.99±0.01	0.52±0.01	0.43±0.02	0.98±0.01	0.98±0.01	0.62±0.03	0.58±0.03
Augment+CN	0.67±0.06	0.64±0.09	0.54±0.03	0.45±0.07	0.70±0.03	0.65±0.03	0.64±0.02	0.58±0.03
cDCGAN+CN	0.80±0.02	0.79±0.02	0.53±0.02	0.43±0.03	0.81±0.02	0.80±0.02	0.60±0.02	0.53±0.03
DOS	0.78±0.03	0.76±0.02	0.54±0.02	0.44±0.02	0.79±0.02	0.77±0.02	0.63±0.02	0.61±0.03
GAMO (Ours)	0.93±0.01	0.93±0.01	**0.57±0.01**	**0.50±0.02**	0.80±0.01	0.80±0.01	**0.70±0.02**	**0.68±0.03**

The best result is boldfaced.

5.3.3 Class imbalanced CelebA and LSUN

The experiments on CelebA and LSUN are undertaken to evaluate the performance of GAMO on images of higher resolution, as well as to assess the effects of an increase in the number of instances from the different classes. In the case of CelebA, the images are scaled to 64×64 size, while for LSUN the same is done on a central patch of resolution 224×224 extracted from each image. In the case of CelebA, we have created two 5 class datasets by selecting samples from non-overlapping classes of hair colors, namely *blonde, black, bald, brown*, and *gray*. The first dataset is the smaller one (having 15000, 1500, 750, 300, and 150 points in the respective classes) with an IR of 100, while the second one is larger (having 28000, 4000, 3000, 1500, and 750 points in the respective classes) with an IR of 37.33. Similarly, in the case of LSUN, we select 5 classes namely *classroom, church outdoor, conference room, dining room*, and *tower*, and two datasets are created. The smaller one (with 15000, 1500, 750, 300, and 150 points from the respective classes) has an IR of 100, while the larger one (with 50000, 5000, 3000, 1500, and 750 points) has an IR of 66.67.

In Table 5.3, we present the ACSA and GM over both the training and test sets for the small and large variants of the two datasets. We can observe that all the algorithms manage to close the gap between their respective training and testing performances as the size of the dataset increases. However, SMOTE+CN shows a high tendency to overfit, which might be caused by the miscalibrated initial baseline CN. The same is observed for Oversample+CN, indicating that such a balancing technique may not provide additional information to a classifier to facilitate better learning. Moreover, while Augment+CN seems to have the lowest tendency to overfit (the smallest difference between training and testing performances), GAMO exhibits a greater ability to retain good performance on the test dataset.

5.3.4 Subset of SUN397

We have randomly selected 50 classes from SUN397 to construct a dataset containing 64×64 sized images (depending on the image size either a 512×512 or a 224×224 center patch is extracted, which is then scaled to 64×64) with an IR of 14.21. The experiment on SUN397 is performed to evaluate the performance of GAMO over a large number of classes. A scrutiny of the result tabulated in Table 5.4 reveals that despite all four contending techniques being

severely affected by the complexity of the classes and the scarcity of data samples from many of the classes, GAMO is able to retain overall better performance than its competitors.

Table 5.4: Comparison of classification performance on subset of SUN397.

Algorithm	ACSA	GMean
Baseline CN	0.26±0.04	0.19±0.05
SMOTE+CN	0.28±0.04	0.21±0.04
Oversample+CN	0.23±0.05	0.00±0.00
Augment+CN	0.30±0.04	0.21±0.04
cDCGAN+CN	0.20±0.05	0.00±0.00
DOS	0.28±0.04	0.20±0.05
GAMO (Ours)	**0.32±0.04**	**0.24±0.03**

The best result is boldfaced.

5.4 GAMO2pix

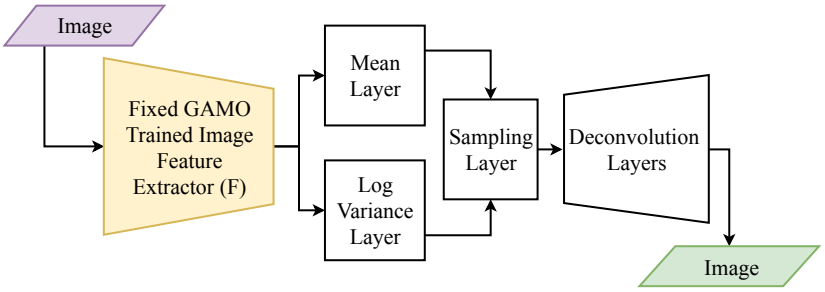
GAMO results ultimately in a classifier trained to properly classify samples from all the classes. However, some applications may require that actual samples be generated by over-sampling to form an artificially balanced dataset. While GAMO directly generates images if flattened images are used, it only generates vectors in the distributed representation space (mapped by the convolutional layers) for the convolutional variant. Therefore, we also propose the GAMO2pix mechanism to obtain images from the GAMO-generated vectors in the distributed representation space.

Table 5.5: Comparison of FID of cDCGAN and GAMO2pix.

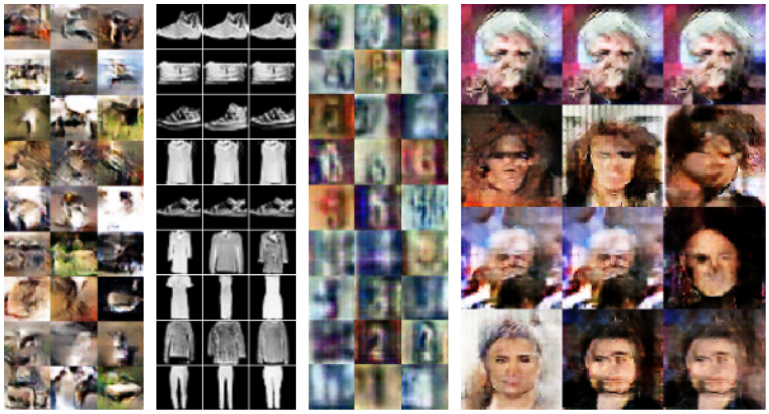
Dataset	GAMO2pix (Ours)	cDCGAN
Fashion-MNIST	**0.75±0.03**	5.57±0.03
SVHN	**0.17±0.02**	0.59±0.04
CIFAR10	**1.59±0.03**	2.96±0.03
CelebA-Small	**11.13±0.04**	15.12±0.05

The best result is boldfaced.

Our network for generating images (as illustrated in Figure 5.5a) from the GAMO-generated vectors is inspired by the Variational Autoencoder (VAE) (Kingma and Welling, 2013; Rezende et al., 2014). VAE, unlike regular autoencoders, is a generative model that attempts to map the encoder output to a standard normal distribution in the latent space, while the decoder is trained to map samples from the latent normal distribution to images. In GAMO2pix, the convolutional feature extractor F trained by GAMO is kept fixed and



(a)



(b)

(c)

Figure 5.5: GAMO2pix model and its performance. (a) GAMO2pix network. (b)-(c) Comparison of images respectively generated by cDCGAN, and GAMO2pix for (left to right) CIFAR10, Fashion-MNIST, SVHN, and CelebA-Small.

connected to two trainable parallel dense layers, which learn the mean ($\boldsymbol{\mu}$) and the log-variance ($\log \boldsymbol{\sigma}^2$) of the posterior distribution. Then samples drawn from $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma})$ are fed to the decoder. The loss of GAMO2pix is the sum of $KL(\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma})||\mathcal{N}(\mathbf{0}, \mathbf{I}))$ and mean squared reconstruction error. The GAMO2pix network is trained separately for each class to learn the

inverse map from the feature space induced by F to the original image space. Thus, we use GAMO2pix to generate realistic images of the concerned class given the GAMO-generated vectors.

We present the images respectively generated by cDCGAN and GAMO2pix on CIFAR10, Fashion-MNIST, SVHN and CelebA-Small in Figures 5.5b-5.5c. We can see that GAMO2pix can indeed generate more realistic and diverse images, compared to cDCGAN which also suffers from mode collapse for minority classes. This is further confirmed by the lower Fréchet Inception Distance (FID) (Heusel et al., 2017) (calculated between real and artificial images from each class and averaged over classes) achieved by GAMO2pix, as shown in Table 5.5.

5.5 Discussion

In this chapter, we presented GAMO, a three-player game between a convex generator, a classifier network, and a discriminator, which results in an effective end-to-end oversampling technique for handling class imbalance in deep learning frameworks. GAMO can be considered as an important step towards training robust discriminative models using adversarial learning. We have observed from our experiments that the convolutional variant of GAMO is more effective due to the distributed representations learned by the convolutional layers. We also found that the LS loss variant of GAMO generally performs better than the CE loss variant. Further, in this chapter, we introduced GAMO2pix a VAE inspired network that can generate realistic images from GAMO generated samples. The images generated by GAMO2pix also attest to the fact that GAMO can indeed provide a diverse set of quality samples while adhering to the class distribution.

Chapter 6

Conclusion

Summary

In this chapter, we focus on evaluating the various contributions made by us in the previous chapters of this thesis. Specifically, we provide a brief summary highlighting the key attributes of our works and discuss their importance in further enriching the active research efforts focusing on the topic of handling class imbalance. Moreover, we detail the future possibilities of our contribution hoping they will encourage further research opportunities. Finally, as a concluding remark, we list the various open problems related to class imbalance.

6.1 Evaluation of contributions

In this section, we briefly evaluate the different contributions made in this thesis. We start with the introductory Chapter 1 which defines the problem of class imbalance and empirically characterizes its effects on the performance of a classifier. The chapter also discusses the plethora of research done by machine learning as well as deep learning communities over the past couple of decades in an attempt to improve the reliance of a classifier in presence of class imbalance. The review highlights the key data level, algorithm level, and hybrid level solutions and describes the multitude of challenges associates with each direction. The introductory chapter concludes by presenting a road map of this thesis and briefly describing the contributions made by the rest of the chapters.

Before we proceed to design classifiers tailored for handling class imbalance we take a look at the applicability of different performance evaluation indices. In Chapter 2 we start by discussing the need for special indices that will not be biased in the presence of class im-

balance and thus will able to offer a fair evaluation. A detailed survey of previous researches done on the topic highlights the requirement of a set of generalized conditions that an index should satisfy to be applicable to a wide range of applications. Thus, we proceed to define two necessary conditions that ensure the invariance of an index with altering the extent of class imbalance and changing the number of classes between training and test/validation set. The invariance of four two-class and five multi-class performance evaluation indices are theoretically validated and remedial modification and/or normalization are proposed as remedial measures. Further, a third condition is proposed to assess the interpretability of an index which satisfies the necessary invariance constraints. A simulation study using class imbalanced subsets of ImageNet and state-of-the-art classifiers tailed to efficiently tackle class imbalance is presented as empirical evidences for the theoretical findings. Finally, under the light of the three conditions a detailed discussion provides an application-specific recommendation for different indices. The key contribution of this chapter is moving away from the traditional often empirical application-specific analysis to a more generalized theoretical discussion through identifying some necessary invariance conditions. Moreover, the chapter focus on validating a set of widely used performance evaluation indices, propose rectification as per requirement and highlight their individual applicability.

We then in Chapter 3 consider the intuitively better FkNN classifier which also offers a performance improvement over canonical kNN. We start by highlighting through a motivational example that the theoretical performance analysis of kNN and FkNN should not follow the same route as the latter instead of labeling returns class memberships for a given test point. Thus, unlike the commonly used misclassification based theoretical study we focus on the convergence of the class membership estimator used in FkNN. Such a convergence can be expressed in terms of bias and MSE of the estimator. We specifically show that with an increasing number of training examples the expected bias and MSE of the estimator converges towards their corresponding minimum at zero. A couple of key advantages of our analysis is its reliance on a set of elementary constraints as well as not being applicable for a particular choice of parameter k. The simple yet effective analysis is also directly extendable from binary to multi-class classification problems. A simulation study on artificial as well as real-world datasets validates the theoretical findings. This analysis also helps us to better understand the behavior of a FkNN classifier in the presence of class imbalance. Specifi-

cally, an imbalanced training set will always contain a fewer number of examples compared to its balanced counterpart and thus likely to result in poorer convergence. To improve the immunity of kNN classifier in presence of class imbalance we employ a simple heuristic locally adaptive weighting scheme which will consider the neighborhood properties of a test point while calculating a class-specific weight for it. The efficacy of the weighting scheme is validated through experimental comparison with FkNN variants tailored for handling class imbalance on a set of imbalanced real-world benchmark datasets having diverse properties.

We retain our focus on FkNN a simple, non-parametric, widely used classifier offering commendable performance in Chapter 4 as well. We start by demonstrating the need for optimizing a class-specific set of feature weights which if considered during distance calculation will be able to clearly distinguish the members of the class under concern from the rest. However, mathematically optimizing such a set of class-specific feature weight may not be easy as finding an objective function satisfying the conditions set by the optimizer can be difficult in practice. Thus, we propose to use a self-adaptive variant of DE which also demonstrates an improved performance over the canonical optimizer while incurring a similar computational cost. We also highlight how finding a good performing global value of the parameter k can be a tiresome process. We show that optimizing the class-specific feature weights and the choice of global k may be reduced to a single problem and thus can be solved simultaneously in the proposed PIFWkNN. Even though class-specific feature weights are likely to offer some immunity against class imbalance additional measures may still be required considering the general scarcity of minority instances. Thus, in PIFW^2kNN we attempt to use a set of class-specific weights to compensate for the class imbalance. We remove the need for expensive cost tuning by simultaneously optimizing the class-specific weights by incorporating them in the optimization problem of PIFWkNN. We show how this can be achieved through a modified encoding along with a different objective function more suitable for accounting the effects of class imbalance. A comparison with state-of-the-art classifiers employing feature weighting establishes the effectiveness of PIFWkNN. The efficacy of PIFW^2kNN is also experimentally validated through a comparison with kNN and FkNN variants capable to sustain their performance in presence of class imbalance on the real-world benchmark imbalanced datasets.

Finally in Chapter 5 we focus on end-to-end deep image classifiers and proceed to propose

an adaptive oversampling technique. The GAMO framework consists of three component networks, namely a convex generator which generates a new sample as a convex combination of all the examples from a given class, a classifier, and a discriminator that guides the generated samples to adhere the corresponding class distribution. The convex generator maintains an adversarial relationship with the classifier such that it can adaptively generate hard to classify instances during oversampling. The convex generator is also adversarially connected with the discriminator following a typical GAN strategy. In essence, GAMO plays a three-player game between the convex generator, the classifier, and the discriminator. Unlike previous approaches (Ando and Huang, 2017) GAMO is capable of not only adaptively generating artificial samples in the distributed feature space which aids the learning of the classifier but also realistic image instances through the proposed GAMO2pix network. Further, the convex generator also attempts to address the issues of mode collapse and boundary distortion of regular GANs trained on a limited amount of data.

To summarize, this thesis starts by attempting to answer the primary question of evaluating the efficacy of a classifier in the presence of class imbalance. Following, it focuses on FkNN by first validating its convergence and explaining its susceptibility to class imbalance in the process. The thesis also aims to offer immunity to FkNN against class imbalance through a locally adaptive class weighting. Subsequently, this thesis proposes two variants of FkNN namely PIFWkNN and PIFW^2kNN which respectively improves upon the base classifier and offers additional resilience against class imbalance. Finally, GAMO aims to address the issues associated with a GAN based adaptive oversampling strategy which is applicable to an end-to-end deep image classifier.

6.2 Future Possibilities

In this section, we describe how our contributions made in this thesis can be further explored to open up new research opportunities.

- In Chapter 2 we discussed how different performance indices may suffer distortions with altering class imbalance and the number of classes between training and test/validation set in a class imbalanced problem. Following a theoretical analysis and simulation study, we have also recommended an application-specific applicability of different indices. A

natural future extension of this work would be to investigate the behavior of indices that are used in imbalanced multi-label (Madjarov et al., 2012) and multi-instance (Carbonneau et al., 2018) classification problems. One may also consider validating the efficacy of the modified/normalized indices on class imbalanced problems where the two types of distortions are naturally occurring. For example, Type 1 distortion is quite inherent in image foreground and background classification (Lin et al., 2017a). This is because the foreground usually spreads over less number of pixels compared to background resulting in class imbalance. Moreover, the fraction of background to foreground in an image i.e. the RRT may significantly vary between images consequently causing distortion in performance indices which fail to satisfy Condition 2.1. Type 1 distortion is also possible during sentiment analysis from tweets, where not all sentiments may occur with equal frequency (Zimbra et al., 2018), while the prior probability of different sentiments may notably change over time. Thus, if a sentiment analyzing classifier is periodically tested after deployment for potential fine-tuning and an index susceptible to Type 1 distortion is utilized for quantifying its performance, then the result may mislead the quality assessment. On the other hand, Type 2 distortion can occur in open set recognition or incremental learning problems where the number of classes may increase over time (Rudd et al., 2018), thus the use of an index which satisfies Condition 2.2 may be beneficial.

- In Chapter 3 we discussed on the theoretical convergence of the FkNN class membership estimator and employed a heuristic locally adaptive class weighting scheme to guard against the effects of class imbalance. A limitation of this work is the assumptions on the properties of the class distributions and the membership functions. Though most of the regular functions do follow the desirable properties, there are notable exceptions as well. One can consider further reducing the restrictions imposed by the assumptions in a future direction of this work. The impact of different distance measures other than Euclidean, such as geometric divergence (Saha and Das, 2016) on the performance of FkNN can be also worth investigating. Another interesting extension of this work may come from analysing the behavior of possibilistic kNN (PkNN) (Frigui and Gader, 2009) where given a test point, its sum of membership to all the classes need not be

equals to 1. Given our convergence analysis of the class membership estimator does not impose any constraints on the sum of memberships, it may very well be extended to the case of PkNN. However, as the membership estimator of PkNN changes from that of FkNN, the applicability of Result 3.1 may not be straightforward anymore. Thus, the problem may be reduced to finding a suitable bound on the class membership estimator of PKNN or expressing the same in some approximated form such that Result 3.1 can be applied. In another direction, it would also be interesting to use a technique similar to Bax (2012) for establishing a bound on the generalization error of FkNN. Further, the simple heuristic point specific class weighting can be improved by optimizing it in a manner similar to Anava and Levy (2016).

- In Chapter 4 we demonstrated the importance of an optimized set of class-specific feature weighting and global choice of parameter k in FkNN. We further highlighted how a set of class-specific global weights can also be simultaneously optimized which can improve the immunity of FkNN in the presence of class imbalance. A direct extension of this work can investigate the usefulness of different DE variants in solving the optimization problem involved in the process. With an increasing number of classes and features, the optimization problem can also become high dimensional which may require special purpose self-adaptive scalable DE based optimizer (Ghosh et al., 2017). Moreover, both of PIFWkNN and PIFW^2kNN are reliant on optimizing global weights and parameter setting. This may further be improved for complex overlap structures and small disjuncts by incorporating locality specific information through neighborhood dependent choices of weights and parameter values.

- In Chapter 5 we introduced a novel GAN inspired oversampling technique applicable to deep image classifiers. An interesting area of future investigation is to improve the quality of the images generated by GAMO2pix by employing a different architecture such as BEGAN (Berthelot et al., 2017). To reduce the tendency of GAMO to overfit as well as to potentially improve its performance, one may consider hybridization with improved GAN variants (Gurumurthy et al., 2017) which can achieve good performance even with less number of training samples. Further, one may explore the efficacy of GAMO to learn new classes by taking inspiration from Memory Replay GAN (Wu et al.,

2018), or study the usefulness of the proposed convex generator for handling boundary distortion in GANs.

6.3 Open problems

In this section, we discuss some open problems related to class imbalance which are likely to introduce new avenues of research.

- We start with the very basic question of defining a classification problem as class imbalanced. Theoretically, a training set can be considered as class imbalanced if the number of representatives from the different classes are not equal. However, in practice, it is almost impossible to obtain a training set that contains an equal number of points from all the classes. Further in Chapter 1 we demonstrated that even if there is an unequal number of class representatives the effect of class imbalance on classification performance largely depends on the individual class structure and the choice of the classifier. Thus, the question remains in which case we will expect a classifier to be significantly affected by class imbalance and will require additional efforts to improve the accuracy. Answer to such a trivial question may be provided by a detailed theoretical study, which will consider the class distribution as well as the nature of the classifier to predict the impact of class imbalance on the classification performance over the minority class.

- Another basic question is involved with the quantification of class imbalance through IR in a multi-class classification problem. In such cases, IR only represents the fraction of points belonging to the largest class and the smallest class ignoring the rest. However, this may not be able to properly express the extent of imbalance in the training set. As an example, we take a three-class classification problem where the classes respectively contain 1000, 1000, 10 points. Now let us consider another training set having 1000, 10, and 10 data instances in the three respective classes. In both cases, the IR is 100 indicating a similar class imbalance. However, in reality, the second training set has two minority classes which are likely to further hinder fair training. Thus, a more expressive form of IR is required which will better represent the learning difficulty.

- In traditional machine learning, the widely used undersampling approach is prone to

suffer from information loss. However, to mitigate such loss techniques should focus on finding better representatives through which the entire class can be properly summarized. A critical obstacle towards achieving this goal comes in the form of disjuncts (Weiss and Hirsh, 2000; Weiss, 2010) which are intra-class clusters usually corresponding to different sub-concepts. Evidently, each of such disjuncts needs to be adequately represented during undersampling the majority class. However, usually not all disjuncts are uniformly represented in the training set as the corresponding sub-concepts are not observable with equal probability (Gao et al., 2019). This results in an intra-class imbalance between different disjuncts usually making the smaller ones harder to classify and thus difficult to undersample. Further, undersampling techniques should also be aware of the locality specific diverse requirement of information. For example, in a spherical class structure, it is acceptable to have less number of training instances in the center while more information is required to correctly identify the class periphery. The problem is even more critical in deep learning systems where the feature extraction is performed simultaneously with classification. Further, deep learners have a large number of parameters proper tuning of which requires a sufficient amount of diverse data.

- We have discussed in Chapter 1 how the commonly used oversampling techniques are required to be adaptive and the artificial samples should adhere to the class distribution. We have also noted how the challenge escalates in deep learning systems given the realistic nature of the data and the end-to-end approach of the learners. In classical machine learning, a plethora of solutions was proposed over the years. However, in deep learning only a handful of recent research works attempted to address these issues (Tripathi et al., 2019; Liu et al., 2019) while the majority followed the traditional cost-sensitive route leaving such a promising direction of research mostly unexplored.

- In cost-sensitive deep learning systems the costs are usually assigned by heuristic approaches or expensive parameter tuning (Lin et al., 2017a; Cui et al., 2019). However, some research works (Khan et al., 2018) attempted to include the cost in the objective function such that they can be properly learned. Not only this avenue of research should gain further attention but also new ways such as adversarially learning the costs

may be considered.

- With the advent of new problems such as few-shot learning (Wang et al., 2019), open set classification (Rudd et al., 2018), multi-instance and multi-label classification (Zhang and Zhang, 2007), weakly supervised learning (Zhou, 2018), fine-grained classification (Xiao et al., 2015), and emerging new data sources such as streaming data (Brzezinski et al., 2019) investigating the scope and effects of class imbalance need to adapt as well.

Appendix A

Supplementary for Chapter 2

A.1 Construction of datasets used in Example 2.1

A.1.1 Datasets used for illustration of Type 1 distortions

For the two class datasets, both of the classes are sampled from multivariate normal distribution $\mathcal{N}(\mu_i, \Sigma_i)$, where μ_i and Σ_i are respectively the mean and co-variance matrix of the i^{th} class, and $i \in \{1, 2\}$. The parameters are as follows:

$$\mu_1 = [3\ 3]^T \text{ and } \mu_2 = [7.5\ 3]^T.$$

$$\Sigma_1 = \begin{bmatrix} 0.45 & 0 \\ 0 & 0.45 \end{bmatrix} \text{ and } \begin{bmatrix} 0.25 & 0 \\ 0 & 0.25 \end{bmatrix}$$

We start with sampling 5000 data points from each class and then sub-sample from the one in the left to construct the datasets of gradually deteriorating behavior.

A.1.2 Datasets used for illustration of Type 2 distortions

Similar to the ones used to construct the datasets used to illustrate the effect of Type 1 distortion, here also the classes are sampled from $\mathcal{N}(\mu_i, \Sigma)$, where $i \in \{1, 2, 3, 4, 5, 6\}$. For the i^{th} class the center μ_i, lies on the i^{th} vertex of a regular hexagon centered at $[0\ 0]^T$, with

edge length of 5 unit. Therefore,

$$\mu_1 = [-2.5 \ -4.33]^T, \mu_2 = [-5 \ 0]^T,$$
$$\mu_3 = [-2.5 \ 4.33]^T, \mu_4 = [2.5 \ 4.33]^T,$$
$$\mu_5 = [5 \ 0]^T, \text{ and } \mu_6 = [2.5 \ -4.33]^T,$$

$$\text{while } \Sigma = \begin{bmatrix} 0.08 & 0 \\ 0 & 0.08 \end{bmatrix}.$$

We start with sampling from the three classes lying respectively on the three adjacent vertices of the regular hexagon. We gradually deteriorate the behavior of the dataset by sampling from the three remaining vertices in an anti-clockwise order. Further, from the i^{th} class we randomly sample n_i number of test points, where $n_1 = 5000$, $n_2 = 1500$, $n_3 = 4000$, $n_4 = 500$, $n_5 = 3500$, and $n_6 = 4500$.

A.2 Description of class imbalanced datasets sampled from ImageNet 2012

In Table A.1 we give the details of the 12 higher-level classes (alongside the leaf classes chosen for each of them) selected from ImageNet. The Table A.2 summarizes the properties (such as number of points, classes, IR, etc.) for each of the created datasets. Moreover, a dataset is named as "ImageNet-*c*-*i*", where c is the number of class and i is a serial index. For example, ImageNet-10-1 indicates the first 10-class dataset.

A.3 Parameter Setting of the classifiers used for empirical evaluation

For Dual-LexiBoost all the parameters are set following the guidelines of the original article. To elaborate, in Dual-LexiBoost the k-nearest neighbor is chosen as the base classifier where the number of neighbors is set to 3; while the number of rounds is set to 10. For NBSVM, a linear kernel (using radial basis kernel did not significantly improve performance, possibly because Inception V3 induces linear separability among classes) is used while the regulariza-

Table A.1: Selected Classes from ImageNet ILSVRC2012

Selected class (Serial*)	No. of Points	Leaf classes	ILSVRC2012 Reference
Animal (1)	6500	Egyptian Cat	n02124075
		Cougar	n02125311
		Gazelle	n02423022
		Great Dane	n02109047
		Zebra	n02391049
Artifact (2)	6394	Revolver	n04086273
		Desk	n03179701
		Chainsaw	n03000684
		Typewriter Keyboard	n04505470
		Teddy, Teddy Bear	n04399382
Dress (3)	2600	Crash Helmet	n03127747
		Gown	n03450230
Factory (4)	1300	Lumber-mill	n03697007
Food (5)	3900	Strawberry	n07745940
		Mashed Potato	n07711569
		Bagel	n07693725
Fungus (6)	3900	Hen-of-the-Woods	n13052670
		Earthstar	n13044778
		Stinkhorn	n13040303
Geological (7)	4200	Cliff	n09246464
		Valley	n09468604
		Coral Reef	n09256479
		Seashore	n09428293
Natural (8)	1300	Rapeseed	n11879895
Person (9)	2600	Groom	n10148035
		Scuba Diver	n10565667
Plant (10)	2600	Daisy	n11939491
		Yellow lady slippers	n12057211
Sport (11)	2600	Racket	n04039381
		Barbell	n02790996
Vehicle (12)	6500	Airliner	n02690373
		Gondola	n03447447
		Mountain Bike	n03792782
		Ambulance	n02701002
		Limousine	n03670208

∗ The serial number will be hereafter used to represent the corresponding class.

tion parameter is varied in the set $\{10, 100\}$. The number of iterations in RUSBoost is kept fixed at 10 while all the other parameters are set to the default as advised in the original article. The MLP is designed with one hidden layer containing $\sqrt{cd}$ number of hidden nodes, while the parameters for SMOTE are set following the corresponding research article.

Table A.2: Properties of the class imbalanced subsets of ImageNet.

Dataset name	c	Serial of classes	Data points in each class	IR
ImageNet-2-1	2	1, 9	3000, 75	40
ImageNet-2-2	2	12, 10		
ImageNet-2-3	2	2, 8		
ImageNet-2-4	2	7, 4	3000, 150	20
ImageNet-2-5	2	5, 11		
ImageNet-2-6	2	6, 3		
ImageNet-2-7	2	1, 8	3000, 300	10
ImageNet-2-8	2	12, 11		
ImageNet-2-9	2	6, 4		
ImageNet-2-10	2	7, 9	3000, 600	5
ImageNet-2-11	2	2, 10		
ImageNet-2-12	2	5, 3		
ImageNet-3-1	3	1, 2, 8	2500, 1250, 100	25
ImageNet-3-2	3	12, 10, 9		
ImageNet-3-3	3	7, 5, 3		
ImageNet-3-4	3	6, 11, 4		
ImageNet-5-1	5	1, 2, 4, 10, 5	3000, 1500, 750, 250, 100	30
ImageNet-5-2	5	7, 12, 9, 11, 3		
ImageNet-5-3	5	6, 2, 9, 10, 8		
ImageNet-10-1	10	1, 2, 5, 6, 7, 12, 10, 11, 4, 8	2000, 1750, 1500, 1250, 1000, 750, 500, 250, 150, 100	20
ImageNet-10-2	10	1, 2, 5, 6, 12, 10, 11, 4, 9, 3		
ImageNet-10-3	10	1, 2, 6, 12, 10, 4, 9, 3, 7, 8		

Appendix B

Supplementary for Chapter 3

B.1 Description of the artificial datasets

In this section we describe the protocol followed to generate the artificial datasets used in Chapter 3. In a c-class d-dimensional dataset $\mathbb{S}$, let us denote the set of S_j number of points which belong to class j by $\mathbb{S}_j$, for all $j \in C$. We denote an identity matrix of size $b \times b$ as I_b, while $\mathbf{x}$ is a d-dimensional real vector represented as $[x^{(1)}, x^{(2)}, \cdots, x^{(d)}]$. Let us also define

a set of three matrices, namely M_1, M_2, and M_3 as follows:

$$M_1 = \begin{bmatrix} -20 & -10 \\ -20 & 0 \\ -20 & 10 \\ -10 & -10 \\ -10 & 0 \\ -10 & 10 \\ 0 & -10 \\ 0 & 0 \\ 0 & 10 \\ 10 & -10 \\ 10 & 0 \\ 10 & 10 \\ 20 & -10 \\ 20 & 0 \\ 20 & 10 \end{bmatrix}, \quad M_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 5 \\ 0 & 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 0 & 5 & 0 & 0 \\ 0 & 0 & 5 & 0 & 5 \\ 0 & 0 & 5 & 5 & 0 \\ 0 & 0 & 5 & 5 & 5 \\ 0 & 5 & 0 & 0 & 0 \\ 0 & 5 & 0 & 0 & 5 \\ 0 & 5 & 0 & 5 & 0 \end{bmatrix},$$

$$M_3 = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 5 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 0 & 5 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 5 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 5 & 5 \\
0 & 0 & 0 & 0 & 0 & 0 & 5 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 5 & 0 & 0 & 5 \\
0 & 0 & 0 & 0 & 0 & 0 & 5 & 0 & 5 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 5 & 0 & 5 & 5 \\
0 & 0 & 0 & 0 & 0 & 0 & 5 & 5 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 5 & 5 & 0 & 5 \\
0 & 0 & 0 & 0 & 0 & 0 & 5 & 5 & 5 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 5 & 5 & 5 & 5 \\
0 & 0 & 0 & 0 & 0 & 5 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 5 & 0 & 0 & 0 & 5 \\
0 & 0 & 0 & 0 & 0 & 5 & 0 & 0 & 5 & 0 \\
0 & 0 & 0 & 0 & 0 & 5 & 0 & 0 & 5 & 5 \\
0 & 0 & 0 & 0 & 0 & 5 & 0 & 5 & 0 & 0
\end{bmatrix}.$$

With the abpve setting we now proceed to detail the generation of artificial dataset in the following Table B.1.

The mean of the classes for higher dimensional datasets are chosen to ensure that no two classes are completely overlapped in all the dimensions. Therefore, to find the mean of a class we first start by encoding the index of that class in binary numeral system and representing it as a vector. Then, we append necessary number of zeros to the left of the binary vector to reach the required dimension. Finally, we multiply each dimension of the binary vector with a constant scalar to obtain the mean for that class. For example, to calculate the mean for the 3-rd class in AD-5-5, we start by representing 3 as [1 1]. We then add 3 zeros to the left of the 2-dimensional binary vector to extend it to the 5-dimensional space forming [0 0 0 1 1].

Table B.1: Construction of artificial datasets.

Dataset	Description
AD-2-2-NO	$S_1 = 200000$ and $\mathbb{S}_1 \sim \mathcal{N}(\mu_1, \Sigma)$, where $\mu_1 = [-5\ 0]$, and $\Sigma = I_2$. $S_2 = 200000$ and $\mathbb{S}_2 \sim \mathcal{N}(\mu_2, \Sigma)$, where $\mu_2 = [5\ 0]$, and $\Sigma = I_2$.
AD-2-2-SO	$S_1 = 200000$ and $\mathbb{S}_1 \sim \mathcal{N}(\mu_1, \Sigma)$, where $\mu_1 = [-3.5\ 0]$, and $\Sigma = I_2$. $S_2 = 200000$ and $S_2 \sim \mathcal{N}(\mu_2, \Sigma)$, where $\mu_2 = [3.5\ 0]$ and $\Sigma = I_2$.
AD-2-2-LO	$S_1 = 200000$ and $\mathbb{S}_1 \sim \mathcal{N}(\mu_1, \Sigma)$, where $\mu_1 = [-2\ 0]$, and $\Sigma = I_2$. $S_2 = 200000$ and $\mathbb{S}_2 \sim \mathcal{N}(\mu_2, \Sigma)$, where $\mu_2 = [2\ 0]$, and $\Sigma = I_2$.
AD-9-2	$S_j = 200000$ and $\mathbb{S}_j \sim \mathcal{N}(\mu_j, \Sigma)$, where $\Sigma = I_2$, and μ_j is the $(j+3)$-rd row of the matrix M_1, for all $j = 1, 2, \cdots, 9$.
AD-15-2	$S_j = 200000$ and $\mathbb{S}_j \sim \mathcal{N}(\mu_j, \Sigma)$, where $\Sigma = I_2$, and μ_j is the j-th row of the matrix M_1, for all $j = 1, 2, \cdots, 15$.
AD-5-5	$S_j = 200000$ and $\mathbb{S}_j \sim \mathcal{N}(\mu_j, \Sigma)$, where $\Sigma = I_5$, and μ_j is the j-th row of the matrix M_2, for all $j = 1, 2, \cdots, 5$.
AD-10-5	$S_j = 200000$ and $\mathbb{S}_j \sim \mathcal{N}(\mu_j, \Sigma)$, where $\Sigma = I_5$, and μ_j is the j-th row of the matrix M_2, for all $j = 1, 2, \cdots, 10$.
AD-10-10	$S_j = 200000$ and $\mathbb{S}_j \sim \mathcal{N}(\mu_j, \Sigma)$, where $\Sigma = I_{10}$, and μ_j is the j-th row of the matrix M_3, for all $j = 1, 2, \cdots, 10$.
AD-20-10	$S_j = 200000$ and $\mathbb{S}_j \sim \mathcal{N}(\mu_j, \Sigma)$, where $\Sigma = I_{10}$, and μ_j is the j-th row of the matrix M_3, for all $j = 1, 2, \cdots, 20$.

Finally, we multiply 5 with each element of the binary vector to construct the third row of M_2.

B.2 Description of real-world datasets

We list the key properties of the 12 real world datasets used in the simulation in Table B.2.The same for the 37 class imbalanced datasets are detailed in Table B.3.

Table B.2: Key properties of the 12 real-world datasets used in the simulation study.

Dataset Name	Number of Points	Number of Dimensions	Number of Classes
Banana	5300	2	2
Diabetic	1151	19	2
Heart	270	13	2
Iris	150	4	3
Letter	20000	16	26
Magic*	13688	10	2
Penbased	10992	16	10
Ring	7400	20	2
Seeds	210	7	3
Segment	2310	19	7
Twonorm	7400	20	2
Waveform	5000	21	3

*Subset of the original dataset is taken.

Table B.3: Key properties of the real world benchmark class imbalanced datasets.

Dataset Name	Number of Points	Number of Dimensions	Number of Classes	Imbalance Ratio
Abalone19	4117	8	2	129.50
Acoustic	78823	50	3	2.16
ADA	4147	48	2	3.03
Balance Scale	625	4	3	5.87
BCWO	684	10	2	1.86
Chess	28056	6	18	168.60
CodRna	59535	8	2	2.00
Colon Cancer	62	2000	2	1.90
Connect	67557	42	3	6.89
Cover Type	581012	54	7	103.13
Dermatology6	358	34	2	16.90
Diabetes	768	8	2	1.86
Ecoli-0-2-6-7_vs_3-5	224	7	2	9.18
Ecoli-0-3-4-7_vs_5-6	257	7	2	9.28
German Numeric	1000	24	2	2.33
Glass7	214	9	2	6.37
Haberman's Survival	306	3	2	2.77
Hayes-Roth	132	4	3	1.70
Hepatitis1	155	19	2	3.84
Hypothyroid3	3772	29	2	38.70
IJCNN1	141691	22	2	9.44
Liver Disorder	346	5	2	1.36
Pageblocks0	5472	10	2	8.79
Poker-8-9_vs_5	2075	10	2	82.00
Poker-8-9_vs_6	1477	10	2	85.55
Shuttle-2_vs_5	3316	9	2	66.67
Soybean12	683	35	2	14.52
SVM Guide2	391	20	3	4.16
SVM Guide4	312	10	6	2.00
SYLVA	13086	216	2	15.25
WDBC	569	30	2	1.68
Wine	178	13	3	1.47
Winequality-red-8_vs_6-7	855	11	2	46.50
Winequality-white-3-9_vs_5	1482	11	2	58.28
Winequality-white-3_vs_7	900	11	2	44.00
Yeast3	1484	8	2	8.10
Yeast4	1484	8	2	28.10

B.3 Detailed results on artificial datasets

The performance of FkNN classifier on 9 artificial datasets are detailed in terms of sum of bias over all classes μ_e, the same for Mean Squared Error σ_e, and Accuracy (β) respectively in Table B.4, B.5, and B.6, .

B.4 Detailed result on real-world datasets

We describe the comparative performances of the two initial membership estimation techniques (namely Kernel Density Estimation (KDE) and Keller's Heuristics (KH)) in terms of

η, μ_e and σ_e, on 12 real-world datasets in the following Table B.7.

B.5 Detailed results on benchmark class imbalanced real-world datasets

We detail the result on 37 real-world benchmark imbalanced datasets in terms of ACSA, GMean, μ_+, and σ_+ respectively in the following Table B.8, B.9, B.10, and B.11.

Table B.4: Performance of FkNN on artificial datasets in terms of μ_e.

N_c	k	AD-2-2-NO	AD-2-2-SO	AD-2-2-LO	AD-9-2	AD-15-2	AD-5-5	AD-10-5	AD-10-10	AD-20-10
10000	1	2.86E-10	1.16E-05	2.43E-03	8.39E-10	8.32E-11	3.47E-02	2.59E-02	1.01E-01	1.04E-01
	$N^{0.1}$	2.84E-10	8.53E-06	1.86E-03	6.38E-10	8.45E-11	2.17E-02	1.85E-02	9.15E-02	8.98E-02
	$N^{0.25}$	2.74E-10	7.24E-06	1.13E-03	5.01E-10	8.78E-11	1.69E-02	1.56E-02	9.45E-02	9.63E-02
	$N^{0.5}$	2.84E-10	5.25E-05	1.16E-03	4.59E-10	4.57E-10	2.15E-02	1.70E-02	1.86E-01	1.79E-01
50000	1	6.37E-15	4.05E-06	1.46E-03	7.71E-13	6.12E-11	2.38E-02	1.65E-02	5.99E-02	7.84E-02
	$N^{0.1}$	2.13E-15	3.19E-06	8.63E-04	7.37E-13	4.85E-11	1.62E-02	1.35E-02	4.98E-02	6.17E-02
	$N^{0.25}$	6.35E-15	4.06E-06	8.16E-04	9.98E-13	3.99E-11	1.03E-02	7.68E-03	5.38E-02	5.53E-02
	$N^{0.5}$	5.98E-14	1.73E-05	7.25E-04	1.07E-12	4.36E-10	1.44E-02	1.15E-02	1.10E-01	1.14E-01
100000	1	4.35E-14	3.60E-07	1.05E-03	1.37E-12	1.84E-12	6.30E-03	1.27E-02	3.60E-02	5.08E-02
	$N^{0.1}$	1.39E-14	1.79E-07	3.88E-04	5.03E-13	6.32E-12	5.24E-03	7.50E-03	2.95E-02	3.40E-02
	$N^{0.25}$	1.40E-14	1.91E-07	3.74E-04	4.88E-13	2.29E-12	3.06E-03	4.50E-03	2.87E-02	3.15E-02
	$N^{0.5}$	5.73E-13	7.62E-07	4.33E-04	5.46E-12	4.86E-11	6.69E-03	6.74E-03	6.41E-02	6.84E-02
150000	1	1.47E-16	3.45E-07	7.65E-04	3.31E-13	3.54E-13	9.54E-03	6.90E-03	2.11E-02	2.55E-02
	$N^{0.1}$	1.41E-16	1.90E-07	4.17E-04	2.32E-13	1.71E-13	7.34E-03	4.70E-03	1.71E-02	1.84E-02
	$N^{0.25}$	1.87E-16	6.22E-08	3.36E-04	1.31E-13	1.73E-13	5.88E-03	2.75E-03	1.67E-02	1.75E-02
	$N^{0.5}$	2.01E-15	8.88E-07	5.59E-04	2.13E-12	1.58E-12	7.64E-03	4.08E-03	4.06E-02	3.60E-02

Table B.5: Performance of FkNN on artificial datasets in terms of σ_e.

N_e	k	AD-2-2-NO	AD-2-2-SO	AD-2-2-LO	AD-9-2	AD-15-2	AD-5-5	AD-10-5	AD-10-10	AD-20-10
10000	1	8.18E-18	5.19E-09	3.17E-05	1.99E-16	3.32E-18	1.59E-02	5.38E-03	5.95E-02	5.66E-02
	$N^{0.1}$	8.07E-18	2.47E-09	2.68E-05	1.34E-16	3.41E-18	4.09E-03	2.21E-03	2.84E-02	2.65E-02
	$N^{0.25}$	7.51E-18	1.76E-09	1.05E-05	6.40E-17	3.71E-18	2.27E-03	1.37E-03	1.72E-02	1.67E-02
	$N^{0.5}$	8.02E-18	1.44E-07	6.28E-06	4.30E-17	5.12E-17	1.60E-03	7.87E-04	2.89E-02	2.49E-02
50000	1	2.86E-27	1.41E-09	2.22E-05	1.07E-22	2.00E-18	6.59E-03	2.35E-03	3.60E-02	4.15E-02
	$N^{0.1}$	2.08E-28	9.73E-10	7.29E-06	2.01E-22	1.27E-18	2.69E-03	1.63E-03	1.42E-02	1.79E-02
	$N^{0.25}$	3.52E-27	1.50E-09	5.84E-06	3.56E-22	8.72E-19	1.00E-03	4.41E-04	7.76E-03	8.04E-03
	$N^{0.5}$	2.54E-25	2.80E-08	4.14E-06	1.45E-22	1.09E-16	9.49E-04	5.01E-04	1.48E-02	1.46E-02
100000	1	1.37E-25	4.62E-12	7.73E-06	5.74E-22	9.08E-22	8.82E-04	3.18E-03	2.00E-02	2.60E-02
	$N^{0.1}$	1.18E-26	1.01E-12	1.54E-06	7.20E-23	1.70E-20	5.37E-04	7.19E-04	6.41E-03	7.48E-03
	$N^{0.25}$	1.17E-26	9.18E-13	1.09E-06	6.93E-23	1.52E-21	1.14E-04	3.03E-04	3.41E-03	3.67E-03
	$N^{0.5}$	2.56E-23	1.72E-11	1.25E-06	9.87E-21	1.02E-18	2.74E-04	2.77E-04	7.43E-03	7.86E-03
150000	1	8.94E-31	7.37E-12	5.93E-06	2.44E-23	7.44E-23	2.18E-03	1.09E-03	9.07E-03	1.08E-02
	$N^{0.1}$	1.25E-30	2.05E-12	1.75E-06	1.73E-23	1.23E-23	1.02E-03	5.71E-04	3.37E-03	3.76E-03
	$N^{0.25}$	2.34E-30	1.98E-13	8.54E-07	4.74E-24	1.52E-23	5.33E-04	1.48E-04	1.85E-03	1.88E-03
	$N^{0.5}$	1.37E-28	6.18E-11	1.12E-06	1.42E-21	8.80E-22	5.12E-04	1.48E-04	5.05E-03	4.06E-03

Table B.6: Performance of FkNN on artificial datasets in terms of Accuracy.

N_c	k	AD-2-2-NO	AD-2-2-SO	AD-2-2-LO	AD-9-2	AD-15-2	AD-5-5	AD-10-5	AD-10-10	AD-20-10
10000	1	1.00	1.00	0.98	1.00	1.00	0.99	0.97	0.95	0.94
	$N^{0.1}$	1.00	1.00	0.98	1.00	1.00	0.99	0.98	0.96	0.95
	$N^{0.25}$	1.00	1.00	0.98	1.00	1.00	0.99	0.98	0.97	0.97
	$N^{0.5}$	1.00	1.00	0.98	1.00	1.00	0.99	0.98	0.98	0.98
50000	1	1.00	1.00	0.98	1.00	1.00	0.98	0.98	0.97	0.95
	$N^{0.1}$	1.00	1.00	0.98	1.00	1.00	0.98	0.99	0.98	0.96
	$N^{0.25}$	1.00	1.00	0.98	1.00	1.00	0.98	0.99	0.98	0.97
	$N^{0.5}$	1.00	1.00	0.98	1.00	1.00	0.98	0.99	0.98	0.98
100000	1	1.00	1.00	0.96	1.00	1.00	0.99	0.98	0.97	0.96
	$N^{0.1}$	1.00	1.00	0.96	1.00	1.00	0.99	0.98	0.98	0.97
	$N^{0.25}$	1.00	1.00	0.96	1.00	1.00	0.99	0.98	0.98	0.97
	$N^{0.5}$	1.00	1.00	0.96	1.00	1.00	0.99	0.98	0.98	0.97
150000	1	1.00	1.00	0.99	1.00	1.00	0.99	0.98	0.98	0.97
	$N^{0.1}$	1.00	1.00	0.99	1.00	1.00	0.99	0.98	0.98	0.98
	$N^{0.25}$	1.00	1.00	0.99	1.00	1.00	0.99	0.98	0.98	0.98
	$N^{0.5}$	1.00	1.00	0.99	1.00	1.00	0.99	0.98	0.99	0.98

Table B.7: Comparison of performance on real world datasets with initial fuzzy memberships estimated using KDE and KH.

			Banana	Diabetic	Heart	Iris	Letter	Magic	Penbased	Ring	Seeds	Segment	Twonorm	Waveform
$k = 1$	η	KDE	0.90	0.53	0.75	0.95	0.94	0.70	0.97	0.75	0.91	0.96	0.94	0.77
		KH	0.87	0.61	0.75	0.95	0.95	0.79	0.99	0.75	0.92	0.96	0.94	0.77
	μ_e	KDE	0.12	0.22	0.30	0.06	0.15	0.19	0.05	0.69	0.14	0.07	0.11	0.39
		KH	0.14	0.43	0.32	0.10	0.15	0.25	0.03	0.33	0.14	0.07	0.12	0.33
	σ_e	KDE	0.01	0.01	0.27	0.03	0.05	0.03	0.03	0.67	0.08	0.07	0.07	0.25
		KH	0.07	0.21	0.15	0.04	0.03	0.12	0.01	0.15	0.06	0.02	0.05	0.16
$k = N^{0.1}$	η	KDE	0.90	0.53	0.75	0.95	0.94	0.69	0.97	0.71	0.93	0.96	0.96	0.80
		KH	0.89	0.62	0.83	0.95	0.95	0.82	0.99	0.60	0.93	0.96	0.96	0.80
	μ_e	KDE	0.11	0.21	0.33	0.05	0.13	0.18	0.05	0.75	0.10	0.09	0.09	0.31
		KH	0.14	0.43	0.29	0.10	0.15	0.24	0.03	0.36	0.12	0.07	0.10	0.30
	σ_e	KDE	0.01	0.01	0.19	0.02	0.03	0.02	0.03	0.62	0.04	0.06	0.04	0.13
		KH	0.05	0.14	0.09	0.03	0.02	0.07	0.01	0.17	0.04	0.02	0.02	0.09
$k = N^{0.25}$	η	KDE	0.90	0.53	0.83	0.95	0.93	0.69	0.97	0.66	0.92	0.96	0.97	0.83
		KH	0.90	0.63	0.83	0.95	0.94	0.81	0.99	0.54	0.93	0.95	0.98	0.83
	μ_e	KDE	0.11	0.31	0.36	0.05	0.17	0.19	0.06	0.83	0.09	0.11	0.11	0.32
		KH	0.14	0.43	0.27	0.10	0.19	0.25	0.03	0.40	0.11	0.08	0.08	0.29
	σ_e	KDE	0.01	0.01	0.18	0.02	0.03	0.02	0.03	0.66	0.03	0.06	0.04	0.12
		KH	0.04	0.13	0.08	0.03	0.02	0.08	0.01	0.17	0.03	0.02	0.01	0.07
$k = N^{0.5}$	acc	KDE	0.90	0.53	0.85	0.96	0.87	0.67	0.96	0.54	0.91	0.94	0.98	0.86
		KH	0.90	0.65	0.85	0.97	0.86	0.79	0.97	0.51	0.93	0.93	0.97	0.85
	μ_e	KDE	0.02	0.01	0.42	0.08	0.58	0.13	0.12	0.96	0.12	0.22	0.15	0.39
		KH	0.14	0.45	0.27	0.09	0.44	0.31	0.07	0.48	0.12	0.13	0.10	0.32
	σ_e	KDE	0.00	0.01	0.18	0.02	0.17	0.04	0.05	0.79	0.04	0.09	0.05	0.14
		KH	0.04	0.12	0.06	0.02	0.08	0.08	0.01	0.23	0.03	0.03	0.01	0.07

Table B.8: Comparison of performance on real-world benchmark class imbalanced datasets in terms of ACSA

Dataset	FkNN	WFkNN +GCW	FkNN +SMOTE	FkNN +RUS	NWFkNN	WFkNN +LACW
Abalone19	0.50†	0.64†	0.58†	0.50†	0.50†	**0.66**
Acoustic	0.62†	0.66†	0.62†	**0.68**$^{\approx}$	0.52†	**0.68**
ADA	0.69†	0.76†	0.73†	0.73†	0.58†	**0.79**
Balance	0.62$^{\approx}$	0.60†	**0.63**$^{\approx}$	0.58†	**0.63**$^{\approx}$	0.62
BCWO	0.95†	0.96$^{\approx}$	0.96$^{\approx}$	0.96$^{\approx}$	0.95†	**0.97**
Chess	0.42†	0.46†	0.43†	0.36†	0.28†	**0.48**
CodRna	0.85†	0.89$^{\approx}$	0.89$^{\approx}$	0.89$^{\approx}$	0.77†	**0.90**
Colon Cancer	0.69†	**0.80**$^{\approx}$	0.77†	0.76†	0.57†	**0.80**
Connect	0.41†	0.47$^{\approx}$	0.45†	0.44†	0.34†	**0.48**
Cover Type	0.56†	0.68†	0.59†	0.51†	0.30†	**0.70**
Dermatology6	0.99†	**1.00**$^{\approx}$	**1.00**$^{\approx}$	**1.00**$^{\approx}$	0.79†	**1.00**
Diabetes	0.68†	0.72$^{\approx}$	0.72$^{\approx}$	0.71†	0.61†	**0.73**
Ecoli-0-2-6-7_vs_3-5	0.78†	0.86$^{\approx}$	0.86$^{\approx}$	0.83†	0.50†	**0.87**
Ecoli-0-3-4-7_vs_5-6	0.86†	0.88†	0.89†	0.91$^{\approx}$	0.50†	**0.92**
German Numeric	0.58†	0.65†	0.65†	0.65†	0.53†	**0.69**
Glass7	0.89†	0.92†	0.93†	0.91†	0.88†	**0.95**
Haberman's Survival	0.58†	0.61†	**0.64**$^{\approx}$	0.63$^{\approx}$	0.54†	0.63
Hayes-Roth	0.68†	0.73†	0.73†	0.71†	0.60†	**0.75**
Hepatitis1	0.69†	0.76†	0.78†	**0.81**$^{\approx}$	0.59†	**0.81**
Hypothyroid3	0.77†	0.92$^{\approx}$	0.91†	0.82$^{\approx}$	0.50†	**0.93**
IJCNN1	0.65†	0.87†	0.84†	0.78†	0.50†	**0.89**
Liver Disorder	**0.65**$^{\approx}$	0.64$^{\approx}$	0.64$^{\approx}$	0.63†	0.64$^{\approx}$	**0.65**
Pageblocks0	0.86†	0.93†	0.94$^{\approx}$	0.92†	0.66†	**0.95**
Poker-8-9_vs_5	0.50†	0.59†	**0.65**†	0.51†	0.50†	0.61
Poker-8-9_vs_6	0.63†	0.85$^{\approx}$	0.85$^{\approx}$	0.73†	0.50†	**0.86**
Shuttle-2_vs_5	**1.00**$^{\approx}$	**1.00**$^{\approx}$	**1.00**$^{\approx}$	**1.00**$^{\approx}$	**1.00**$^{\approx}$	**1.00**
Soybean12	0.87†	0.95†	**0.99**$^{\approx}$	0.92†	0.50 †	0.98
SVM Guide2	0.62†	0.70†	0.67†	0.68†	0.53†	**0.72**
SVM Guide4	0.63†	0.65†	0.63†	0.64†	0.66$^{\approx}$	**0.67**
SYLVA	0.65†	0.92$^{\approx}$	0.92$^{\approx}$	0.79†	0.50†	**0.93**
WDBC	0.94$^{\approx}$	0.93†	0.92†	**0.95**$^{\approx}$	0.94$^{\approx}$	**0.95**
Wine	0.96$^{\approx}$	0.96$^{\approx}$	0.96$^{\approx}$	0.95†	0.96$^{\approx}$	**0.97**
Winequality-red-8_vs_6-7	0.55†	0.65$^{\approx}$	0.61†	0.56†	0.50†	**0.66**
Winequality-white-3-9_vs_5	0.52†	0.63†	0.65$^{\approx}$	0.53†	0.50†	**0.66**
Winequality-white-3_vs_7	0.52†	0.56†	**0.63**†	0.53†	0.62$^{\approx}$	0.61
Yeast3	0.83†	0.90†	0.90†	0.90†	0.62†	**0.92**
Yeast4	0.55†	0.82$^{\approx}$	0.79†	0.73†	0.50†	**0.83**

$\dagger$: The performances are significantly different as indicated by the WRS.
$\approx$: The performances are statistically comparable as indicated by the WRS.
The best result is boldfaced.

Table B.9: Comparison of performance on real-world benchmark class imbalanced datasets in terms of GMean

Dataset	FkNN	WFkNN +GCW	FkNN +SMOTE	FkNN +RUS	NWFkNN	WFkNN +LACW
Abalone19	$0.00^{\dagger}$	$0.60^{\dagger}$	$0.45^{\dagger}$	$0.00^{\dagger}$	$0.00^{\dagger}$	**0.62**
Acoustic	$0.61^{\dagger}$	$0.64^{\dagger}$	$0.61^{\dagger}$	$0.67^{\approx}$	$0.42^{\dagger}$	**0.68**
ADA	$0.65^{\dagger}$	$0.74^{\approx}$	$0.72^{\dagger}$	$0.73^{\dagger}$	$0.39^{\dagger}$	**0.75**
Balance	$0.50^{\dagger}$	$0.48^{\dagger}$	**0.52**$^{\approx}$	$0.42^{\dagger}$	**0.52**$^{\approx}$	0.51
BCWO	$0.95^{\approx}$	**0.96**$^{\approx}$	**0.96**$^{\approx}$	**0.96**$^{\approx}$	$0.95^{\approx}$	**0.96**
Chess	$0.14^{\dagger}$	**0.27**$^{\approx}$	$0.08^{\dagger}$	$0.07^{\dagger}$	$0.00^{\dagger}$	**0.27**
CodRna	$0.84^{\dagger}$	$0.89^{\approx}$	$0.89^{\approx}$	$0.89^{\approx}$	$0.74^{\dagger}$	**0.90**
Colon Cancer	$0.54^{\dagger}$	$0.75^{\approx}$	$0.74^{\dagger}$	$0.74^{\dagger}$	$0.28^{\dagger}$	**0.76**
Connect	$0.18^{\dagger}$	**0.46**$^{\approx}$	$0.17^{\dagger}$	$0.41^{\dagger}$	$0.00^{\dagger}$	**0.46**
Cover Type	$0.36^{\dagger}$	$0.67^{\approx}$	$0.37^{\dagger}$	$0.33^{\dagger}$	$0.00^{\dagger}$	**0.68**
Dermatology6	$0.99^{\approx}$	**1.00**$^{\approx}$	**1.00**$^{\approx}$	**1.00**$^{\approx}$	$0.66^{\dagger}$	**1.00**
Diabetes	$0.66^{\dagger}$	$0.71^{\dagger}$	$0.72^{\approx}$	$0.71^{\dagger}$	$0.50^{\dagger}$	**0.73**
Ecoli-0-2-6-7_vs_3-5	$0.65^{\dagger}$	$0.80^{\dagger}$	$0.81^{\dagger}$	$0.79^{\dagger}$	$0.00^{\dagger}$	**0.83**
Ecoli-0-3-4-7_vs_5-6	$0.82^{\dagger}$	$0.88^{\dagger}$	$0.88^{\dagger}$	$0.90^{\approx}$	$0.00^{\dagger}$	**0.91**
German Numeric	$0.45^{\dagger}$	$0.65^{\dagger}$	$0.65^{\dagger}$	$0.66^{\dagger}$	$0.20^{\dagger}$	**0.68**
Glass7	$0.87^{\dagger}$	$0.90^{\dagger}$	$0.92^{\dagger}$	$0.90^{\dagger}$	$0.86^{\dagger}$	**0.94**
Haberman's Survival	$0.49^{\dagger}$	$0.61^{\dagger}$	**0.63**$^{\approx}$	$0.62^{\approx}$	$0.20^{\dagger}$	0.62
Hayes-Roth	$0.61^{\dagger}$	$0.68^{\dagger}$	$0.69^{\dagger}$	$0.67^{\dagger}$	$0.37^{\dagger}$	**0.71**
Hepatitis1	$0.58^{\dagger}$	$0.74^{\dagger}$	$0.74^{\dagger}$	$0.80^{\approx}$	$0.29^{\dagger}$	**0.81**
Hypothyroid3	$0.73^{\dagger}$	$0.91^{\approx}$	$0.91^{\approx}$	$0.80^{\dagger}$	$0.00^{\dagger}$	**0.92**
IJCNN1	$0.47^{\dagger}$	$0.87^{\dagger}$	$0.82^{\dagger}$	$0.73^{\dagger}$	$0.00^{\dagger}$	**0.89**
Liver Disorder	$0.63^{\approx}$	**0.64**$^{\approx}$	$0.63^{\approx}$	$0.62^{\dagger}$	$0.63^{\approx}$	**0.64**
Pageblocks0	$0.85^{\dagger}$	$0.92^{\dagger}$	**0.94**$^{\approx}$	$0.92^{\dagger}$	$0.56^{\dagger}$	**0.94**
Poker-8-9_vs_5	$0.01^{\dagger}$	$0.38^{\dagger}$	**0.47**†	$0.05^{\dagger}$	$0.00^{\dagger}$	**0.45**
Poker-8-9_vs_6	$0.30^{\dagger}$	$0.84^{\approx}$	$0.84^{\approx}$	$0.52^{\dagger}$	$0.00^{\dagger}$	0.85
Shuttle-2_vs_5	**1.00**$^{\approx}$	**1.00**$^{\approx}$	**1.00**$^{\approx}$	**1.00**$^{\approx}$	**1.00**$^{\approx}$	**1.00**
Soybean12	$0.84^{\dagger}$	$0.95^{\dagger}$	**0.99**$^{\approx}$	$0.91^{\dagger}$	$0.00^{\dagger}$	0.98
SVM Guide2	$0.48^{\dagger}$	$0.67^{\dagger}$	$0.59^{\dagger}$	$0.65^{\dagger}$	$0.32^{\dagger}$	**0.69**
SVM Guide4	$0.44^{\dagger}$	$0.59^{\dagger}$	$0.45^{\dagger}$	$0.37^{\dagger}$	**0.61**$^{\approx}$	**0.61**
SYLVA	$0.44^{\dagger}$	**0.92**$^{\approx}$	$0.91^{\approx}$	$0.68^{\dagger}$	$0.00^{\dagger}$	**0.92**
WDBC	$0.94^{\approx}$	$0.93^{\dagger}$	$0.92^{\dagger}$	$0.95^{\approx}$	$0.93^{\dagger}$	**0.95**
Wine	**0.96**$^{\approx}$	**0.96**$^{\approx}$	**0.96**$^{\approx}$	$0.94^{\dagger}$	**0.96**$^{\approx}$	**0.96**
Winequality-red-8_vs_6-7	$0.14^{\dagger}$	$0.55^{\dagger}$	$0.38^{\dagger}$	$0.19^{\dagger}$	$0.00^{\dagger}$	**0.58**
Winequality-white-3-9_vs_5	$0.06^{\dagger}$	$0.43^{\dagger}$	$0.51^{\dagger}$	$0.11^{\dagger}$	$0.00^{\dagger}$	**0.52**
Winequality-white-3_vs_7	$0.05^{\dagger}$	$0.23^{\dagger}$	**0.45**†	$0.09^{\dagger}$	**0.44**†	0.39
Yeast3	$0.81^{\dagger}$	$0.90^{\dagger}$	$0.90^{\dagger}$	$0.90^{\dagger}$	$0.47^{\dagger}$	**0.92**
Yeast4	$0.17^{\dagger}$	$0.81^{\approx}$	$0.77^{\dagger}$	$0.68^{\dagger}$	$0.00^{\dagger}$	**0.82**

†: The performances are significantly different as indicated by the WRS.
≈: The performances are statistically comparable as indicated by the WRS.
The best result is boldfaced.

Table B.10: Comparison of performance on real-world benchmark class imbalanced datasets in terms of μ_+.

Dataset	FkNN	WFkNN +GCW	FkNN +SMOTE	FkNN +RUS	NWFkNN	WFkNN +LACW
Abalone19	$0.50^{\dagger}$	$0.23^{\approx}$	$0.41^{\dagger}$	$0.48^{\dagger}$	$0.52^{\dagger}$	**0.22**
Acoustic	$0.29^{\dagger}$	$0.20^{\approx}$	$0.30^{\dagger}$	$0.24^{\dagger}$	$0.40^{\dagger}$	**0.19**
ADA	$0.26^{\dagger}$	**0.14**$^{\approx}$	$0.21^{\dagger}$	$0.16^{\dagger}$	$0.46^{\dagger}$	**0.14**
Balance	$0.51^{\dagger}$	$0.30^{\dagger}$	**0.29**$^{\approx}$	$0.39^{\dagger}$	$0.57^{\dagger}$	**0.29**
BCWO	$0.06^{\dagger}$	**0.04**$^{\approx}$	**0.04**$^{\approx}$	**0.04**$^{\approx}$	$0.09^{\dagger}$	**0.04**
Chess	$0.38^{\dagger}$	$0.24^{\approx}$	$0.38^{\dagger}$	$0.29^{\dagger}$	$0.53^{\dagger}$	**0.23**
CodRna	$0.16^{\dagger}$	**0.10**$^{\approx}$	$0.11^{\approx}$	$0.11^{\approx}$	$0.27^{\dagger}$	**0.10**
Colon Cancer	$0.27^{\dagger}$	$0.18^{\dagger}$	$0.15^{\approx}$	$0.22^{\dagger}$	$0.39^{\dagger}$	**0.14**
Connect	$0.44^{\dagger}$	**0.26**$^{\approx}$	$0.45^{\dagger}$	$0.33^{\dagger}$	$0.54^{\dagger}$	**0.26**
Cover Type	$0.30^{\dagger}$	$0.28^{\dagger}$	**0.21**†	$0.29^{\dagger}$	$0.53^{\dagger}$	0.27
Dermatology6	$0.05^{\dagger}$	**0.00**$^{\approx}$	$0.00^{\approx}$	$0.02^{\dagger}$	$0.46^{\dagger}$	**0.00**
Diabetes	$0.26^{\dagger}$	**0.17**$^{\approx}$	$0.20^{\dagger}$	$0.20^{\dagger}$	$0.38^{\dagger}$	**0.17**
Ecoli-0-2-6-7_vs_3-5	$0.24^{\dagger}$	$0.17^{\dagger}$	$0.17^{\dagger}$	$0.19^{\dagger}$	$0.57^{\dagger}$	**0.15**
Ecoli-0-3-4-7_vs_5-6	$0.21^{\dagger}$	$0.21^{\dagger}$	**0.20**$^{\approx}$	$0.22^{\dagger}$	$0.56^{\dagger}$	**0.20**
German Numeric	$0.32^{\dagger}$	$0.19^{\dagger}$	$0.23^{\dagger}$	$0.23^{\dagger}$	$0.47^{\dagger}$	**0.17**
Glass7	$0.11^{\dagger}$	$0.08^{\dagger}$	$0.07^{\dagger}$	$0.09^{\dagger}$	$0.18^{\dagger}$	**0.05**
Haberman's Survival	$0.33^{\dagger}$	$0.22^{\dagger}$	$0.23^{\dagger}$	$0.22^{\dagger}$	$0.47^{\dagger}$	**0.20**
Hayes-Roth	$0.30^{\dagger}$	$0.22^{\dagger}$	$0.26^{\dagger}$	$0.29^{\dagger}$	$0.41^{\dagger}$	**0.21**
Hepatitis1	$0.30^{\dagger}$	$0.16^{\dagger}$	$0.16^{\dagger}$	$0.17^{\dagger}$	$0.48^{\dagger}$	**0.12**
Hypothyroid3	$0.22^{\dagger}$	$0.23^{\dagger}$	$0.12^{\approx}$	$0.20^{\dagger}$	$0.64^{\dagger}$	**0.11**
IJCNN1	$0.27^{\dagger}$	**0.19**$^{\approx}$	$0.21^{\approx}$	$0.23^{\dagger}$	$0.53^{\dagger}$	0.20
Liver Disorder	$0.25^{\dagger}$	**0.20**$^{\approx}$	$0.22^{\dagger}$	$0.22^{\dagger}$	$0.31^{\dagger}$	**0.20**
Pageblocks0	$0.14^{\dagger}$	$0.14^{\dagger}$	**0.08**†	$0.14^{\dagger}$	$0.45^{\dagger}$	0.15
Poker-8-9_vs_5	$0.50^{\dagger}$	**0.21**$^{\approx}$	$0.22^{\approx}$	$0.47^{\dagger}$	$0.52^{\dagger}$	0.22
Poker-8-9_vs_6	$0.39^{\dagger}$	$0.36^{\dagger}$	$0.33^{\approx}$	$0.45^{\dagger}$	$0.55^{\dagger}$	**0.34**
Shuttle-2_vs_5	$0.02^{\dagger}$	$0.17^{\approx}$	**0.00**†	$0.01^{\dagger}$	$0.74^{\dagger}$	0.17
Soybean12	$0.23^{\dagger}$	$0.24^{\dagger}$	**0.21**†	$0.22^{\approx}$	$0.59^{\dagger}$	0.23
SVM Guide2	$0.34^{\dagger}$	$0.20^{\dagger}$	**0.14**†	$0.28^{\dagger}$	$0.48^{\dagger}$	0.18
SVM Guide4	$0.31^{\dagger}$	**0.24**$^{\approx}$	$0.31^{\dagger}$	$0.27^{\dagger}$	$0.39^{\dagger}$	**0.24**
SYLVA	$0.30^{\dagger}$	**0.25**$^{\approx}$	$0.27^{\approx}$	$0.32^{\dagger}$	$0.58^{\dagger}$	0.26
WDBC	$0.08^{\dagger}$	$0.06^{\approx}$	$0.05^{\approx}$	$0.06^{\approx}$	$0.10^{\dagger}$	0.06
Wine	$0.02^{\approx}$	**0.01**$^{\approx}$	$0.01^{\approx}$	$0.01^{\approx}$	$0.03^{\approx}$	**0.01**
Winequality-red-8_vs_6-7	$0.45^{\dagger}$	**0.20**$^{\approx}$	$0.33^{\dagger}$	$0.40^{\dagger}$	$0.52^{\dagger}$	**0.20**
Winequality-white-3-9_vs_5	$0.47^{\dagger}$	$0.36^{\dagger}$	$0.35^{\dagger}$	$0.45^{\dagger}$	$0.53^{\dagger}$	**0.31**
Winequality-white-3_vs_7	$0.48^{\dagger}$	$0.38^{\dagger}$	**0.34**†	$0.46^{\dagger}$	$0.52^{\dagger}$	0.36
Yeast3	$0.20^{\dagger}$	$0.12^{\dagger}$	$0.11^{\approx}$	$0.11^{\approx}$	$0.52^{\dagger}$	**0.10**
Yeast4	$0.38^{\dagger}$	$0.20^{\dagger}$	$0.20^{\dagger}$	$0.27^{\dagger}$	$0.58^{\dagger}$	**0.17**

$\dagger$: The performances are significantly different as indicated by the WRS.
$\approx$: The performances are statistically comparable as indicated by the WRS.
The best result is boldfaced.

Table B.11: Comparison of performance on real-world benchmark class imbalanced datasets in terms of σ_+.

Dataset	FkNN	WFkNN +GCW	FkNN +SMOTE	FkNN +RUS	NWFkNN	WFkNN +LACW
Abalone19	$0.26^{\dagger}$	$0.08^{\dagger}$	$0.20^{\dagger}$	$0.23^{\dagger}$	$0.27^{\dagger}$	**0.04**
Acoustic	$0.11^{\dagger}$	$0.06^{\dagger}$	$0.12^{\dagger}$	$0.08^{\dagger}$	$0.18^{\dagger}$	**0.04**
ADA	$0.10^{\dagger}$	$0.04^{\dagger}$	$0.08^{\dagger}$	$0.05^{\dagger}$	$0.23^{\dagger}$	**0.03**
Balance	$0.26^{\dagger}$	$0.10^{\dagger}$	$0.14^{\dagger}$	$0.17^{\dagger}$	$0.32^{\dagger}$	**0.09**
BCWO	$0.02^{\approx}$	**0.01**$^{\approx}$	**0.01**$^{\approx}$	$0.01^{\approx}$	$0.03^{\dagger}$	**0.01**
Chess	$0.17^{\dagger}$	$0.09^{\dagger}$	$0.17^{\dagger}$	$0.11^{\dagger}$	$0.28^{\dagger}$	**0.07**
CodRna	$0.05^{\dagger}$	**0.02**$^{\approx}$	$0.03^{\approx}$	$0.03^{\dagger}$	$0.10^{\dagger}$	**0.02**
Colon Cancer	$0.11^{\dagger}$	$0.07^{\approx}$	**0.05**†	$0.07^{\approx}$	$0.18^{\dagger}$	0.07
Connect	$0.21^{\dagger}$	**0.08**$^{\approx}$	$0.22^{\dagger}$	$0.13^{\dagger}$	$0.29^{\dagger}$	**0.08**
Cover Type	$0.13^{\dagger}$	$0.09^{\dagger}$	$0.13^{\dagger}$	$0.08^{\dagger}$	$0.28^{\dagger}$	**0.07**
Dermatology6	$0.01^{\dagger}$	**0.00**$^{\approx}$	**0.00**$^{\approx}$	**0.00**$^{\approx}$	$0.23^{\dagger}$	**0.00**
Diabetes	$0.10^{\dagger}$	$0.05^{\dagger}$	$0.07^{\dagger}$	$0.07^{\dagger}$	$0.17^{\dagger}$	**0.03**
Ecoli-0-2-6-7_vs_3-5	$0.09^{\dagger}$	**0.04**$^{\approx}$	$0.07^{\dagger}$	$0.05^{\dagger}$	$0.33^{\dagger}$	**0.04**
Ecoli-0-3-4-7_vs_5-6	$0.08^{\dagger}$	$0.06^{\approx}$	**0.05**$^{\approx}$	$0.06^{\approx}$	$0.32^{\dagger}$	0.06
German Numeric	$0.13^{\dagger}$	$0.06^{\dagger}$	$0.09^{\dagger}$	$0.08^{\dagger}$	$0.24^{\dagger}$	**0.05**
Glass7	$0.05^{\dagger}$	$0.04^{\dagger}$	**0.03**$^{\approx}$	$0.04^{\approx}$	$0.09^{\dagger}$	**0.03**
Haberman's Survival	$0.14^{\dagger}$	$0.06^{\dagger}$	$0.09^{\dagger}$	$0.08^{\dagger}$	$0.24^{\dagger}$	**0.05**
Hayes-Roth	$0.12^{\dagger}$	$0.08^{\approx}$	$0.11^{\dagger}$	$0.12^{\dagger}$	$0.20^{\dagger}$	**0.07**
Hepatitis1	$0.13^{\dagger}$	$0.05^{\approx}$	$0.07^{\approx}$	$0.06^{\approx}$	$0.25^{\dagger}$	0.06
Hypothyroid3	$0.09^{\dagger}$	$0.07^{\approx}$	**0.06**$^{\approx}$	$0.08^{\approx}$	$0.43^{\dagger}$	0.07
IJCNN1	$0.11^{\dagger}$	$0.06^{\dagger}$	$0.07^{\dagger}$	$0.06^{\dagger}$	$0.29^{\dagger}$	**0.01**
Liver Disorder	$0.09^{\dagger}$	$0.06^{\dagger}$	$0.08^{\dagger}$	$0.08^{\dagger}$	$0.13^{\dagger}$	**0.05**
Pageblocks0	$0.05^{\dagger}$	$0.03^{\dagger}$	**0.02**$^{\approx}$	$0.03^{\dagger}$	$0.23^{\dagger}$	**0.02**
Poker-8-9_vs_5	$0.25^{\dagger}$	$0.07^{\dagger}$	$0.18^{\dagger}$	$0.23^{\dagger}$	$0.27^{\dagger}$	**0.05**
Poker-8-9_vs_6	$0.12^{\dagger}$	$0.14^{\dagger}$	**0.01**†	$0.09^{\dagger}$	$0.31^{\dagger}$	0.13
Shuttle-2_vs_5	$0.04^{\dagger}$	$0.05^{\dagger}$	**0.00**†	$0.04^{\approx}$	$0.55^{\dagger}$	0.04
Soybean12	$0.04^{\dagger}$	$0.07^{\dagger}$	**0.01**†	$0.03^{\dagger}$	$0.35^{\dagger}$	0.06
SVM Guide2	$0.15^{\dagger}$	$0.09^{\dagger}$	$0.08^{\approx}$	$0.11^{\dagger}$	$0.25^{\dagger}$	**0.08**
SVM Guide4	$0.13^{\dagger}$	**0.09**$^{\approx}$	$0.13^{\dagger}$	$0.10^{\dagger}$	$0.18^{\dagger}$	**0.09**
SYLVA	$0.13^{\dagger}$	$0.07^{\dagger}$	$0.03^{\dagger}$	**0.08**$^{\approx}$	$0.34^{\dagger}$	0.08
WDBC	$0.03^{\approx}$	**0.02**$^{\approx}$	**0.02**$^{\approx}$	**0.02**$^{\approx}$	$0.04^{\dagger}$	**0.02**
Wine	**0.00**$^{\approx}$	**0.00**$^{\approx}$	**0.00**$^{\approx}$	**0.00**$^{\approx}$	$0.01^{\dagger}$	**0.00**
Winequality-red-8_vs_6-7	$0.22^{\dagger}$	**0.07**$^{\approx}$	$0.15^{\dagger}$	$0.19^{\dagger}$	$0.28^{\dagger}$	**0.07**
Winequality-white-3-9_vs_5	$0.23^{\dagger}$	**0.15**$^{\approx}$	**0.15**$^{\approx}$	$0.21^{\dagger}$	$0.28^{\dagger}$	**0.15**
Winequality-white-3_vs_7	$0.25^{\dagger}$	$0.17^{\dagger}$	$0.20^{\dagger}$	$0.23^{\dagger}$	$0.28^{\dagger}$	**0.16**
Yeast3	$0.07^{\dagger}$	**0.02**$^{\approx}$	$0.04^{\dagger}$	$0.03^{\dagger}$	$0.29^{\dagger}$	**0.02**
Yeast4	$0.17^{\dagger}$	**0.08**$^{\approx}$	$0.10^{\dagger}$	$0.10^{\dagger}$	$0.34^{\dagger}$	**0.08**

†: The performances are significantly different as indicated by the WRS.
≈: The performances are statistically comparable as indicated by the WRS.
The best result is boldfaced.

Appendix C

Supplementary for Chapter 4

C.1 Details of the datasets used for the empirical validation of PIFWkNN and PIFW^2kNN

The key properties of the 20 real world datasets used for validating the efficacy of PIFWkNN are detailed in the following Table C.1. The different attributes of the 20 real world class imbalanced datasets used for evaluating the performance of PIFW^2kNN are listed in Table C.2.

C.2 Detailed results on real-world datasets

The comparative study in terms of Accuracy is listed in Table C.4, while the optimized values of parameters used for obtaining the reported results are detailed in Table C.3. The performance of PIFW^2kNN along with its contenders on real-world class imbalanced benchmark datasets in terms of GMean and ACSA are respectively detailed in Table C.5 and C.6.

Table C.1: Key properties of the real-world datasets used to validate the efficacy of PIFWkNN.

Dataset Name	Number of Points	Number of Dimensions	Number of Classes
Australian	690	14	2
Balance Scale	625	4	5
Breats Tissue	106	10	6
Breast Cancer	569	32	2
Diabetes	768	8	2
Ecoli	366	8	8
Haberman	306	3	2
Heart Statlog	270	13	2
Hill Valley	606	100	2
Ionosphere	351	34	2
Ozone	2536	73	2
Pendigits	10922	16	10
Postoperative	90	9	3
Sonar	208	60	2
Transfusion	151	5	3
Vehicle	946	18	4
Vote	435	16	2
Vowel	910	11	11
Wine	178	13	3
Yeast	1484	8	10

Table C.2: Key properties of the real-world class imbalanced benchmark datasets used to validate the efficacy of PIFW^2kNN.

Dataset Name	Number of Points	Number of Dimensions	Number of Classes	Imbalance Ratio
Abalone19	4117	8	2	129.50
Acoustic	78823	50	3	2.16
ADA	4147	48	2	3.03
Biodeg	1055	41	2	1.96
BCWO	684	10	2	1.86
Chess	28056	6	18	168.60
CodRna	59535	8	2	2.00
Colon Cancer	62	2000	2	1.90
Dermatology6	358	34	2	16.90
German Numeric	1000	24	2	2.33
Glass7	214	9	2	6.37
Hepatitis1	155	19	2	3.84
Hypothyroid3	3772	29	2	38.70
IJCNN1	141691	22	2	9.44
Liver Disorder	346	5	2	1.36
Shuttle	43314	9	3	13.85
Soybean12	683	35	2	14.52
SVM Guide2	391	20	3	4.16
SVM Guide4	312	10	6	2.00
SYLVA	13086	216	2	15.25

Table C.3: Optimized parameter values for obtaining the Accuracy listed in the following Table C.4.

Datasets	kNN	FkNN	ENN	GA-Fuzzy kNN			EF-kNN IVFS	
	k	k	k	k	Crossover Probability	Mutation Probability	k	Initial Divergence Rate
Australian	11	9	9	5	0.9	0.010	9	0.35
Balance Scale	11	7	9	7	0.7	0.050	9	0.35
Breast Cancer	11	11	11	5	0.7	0.010	$\sqrt{N}$	0.35
Breast Tissue	1	5	3	5	0.8	0.010	1	0.35
Diabetes	5	5	5	3	0.8	0.025	5	0.35
Ecoli	11	7	7	5	0.8	0.010	9	0.35
Haberman	5	7	5	5	0.8	0.010	7	0.50
Heart Statlog	11	$\sqrt{N}$	$\sqrt{N}$	9	0.8	0.010	9	0.35
Hill Valley	3	7	5	5	0.8	0.010	3	0.35
Ionosphere	1	1	3	1	0.8	0.010	5	0.50
Ozone	$\sqrt{N}$	$\sqrt{N}$	$\sqrt{N}$	11	0.9	0.010	5	0.35
Pendigits	11	5	5	5	0.9	0.010	9	0.35
Postoperative	$\sqrt{N}$	$\sqrt{N}$	$\sqrt{N}$	7	0.8	0.025	$\sqrt{N}$	0.35
Sonar	3	3	3	5	0.8	0.010	9	0.35
Transfusion	$\sqrt{N}$	$\sqrt{N}$	$\sqrt{N}$	$\sqrt{N}$	0.8	0.010	$\sqrt{N}$	0.20
Vehicle	1	3	3	5	0.8	0.010	5	0.35
Vote	11	3	3	5	0.8	0.010	5	0.35
Vowel	1	3	3	5	0.7	0.010	1	0.35
Wine	1	3	3	5	0.8	0.010	1	0.50
Yeast	$\sqrt{N}$	$\sqrt{N}$	$\sqrt{N}$	$\sqrt{N}$	0.8	0.025	11	0.20

Table C.4: Performance comparison of PIFWkNN in terms of Accuracy.

Dataset	PIFWkNN	kNN	FkNN	ENN	GA-Fuzzy kNN	EF-kNN IVFS	kLDEDW	CW	DE4
Australian	**0.87**	0.84†	0.85$^{\approx}$	0.84†	0.83†	0.85†	0.86	0.81	0.82
Balance Scale	**0.90**	0.89$^{\approx}$	0.89$^{\approx}$	0.67†	0.88†	0.88†	0.88	0.85	0.58
Breast Cancer	0.96	0.96$^{\approx}$	0.96$^{\approx}$	0.96$^{\approx}$	0.96$^{\approx}$	0.96$^{\approx}$	0.96	**0.97**	0.96
Breast Tissue	**0.72**	0.70†	0.67†	0.70†	0.64†	0.71†	0.71	0.71	0.67
Diabetes	**0.75**	0.71†	0.72†	0.69†	0.73†	0.72†	0.74	0.69	0.67
Ecoli	**0.88**	0.85†	0.87$^{\approx}$	0.72†	0.82†	0.83†	0.86	0.80	0.80
Haberman	**0.75**	0.71†	0.72†	0.63†	0.70†	0.71†	0.74	0.71	0.56
Heart Statlog	**0.84**	0.81†	0.81†	0.80†	0.78†	0.77†	0.79	0.76	0.74
Hill Valley	**0.57**	0.55†	0.55†	0.55†	0.51†	0.53†	0.53	0.53	0.52
Ionosphere	0.93	0.86†	0.87†	0.83†	**0.95**†	0.87†	0.92	0.91	0.92
Ozone	**0.97**	0.96	0.96$^{\approx}$	0.90†	0.96$^{\approx}$	0.96$^{\approx}$	0.94	0.93	0.78
Pendigits	**0.99**	**0.99**$^{\approx}$	**0.99**$^{\approx}$	**0.99**$^{\approx}$	**0.99**$^{\approx}$	**0.99**$^{\approx}$	**0.99**	**0.99**	**0.99**
Postoperative	**0.74**	0.68†	0.70†	0.37†	0.66†	0.70†	0.69	0.64	0.44
Sonar	0.88	0.84†	0.83†	0.86†	0.84†	0.82†	0.86	**0.89**	0.85
Transfusion	**0.79**	**0.79**$^{\approx}$	0.53†	0.64†	0.74†	0.73†	0.77	0.68	0.64
Vehicle	**0.72**	0.65†	0.65†	0.67†	0.70†	0.69†	**0.72**	**0.72**	0.71
Vote	**0.96**	0.94$^{\approx}$	0.80†	0.94$^{\approx}$	0.93†	0.95$^{\approx}$	0.95	0.95	0.94
Vowel	0.98	0.97$^{\approx}$	0.95†	0.96†	0.98$^{\approx}$	**0.99**$^{\approx}$	0.98	**0.99**	**0.99**
Wine	**0.98**	0.96†	0.96†	0.96†	0.96†	0.94†	**0.98**	**0.98**	0.97
Yeast	**0.60**	0.56†	0.58†	0.50†	0.58†	0.58†	0.57	0.53	0.51

†: The performances are significantly different as indicated by the WRS.
≈: The performances are statistically comparable as indicated by the WRS.
*: For these classifiers the results are quoted from the original article hence WRS cannot be performed.
The best result is boldfaced.

Table C.5: Comparison of PIFW^2kNN on real-world class imbalanced benchmark datasets in terms of GMean.

Dataset	dyn$k$NN	CCNND	CWkNN	WkNN +GCW	FkNN +SMOTE	FkNN +RUS	PIFW^2kNN
Abalone19	0.00†	**0.67**†	0.00†	0.60†	0.45$^\approx$	0.00†	0.46
Acoustic	0.64†	0.51†	0.47†	0.50†	0.61†	0.37†	**0.71**
ADA	0.65†	0.64†	0.39†	**0.72**$^\approx$	**0.72**$^\approx$	0.67†	0.71
Biodeg	0.84†	0.74†	0.82†	0.84†	0.85$^\approx$	0.83†	**0.87**
BCWO	0.96$^\approx$	0.96$^\approx$	0.93†	0.65†	0.92†	0.95†	**0.98**
Chess	0.12†	0.21†	0.00†	0.24$^\approx$	0.08†	0.07†	**0.26**
CodRna	0.86†	0.87†	0.66†	0.88$^\approx$	0.89$^\approx$	0.89$^\approx$	**0.90**
Colon Cancer	0.55†	0.66†	0.20†	0.77†	0.74†	0.74†	**0.82**
Dermatology6	0.97†	0.94†	0.24†	**1.00**$^\approx$	**1.00**$^\approx$	**1.00**$^\approx$	**1.00**
German Numeric	0.60†	0.62†	0.12†	**0.67**$^\approx$	0.65$^\approx$	0.66$^\approx$	**0.67**
Glass7	0.86†	0.85†	0.83†	0.94†	0.92†	0.90†	**0.97**
Hepatitis1	0.64†	0.70†	0.05†	0.79†	0.74†	0.80†	**0.85**
Hypothyroid3	0.76†	0.92†	0.05†	0.93†	0.91†	0.80†	**0.95**
IJCNN1	0.73†	0.90	0.00†	0.89†	0.82†	0.73†	**0.92**
Liver Disorder	0.60†	0.60†	0.57†	0.33†	0.63†	0.62†	**0.73**
Magic	0.71†	0.57†	0.57†	0.76$^\approx$	0.75$^\approx$	0.74†	**0.77**
Shuttle	**0.99**$^\approx$	0.98$^\approx$	0.94$^\approx$	**0.99**$^\approx$	0.96$^\approx$	0.95†	0.98
SVM Guide2	0.59†	0.60†	0.09†	0.72†	0.59†	0.65†	**0.75**
SVM Guide4	0.46†	0.25†	0.10†	0.51†	0.45†	0.37†	**0.74**
SYLVA	0.74†	0.90†	0.00†	**0.95**$^\approx$	0.91†	0.68†	**0.95**

†: The performances are significantly different as indicated by the WRS.
≈: The performances are statistically comparable as indicated by the WRS.
The best result is boldfaced.

Table C.6: Comparison of PIFW^2kNN on real-world class imbalanced benchmark datasets in terms of ACSA.

Dataset	dyn$k$NN	CCNND	CWkNN	WkNN +GCW	FkNN +SMOTE	FkNN +RUS	PIFW^2kNN
Abalone19	0.50†	**0.68**†	0.50†	0.63†	0.58$^\approx$	0.50†	0.56
Acoustic	0.66†	0.66†	0.64†	0.60†	0.62†	0.68†	**0.72**
ADA	0.64†	0.66†	0.58†	**0.76**†	0.73$^\approx$	0.73$^\approx$	0.72
Biodeg	0.84†	0.81†	0.84†	0.84†	0.86$^\approx$	0.84†	**0.87**
BCWO	0.96$^\approx$	0.86†	0.93†	0.71†	0.92†	0.95$^\approx$	**0.98**
Chess	0.20†	0.36$^\approx$	0.24†	0.28†	**0.43**†	0.36$^\approx$	0.34
CodRNA	0.87$^\approx$	0.87$^\approx$	0.72†	**0.90**$^\approx$	0.89$^\approx$	0.89$^\approx$	**0.90**
Colon Cancer	0.66†	0.73†	0.55†	0.81†	0.77†	0.76†	**0.83**
Dermatology6	0.98†	0.95†	0.60†	**1.00**$^\approx$	**1.00**$^\approx$	**1.00**$^\approx$	**1.00**
German Numeric	0.64†	0.62†	0.51†	**0.67**$^\approx$	0.65$^\approx$	0.65$^\approx$	**0.67**
Glass7	0.87†	0.86†	0.85†	0.92†	0.93$^\approx$	0.91†	**0.95**
Hepatitis1	0.73†	0.72†	0.51†	0.80†	0.78†	0.81†	**0.85**
Hypothyroid3	0.80†	0.92†	0.51†	0.93†	0.91†	0.82†	**0.95**
IJCNN1	0.77†	0.89$^\approx$	0.50†	0.90$^\approx$	0.84†	0.78†	**0.91**
Liver Disorder	0.61†	0.62†	0.62†	0.51†	0.64†	0.63†	**0.73**
Magic	0.75†	0.65†	0.66†	0.77$^\approx$	0.77$^\approx$	0.75†	**0.79**
Shuttle	**0.99**$^\approx$	**0.99**$^\approx$	0.98$^\approx$	**0.99**$^\approx$	0.97$^\approx$	0.96$^\approx$	0.98
SVM Guide2	0.66†	0.62†	0.35†	0.73$^\approx$	0.67†	0.68†	**0.74**
SVM Guide4	0.63†	0.54†	0.46†	0.61†	0.63†	0.64†	**0.76**
SYLVA	0.78†	0.92†	0.50†	**0.95**$^\approx$	0.92†	0.79†	**0.95**

†: The performances are significantly different as indicated by the WRS.
≈: The performances are statistically comparable as indicated by the WRS.
The best result is boldfaced.

Appendix D

Supplementary for Chapter 5

D.1 Notes on imbalanced dataset creation

In this section we provide a couple of notes regarding our data creation process.

- Widely used image datasets such as MNIST, Fashion-MNIST, CIFAR-10, and SVHN are not considered as class imbalanced in regular practice. Thus, we created imbalanced variants of these dataset by randomly undersampling different classes in the training set as described in Section 5.3. However, as noted earlier in Section 1.2 the effect of class imbalance depends on multiple factors such as data distribution, data scale, and the choice of the classifier. Even though the impact of real world benchmark imbalanced datasets on classification performance are well documented the same cannot be said about the imbalanced variants of image datasets created by us. Thus, we empirically validate if our imbalanced image datasets are indeed capable to affect the Baseline CN classifier. We compare the performances of the Baseline CN respectively trained on the class imbalanced training set and their balanced counterparts in terms of GMean and ACSA on the same test set. From the following Table D.1 we can observe that the average performance over 10 runs of the Baseline CN trained on an imbalanced training set is always poorer than that trained on the corresponding balanced training set.

- During development we had experimented on 10 imbalanced variants of MNIST and Fashion-MNIST (retaining the class order in one, while randomly shuffling it in others). The performance of the algorithms remained consistent in terms of the average standard deviation across the variant datasets (0.02). Thus, for simplicity and ease of readability,

we had reported the results only on the ordered datasets.

Table D.1: The imbalanced datasets designed by us are indeed capable to affect a classifier's performance.

Datasets		Baseline CN	
		GMean	ACSA
MNIST	All classes contain 4000 training instances	**0.96**	**0.96**
	Classes contain training instances as described in Section 5.3.1	0.86	0.88
Fashion-MNIST	All classes 4000 contain training instances	**0.89**	**0.89**
	Classes contain training instances as described in Section 5.3.1	0.79	0.81
CIFAR-10	All classes contain 4500 training instances	**0.62**	**0.65**
	Classes contain training instances as described in Section 5.3.2	0.37	0.45
SVHN	All classes contain 4500 training instances	**0.84**	**0.85**
	Classes contain training instances as described in Section 5.3.2	0.73	0.74

D.2 Network architecture and hyperparameter selection

D.2.1 SMOTE

The number of intra-class neighbours are varied between $\{3, 5, 7\}$, and finally set to 5 which is found to be the best performer.

D.2.2 Common settings

For all of the networks the batch size is set to 32. For all the generators involved in different algorithms the latent dimension is taken as 100 (Mirza and Osindero, 2014). For GAMO using convolutional feature extraction the dimension of the feature space is taken as 512. The momentum parameter in batch normalization is set to 0.9, while the α in LeakyReLU is taken as 0.1 (Keras default settings). For convolutional layers the stride is 1, while that for deconvolution layers is set to 2 (for increasing the resolution of the image). We have used Adam (Kingma and Ba, 2015) optimizer in all cases, for which the β_2 parameter is set to 0.5. The latent dimension for GAMO2pix is also set to 100. The maximum number of steps for different algorithms and datasets are listed in the following Table D.2.

D.2.3 Augmentation

Data augmentation is performed using the "preprocessing" function of the "ImageDataGenerator" class available in "Keras" deep learning API. Table D.3 lists the different parameters used for augmentation along with their associated values.

D.2.4 GAMO network

The GAMO network architecture and hyperparameters, along with the grid search space and the final network is listed in Table D.4.

D.2.5 cGAN/cDCGAN network

The cGAN and cDCGAN network architecture and hyperparameters grid search space and the final network is listed in Table D.5.

D.2.6 Classifier network

The classifier network architecture and hyperparameters grid search space and the final network is listed in Table D.6.

D.2.7 DOS

The DOS network for a dataset is designed similarly to the baseline classifier network. The neighborhood size is set following the guideline of the original article (Ando and Huang, 2017).

Note

The Dense parts are kept similar throughout analogous networks as that particular architecture is found to be performing better on average over all algorithms, after the grid search.

D.2.8 GAMO2pix

The GAMO2pix network architecture and hyperparameters of the final network is listed in Table D.7.

Table D.2: List of maximum number of steps used by an algorithm on a dataset

Algorithm	Dataset	Maximum number of steps
Baseline CN	MNIST, Fashion-MNIST, CIFAR10, SVHN	50000
Baseline CN	CelebA, LSUN, SUN397	150000
SMOTE+CN	MNIST	Same as Baseline CN
Augment+CN	Fashion-MNIST, CIFAR10, SVHN, CelebA, LSUN, SUN397	Same as Baseline CN
cGAN+CN	MNIST	Same as Baseline CN
cG+CN	MNIST, Fashion-MNIST	Same as Baseline CN
cG+D+CN	MNIST, Fashion-MNIST	Same as Baseline CN
cDCGAN+CN	Fashion-MNIST, CIFAR10, SVHN, CelebA, LSUN, SUN397	Same as Baseline CN
DOS	Fashion-MNIST, CIFAR10, SVHN, CelebA, LSUN, SUN397	Same as Baseline CN
GAMO\D	MNIST, Fashion-MNIST, CIFAR10, SVHN, CelebA, LSUN, SUN397	Same as Baseline CN
GAMO	MNIST, Fashion-MNIST, CIFAR10, SVHN, CelebA, LSUN, SUN397	Same as Baseline CN
GAMO2pix	Fashion-MNIST, CIFAR10	25000
GAMO2pix	CelebA	50000

Table D.3: List of parameters along with their corresponding values chosen for augmenting the datasets.

Parameters	Fashion-MNIST CIFAR10 SVHN	CelebA LSUN SUN397
rotation_range	20	20
width_shift_range	0.2	0.2
height_shift_range	0.2	0.2
shear_range	0.2	0.2
zoom_range	0.2	0.2
brightness_range	$(0.1, 1)$	$(0.1, 1)$
fill_mode	nearest	nearest
horizontal_flip	False	True

The name and value of the parameters follow the convention of the standard "Keras" implementation.

Table D.4: Grid search space along with the selected network architecture and hyperparameter settings of GAMO framework.

Dataset	Parameters	Grid search space	Final network
MNIST	No. of layers in cTG	$\{2, 3, 4\}$	Dense, 256, ReLU Dense, 64, ReLU
	BN in cTG	{True, False}	True
	No. of layers in IGU_i	-	Dense, n_i, softmax
	No. of layers in D	$\{2, 3, 4\}$	Dense, 256, LeakyReLU Dense, 128, LeakyReLU Dense, 1, sigmoid
	No. of layers in CN	$\{2, 3, 4\}$	Dense, 256, LeakyReLU Dense, 128, LeakyReLU Dense, 10, softmax
Fashion-MNIST	No. of layers in C	$\{2, 3, 4\}$	5×5 Conv., 32, LeakyReLU 5×5 Conv., 32, LeakyReLU Dense, 512, tanh
	Average Pooling in C	{True, False}	True
	BN in cTG	{True, False}	True
	Other parameters	-	Identical to MNIST
CIFAR10	No. of layers in C	-	Similar to Fashion-MNIST
	Average Pooling in C	-	True
	No. of layers in cTG	$\{2, 3\}$	Dense, 256, ReLU Dense, 64, ReLU
	BN in cTG	-	True
	No. of layers in IGU_i	-	Dense, n_i, softmax
	No. of layers in D	$\{3, 4\}$	Dense, 256, LeakyReLU Dense, 128, LeakyReLU Dense, 1, sigmoid
	No. of layers in CN	$\{3, 4\}$	Dense, 256, LeakyReLU Dense, 128, LeakyReLU Dense, 10, softmax
CelebA	No. of layers in C	$\{4, 5, 6\}$	5×5, Conv., 32, LeakyReLU 5×5, Conv., 32, LeakyReLU 5×5, Conv., 32, LeakyReLU 5×5, Conv., 32, LeakyReLU Dense, 512, tanh
	Average Pooling in C	-	True
	No. of layers in cTG	$\{3, 4\}$	Dense, 256, ReLU Dense, 64, ReLU
	BN in cTG	-	True
	No. of layers in IGU_i	-	Dense, n_i, softmax
	No. of layers in D	$\{3, 4\}$	Dense, 256, LeakyReLU Dense, 128, LeakyReLU Dense, 1, sigmoid
	No. of layers in CN	$\{3, 4\}$	Dense, 256, LeakyReLU Dense, 128, LeakyReLU Dense, 5, softmax
Optimizer	Adam (β_1)	$\{0.0002, 0.002, 0.02\}$	0.0002

The β_2 parameter of Adam optimizer is set to 0.5 for all experiments.
If True then BN or Batch Normalization Ioffe and Szegedy (2015) is applied after every dense layer of cTG.
If True then 2×2 AveragePooling is applied after every convolution layer.
Due to the similar nature of the two datasets, the optimum architecture and parameter settings for CIFAR10 are also used for SVHN.
Due to the similar nature of the three datasets, the optimum architecture and parameter settings for CelebA are also used for LSUN, and SUN397.

Table D.5: Grid search space along with the selected network architecture and hyperparameter settings of the cGAN/cDCGAN network.

Dataset	Parameters	Grid search space	Final network
MNIST	No. of layers in Gen.	$\{3, 4\}$	Dense, 128, ReLU Dense, 256, ReLU Dense, 784, tanh
	No. of layers in Dis.	$\{3, 4\}$	Dense, 256, LeakyReLU Dense, 128, LeakyReLU Dense, 1, sigmoid
Fashion-MNIST	No. of layers in Gen.	$\{4, 5, 6, 7, 8\}$	Dense, 6272, LeakyReLU 4×4 Conv., 128, LeakyReLU 4×4 Deconv., 128, LeakyReLU 4×4 Conv., 128, LeakyReLU 4×4 Deconv., 128, LeakyReLU 5×5 Conv., 128, LeakyReLU 5×5 Conv., 1, tanh
	BN in Gen.	{True, False}	True
	Average Pooling in Gen.	{True, False}	False
	No. of layers in Dis.	$\{5, 6\}$	5×5 Conv., 32, LeakyReLU 5×5 Conv., 32, LeakyReLU Dense, 256, LeakyReLU Dense, 128, LeakyReLU Dense, 1, sigmoid
	Average Pooling in Dis.	{True, False}	True
CIFAR10	No. of layers in Gen.	$\{4, 5, 6, 7, 8\}$	Dense, 512, LeakyReLU 4×4, Deconv., 32, LeakyReLU 4×4, Deconv., 32, LeakyReLU 5×5, Deconv., 3, tanh
	BN in Gen.	-	True
	Average Pooling in Gen.	-	False
	No. of layers in Dis.	$\{5, 6\}$	5×5 Conv., 32, LeakyReLU 5×5 Conv., 32, LeakyReLU Dense, 256, LeakyReLU Dense, 128, LeakyReLU Dense, 1, sigmoid
	Average Pooling in Dis.	-	True
CelebA	No. of layers in Gen.	$\{4, 5, 6, 7, 8\}$	Dense, 2048, LeakyReLU 4×4, Deconv., 64, LeakyReLU 4×4, Deconv., 64, LeakyReLU 4×4, Deconv., 64, LeakyReLU 4×4, Deconv., 3, tanh
	BN in Gen.	-	True
	Average Pooling in Gen.	-	False
	No. layers in Dis.	$\{6, 7, 8\}$	5×5, Conv., 32, LeakyReLU 5×5, Conv., 32, LeakyReLU 5×5, Conv., 32, LeakyReLU 5×5, Conv., 32, LeakyReLU Dense, 256, LeakyReLU Dense, 128, LeakyReLU Dense, 1, sigmoid
	Average Pooling in Dis.	-	True
Optimizer	Adam (β_1)	$\{0.0002, 0.002, 0.02\}$	0.0002

Gen. and Dis. respectively stands for Generator and Discriminator. The β_2 parameter of Adam optimizer is set to 0.5 for all experiments. If True then BN or Batch Normalization (Ioffe and Szegedy, 2015) is applied after every dense layer of conditional generator. In case of MNIST and Fashion-MNIST the generator used in cG+CN, and cG+D+CN has similar architecture to that of cGAN. If True then 2×2 AveragePooling is applied after every convolution layer. Due to the similar nature of the two datasets, the optimum architecture and parameter settings for CIFAR10 are also used for SVHN. Due to the similar nature of the three datasets, the optimum architecture and parameter settings for CelebA are also used for LSUN, and SUN397.

Table D.6: Grid search space along with the selected network architecture and hyperparameter settings of the classifier network.

Dataset	Parameters	Grid search space	Final network
MNIST	No. of Dense layers	$\{2, 3, 4\}$	Dense, 256, LeakyReLU Dense, 128, LeakyReLU Dense, 10, softmax
Fashion-MNIST	No. of Conv. layers	$\{2, 3, 4\}$	5×5 Conv., 32, LeakyReLU 5×5 Conv., 32, LeakyReLU
	Average Pooling	{True, False}	True
	Other parameters	-	Identical to MNIST
CIFAR10	No. of Conv. layers	$\{2, 3, 4\}$	5×5, Conv., 32, LeakyReLU 5×5, Conv., 32, LeakyReLU
	Average Pooling	{True, False}	True
	No. of Dense layers	$\{2, 3, 4\}$	Dense, 256, LeakyReLU Dense, 128, LeakyReLU Dense, 10, softmax
CelebA	No. of Conv. layers	$\{3, 4, 5\}$	5×5, Conv., 32, LeakyReLU 5×5, Conv., 32, LeakyReLU 5×5, Conv., 32, LeakyReLU 5×5, Conv., 32, LeakyReLU
	Average Pooling	-	True
	No. of Dense layers	$\{2, 3, 4\}$	Dense, 256, LeakyReLU Dense, 128, LeakyReLU Dense, 5, softmax
Optimizer	Adam (β_1)	$\{0.0002, 0.002, 0.02\}$	0.0002

The β_2 parameter of Adam optimizer is set to 0.5 for all experiments.
If True then 2×2 AveragePooling is applied after every convolution layer.
Due to the similar nature of the two datasets, the optimum architecture and parameter settings for CIFAR10 are also used for SVHN.
Due to the similar nature of the three datasets, the optimum architecture and parameter settings for CelebA are also used for LSUN, and SUN397.

Table D.7: Architecture and hyperparameter settings of the GAMO2pix network.

Dataset	Parameters	Final network
Fashion-MNIST	Mean layer	Dense, 100
	Log Variance layer	Dense, 100
	Decoder	Dense, 1568, LeakyReLU
		4×4, Deconv., 32, LeakyReLU
		4×4, Deconv., 32, LeakyReLU
		4×4, Conv., 1, tanh
CIFAR10	Decoder	Dense, 512, LeakyReLU
		4×4, Deconv., 32, LeakyReLU
		4×4, Deconv., 32, LeakyReLU
		4×4, Deconv., 32, LeakyReLU
		4×4, Conv., 3, tanh
CelebA	Decoder	Dense, 512, LeakyReLU
		4×4, Deconv., 32, LeakyReLU
		4×4, Deconv., 32, LeakyReLU
		4×4, Deconv., 32, LeakyReLU
		4×4, Deconv., 32, LeakyReLU
		4×4, Conv., 3, tanh
Optimizer	Adam (β_1)	0.0002

For all the datasets, the corresponding feature extractor network trained by GAMO is connected to the Mean and Log Variance layer. Among all the components of GAMO2pix only the feature extractor is kept fixed throughout the training period.
The β_2 parameter of Adam optimizer is set to 0.5 for all experiments.
Due to the similar nature of the two datasets, the optimum architecture and parameter settings for CIFAR10 are also used for SVHN.
The Mean layer and the Log Variance layer is kept similar to Fashion-MNIST for CIFAR10, SVHN, and CelebA.

D.3 Results on non-image class imbalanced benchmark datasets

The five non-image datasets used for additionally validating the performance of GAMO are collected from University of California, Irvine, Machine Learning Repository (Dua and Graff, 2017), KEEL (Triguero et al., 2011), and LibSVM (Chang and Lin, 2011). The different properties of these datasets are detailed in Table D.8. The results of the comparison between GAMO, Baseline CN, cGAN+CN, SMOTE+CN, and GAMO\D are summarized in the following Table D.9. We have used a 10-fold stratified cross-validation and report the mean ACSA and GM. The best ACSA and GM among the contenders are boldfaced for each dataset. It is evident from Table D.9 that GAMO on average can retain its commendable performance on non-image datasets as well. Interestingly, the average performance of SMOTE+CN in terms of ACSA is close to that of GAMO, justifying the popularity of SMOTE over the past couple of decades. However, compared to SMOTE+CN, the proposed GAMO shows a better consistency over all the classes as indicated by the higher average GMean (γ_c).

Table D.8: Detailed description of the datasets.

Dataset name	Number of points	Number of dimensions	Number of classes	IR
Abalone19	4177	8	2	129.5
Chess	28056	6	18	168.6
Cover Type	581012	54	7	103.13
IJCNN1	141691	22	2	9.4
Magic	19020	10	2	1.8

Table D.9: Results on non-image class imbalanced benchmark datasets.

Datasets	Baseline CN		SMOTE+CN		cGAN+CN		GAMO\D		GAMO	
	ACSA	γ_c	ACSA	γ_c	ACSA	γ_c	ACSA	γ_c	ACSA	γ_c
Abalone19	0.50	0.00	0.58	**0.48**	**0.59**	**0.48**	0.46	0.00	**0.59**	**0.48**
Chess	0.29	0.00	0.30	0.00	0.25	0.00	0.16	0.00	**0.32**	0.20
Cover Type	0.51	0.43	**0.70**	**0.64**	0.57	0.47	0.46	0.31	0.66	**0.64**
IJCNN1	0.91	0.90	0.93	0.93	0.93	0.93	0.89	0.88	**0.95**	**0.95**
Magic	0.82	0.82	0.81	0.81	0.82	0.82	0.73	0.72	**0.83**	**0.83**
Avg. Performance	0.60	0.43	0.66	0.57	0.63	0.54	0.54	0.38	**0.67**	**0.62**

List of Publications

I. Banerjee, S. S. Mullick, and S. Das. On convergence of the class membership estimator in fuzzy k-nearest neighbor classifier. *IEEE Transactions on Fuzzy Systems*, 27(6):1226–1236, 2019.

N. Biswas, S. Chakraborty, S. S. Mullick, and S. Das. A parameter independent fuzzy weighted k-nearest neighbor classifier. *Pattern Recognition Letters*, 101:80–87, 2018.

S. Das and S. S. Mullick. On classification of imbalanced datasets: An updated perspective and some future challenges. *Submitted to ACM Computing Surveys*, 2020.

S. S. Mullick, S. Datta, and S. Das. Generative adversarial minority oversampling. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1695–1704, 2019.

S. S. Mullick, S. Datta, S. G. Dhekane, and S. Das. Appropriateness of performance indices for imbalanced data classification: An analysis. *Pattern Recognition*, 102:107197, 2020. doi: https://doi.org/10.1016/j.patcog.2020.107197.

References

L. Abdi and S. Hashemi. To combat multi-class imbalanced problems by means of over-sampling techniques. *IEEE transactions on Knowledge and Data Engineering*, 28(1):238–251, 2015. 17

S. Afzal, M. Maqsood, F. Nazir, U. Khan, F. Aadil, K. M. Awan, I. Mehmood, and O.-Y. Song. A data augmentation-based framework to handle class imbalance problem for alzheimers stage detection. *IEEE Access*, 7:115528–115539, 2019. 28

C. C. Aggarwal, A. Hinneburg, and D. A. Keim. On the surprising behavior of distance metrics in high dimensional space. In *International conference on database theory*, pages 420–434. Springer, 2001. 17

R. Akbani, S. Kwek, and N. Japkowicz. Applying support vector machines to imbalanced datasets. In *Machine Learning: ECML 2004*, pages 39–50, 2004. 13, 25, 96, 117

R. Alaiz-Rodríguez and N. Japkowicz. Assessing the impact of changing environments on classifier performance. In *Conference of the Canadian Society for Computational Studies of Intelligence*, pages 13–24. Springer, 2008. 40, 41

R. Alejo, V. García, J. M. Sotoca, R. A. Mollineda, and J. S. Sánchez. Improving the performance of the rbf neural networks trained with imbalanced samples. In *International work-conference on artificial neural networks*, pages 162–169. Springer, 2007. 19

R. Alejo, R. M. Valdovinos, V. García, and J. H. Pacheco-Sanchez. A hybrid method to face class overlap and class imbalance on neural networks and multi-class scenarios. *Pattern Recognition Letters*, 34(4):380–388, 2013. 7

A. Ali-Gombe and E. Elyan. Mfc-gan: Class-imbalanced dataset classification using multiple fake class generative adversarial network. *Neurocomputing*, 361:212 – 221, 2019. 28, 123

A. AlSukker, R. Khushaba, and A. Al-Ani. Optimizing the knn metric weights using differential evolution. In *Multimedia Computing and Information Technology (MCIT)*, pages 89–92, 2010. 105, 107, 117

S.-I. Amari and S. Wu. Improving support vector machine classifiers by modifying kernel functions. *Neural Networks*, 12(6):783–789, 1999. 21

O. Anava and K. Levy. k^*-nearest neighbors: From global to local. In *Advances in Neural Information Processing Systems 29*, pages 4916–4924, 2016. 71, 75, 77, 150

S. Ando and C. Y. Huang. Deep over-sampling framework for classifying imbalanced data. In *Machine Learning and Knowledge Discovery in Databases*, pages 770–785. Springer International Publishing, 2017. 28, 123, 127, 148, 179

F. Angiulli and F. Fassetti. Nearest neighbor-based classification of uncertain data. *ACM Transactions on Knowledge Discovery from Data*, 7(1):1:1–1:35, 2013. 102

T. M. Apostol. *Mathematical analysis*. Addison-Wesley, 1964. 80

Ö. F. Arar and K. Ayan. Software defect prediction using cost-sensitive neural network. *Applied Soft Computing*, 33:263–277, 2015. 19

M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017. 28

A. Aşkan and S. Sayın. Svm classification for imbalanced data sets using a multiobjective optimization framework. *Annals of Operations Research*, 216(1):191–203, 2014. 12, 22

J. Attenberg and S. Ertekin. Class imbalance and active learning. *Imbalanced Learning: Foundations, Algorithms, and Applications*, pages 101–149, 2013. 21

J. Attenberg and F. Provost. Why label when you can search? alternatives to active learning for applying human resources to build classification models under extreme class imbalance. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 423–432, 2010. 21

N. H. Awad, M. Z. Ali, J. J. Liang, B. Y. Qu, and P. N. Suganthan. Problem definitions and evaluation criteria for the cec 2017 special session and competition on single objective bound constrained real-parameter numerical optimization. Technical report, NTU, Singapore, 2016. 109, 111

D. Ballabio, F. Grisoni, and R. Todeschini. Multivariate comparison of classification performance measures. *Chemometrics and Intelligent Laboratory Systems*, 174:33 – 44, 2018. 30, 39

D. H. Ballard. Modular learning in neural networks. In *AAAI*, pages 279–284, 1987. 124

L. Bao, C. Juan, J. Li, and Y. Zhang. Boosted near-miss under-sampling on svm ensembles for concept detection in large-scale imbalanced datasets. *Neurocomputing*, 172:198 – 206, 2016. 24

R. Barandela, R. M. Valdovinos, and J. S. Sánchez. New applications of ensembles of classifiers. *Pattern Analysis & Applications*, 6(3):245–256, 2003. 24

S. Barua, M. M. Islam, X. Yao, and K. Murase. MWMOTE–majority weighted minority oversampling technique for imbalanced data set learning. *IEEE Transactions on Knowledge and Data Engineering*, 26(2):405–425, 2014. 16, 124

G. Batista, D. F. Silva, and R. Prati. An experimental design to evaluate class imbalance treatment methods. In *2012 11th International Conference on Machine Learning and Applications*, volume 2, pages 95–101. IEEE, 2012. 15

G. E. Batista, R. C. Prati, and M. C. Monard. A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD explorations newsletter*, 6 (1):20–29, 2004. 16

E. Bax. Validation of nearest neighbor classifiers. *IEEE Transactions on Information Theory*, 46(7):2746–2752, 2000. 74

E. Bax. Validation of k-nearest neighbor classifiers. *IEEE Transactions on Information Theory*, 58(5):3225–3234, 2012. 74, 150

C. Bellinger, S. Sharma, and N. Japkowicz. One-class versus binary classification: Which and when? In *2012 11th International Conference on Machine Learning and Applications*, volume 2, pages 102–106. IEEE, 2012. 20

C. Bellinger, N. Japkowicz, and C. Drummond. Synthetic oversampling for advanced radioactive threat detection. In *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pages 948–953. IEEE, 2015. 17

C. Bellinger, S. Sharma, O. R. Zaıane, and N. Japkowicz. Sampling a longer life: Binary versus one-class classification revisited. In *First International Workshop on Learning with Imbalanced Domains: Theory and Applications*, pages 64–78, 2017. 20

D. Berthelot, T. Schumm, and L. Metz. BEGAN: boundary equilibrium generative adversarial networks. *arXiv preprint arXiv:1703.10717*, 2017. 150

A. Bertoni, M. Frasca, and G. Valentini. Cosnet: a cost sensitive neural network for semi-supervised learning in graphs. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 219–234. Springer, 2011. 19

G. Bhattacharya, K. Ghosh, and A. S. Chowdhury. An affinity-based new local distance function and similarity measure for knn algorithm. *Pattern Recongnition Letters*, 33:356–363, 2012. 102

U. Bhowan, M. Johnston, M. Zhang, and X. Yao. Evolving diverse ensembles using genetic programming for classification with unbalanced data. *IEEE Transactions on Evolutionary Computation*, 17(3):368–386, 2013. 22

U. Bhowan, M. Johnston, M. Zhang, and X. Yao. Reusing genetic programming for ensemble selection in classification of unbalanced data. *IEEE Transactions on Evolutionary Computation*, 18(6):893–908, 2014. 22

S. K. Biswas, M. Chakraborty, H. R. Singh, D. Devi, B. Purkayastha, and A. K. Das. Hybrid case-based reasoning system by cost-sensitive neural network for classification. *Soft Computing*, 21(24):7579–7596, 2017. 19

J. Blaszczyński and J. Stefanowski. Neighbourhood sampling in bagging for imbalanced data. *Neurocomputing*, 150:529 – 542, 2015. 24

J. Błaszczyński, M. Deckert, J. Stefanowski, and S. Wilk. Integrating selective pre-processing of imbalanced data with ivotes ensemble. In *International conference on rough sets and current trends in computing*, pages 148–157. Springer, 2010. 18

K. Boonchuay, K. Sinapiromsaran, and C. Lursinsap. Decision tree induction based on minority entropy for the class imbalance problem. *Pattern Analysis and Applications*, 20(3):769–782, 2017. 19

A. P. Bradley, R. P. W. Duin, P. Paclik, and T. C. W. Landgrebe. Precision-recall operating characteristic (p-roc) curves in imprecise environments. In *18th International Conference on Pattern Recognition (ICPR'06)*, volume 4, pages 123–127, 2006. 30, 40, 45, 46, 52

P. Branco, L. Torgo, and R. P. Ribeiro. A survey of predictive modeling on imbalanced domains. *ACM Computing Surveys (CSUR)*, 49(2):31, 2016. 6, 38, 86, 122, 134

P. Brazdil, C. Giraud-Carrier, C. Soares, and R. Vilalta. *Metalearning: Applications to Data Mining*. Springer Publishing Company, Incorporated, 1 edition, 2008. ISBN 3540732624. 14

L. Breiman. *Classification and regression trees*. Routledge, 2017. 63

D. Brzezinski, J. Stefanowski, R. Susmaga, and I. Szczech. Visual-based analysis of classification measures and their properties for class imbalanced problems. *Information Sciences*, 462:242 – 261, 2018. 30, 40

D. Brzezinski, J. Stefanowski, R. Susmaga, and I. Szczech. On the dynamics of classification measures for imbalanced and streaming data. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–11, 2019. 40, 41, 153

M. Buckland and F. Gey. The relationship between recall and precision. *Journal of the American society for information science*, 45(1):12, 1994. 38, 39

M. Buda, A. Maki, and M. A. Mazurowski. A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks*, 106:249–259, 2018. 25, 123

S. R. Bulò, G. Neuhold, and P. Kontschieder. Loss max-pooling for semantic image segmentation. In *Proceedings of the IEEE Conference on CVPR*, pages 2126–2135, 2017. 4, 123

C. Bunkhumpornpat and K. Sinapiromsaran. Dbmute: density-based majority undersampling technique. *Knowledge and Information Systems*, 50(3):827–850, 2017. 14

C. Bunkhumpornpat, K. Sinapiromsaran, and C. Lursinsap. Safe-Level-SMOTE: safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem. In *Advances in Knowledge Discovery and Data Mining*, pages 475–482, 2009. 16, 124

C. Bunkhumpornpat, K. Sinapiromsaran, and C. Lursinsap. Dbsmote: density-based synthetic minority over-sampling technique. *Applied Intelligence*, 36(3):664–684, 2012. 16

K. Cao, C. Wei, A. Gaidon, N. Arechiga, and T. Ma. Learning imbalanced datasets with label-distribution-aware margin loss. In *Advances in Neural Information Processing Systems*, pages 1567–1578, 2019. 29

P. Cao, D. Zhao, and O. Zaiane. An optimized cost-sensitive svm for imbalanced data learning. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 280–292. Springer, 2013a. 19

P. Cao, D. Zhao, and O. R. Zaïane. A pso-based cost-sensitive neural network for imbalanced data classification. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 452–463. Springer, 2013b. 19

M.-A. Carbonneau, V. Cheplygina, E. Granger, and G. Gagnon. Multiple instance learning: A survey of problem characteristics and applications. *Pattern Recognition*, 77:329–353, 2018. 149

C. L. Castro and A. P. Braga. Novel cost-sensitive approach to improve the multilayer perceptron performance on imbalanced data. *IEEE transactions on neural networks and learning systems*, 24(6):888–899, 2013. 19

J. Cervantes, X. Li, and W. Yu. Imbalanced data classification via support vector machines and genetic algorithms. *Connection Science*, 26(4):335–348, 2014. 17

G. Chandrashekar and F. Sahin. A survey on feature selection methods. *Computers & Electrical Engineering*, 40(1):16 – 28, 2014. 103

C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. 97, 117, 185

N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002. 12, 15, 63, 87, 97, 123, 124, 127

N. V. Chawla, A. Lazarevic, L. O. Hall, and K. W. Bowyer. SMOTEBoost: improving prediction of the minority class in boosting. In *Knowledge Discovery in Databases: PKDD 2003*, pages 107–119, 2003. 13, 24, 124

S. Chen, H. He, and E. A. Garcia. Ramoboost: Ranked minority oversampling in boosting. *IEEE Transactions on Neural Networks*, 21(10):1624–1642, 2010. 24

X. Chen and Y. Gu. Class-specific feature selection with local geometric structure and discriminative information based on sparse similar samples. *IEEE Geoscience and Remote Sensing Letters*, 12(7):1392–1396, 2015. 104

K. Cheng, C. Zhang, H. Yu, X. Yang, H. Zou, and S. Gao. Grouped smote with noise filtering mechanism for classifying imbalanced data. *IEEE Access*, 7:170668–170681, 2019. 16

C. Chira and C. Lemnaru. A multi-objective evolutionary approach to imbalanced classification problems. In *2015 IEEE International Conference on Intelligent Computer Communication and Processing (ICCP)*, pages 149–154. IEEE, 2015. 22

Y.-A. Chung, H.-T. Lin, and S.-W. Yang. Cost-aware pre-training for multiclass cost-sensitive deep learning. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, pages 1411–1417. AAAI Press, 2016. 123

D. A. Cieslak and N. V. Chawla. Learning decision trees for unbalanced data. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 241–256. Springer, 2008. 20

D. A. Cieslak, T. R. Hoens, N. V. Chawla, and W. P. Kegelmeyer. Hellinger distance decision trees are robust and skew-insensitive. *Data Mining and Knowledge Discovery*, 24(1):136–158, 2012. 24

C. Cortes and V. Vapnik. Support vector networks. *Machine Leraning*, 20(3):273–297, 1995. 2, 7

T. Cover. Estimation by the nearest neighbor rule. *IEEE Transactions on Information Theory*, 14(1):50–55, 1968. 73, 74

T. M. Cover and P. E. Hart. Nearest neighbour pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, 1967. 73, 76, 78, 102

Y. Cui, M. Jia, T.-Y. Lin, Y. Song, and S. Belongie. Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9268–9277, 2019. 29, 152

A. D'Addabbo and R. Maglietta. Parallel selective sampling method for imbalanced and large data classification. *Pattern Recognition Letters*, 62:61–67, 2015. 14

B. Das, N. C. Krishnan, and D. J. Cook. Racog and wracog: Two probabilistic oversampling techniques. *IEEE Transactions on Knowledge and Data Engineering*, 27(1):222–234, 2015. 17

S. Das and P. N. Suganthan. Differential evolution: a survey of the state-of-the-art. *IEEE transactions on evolutionary computation*, 15(1):4–31, 2011. 35, 112

S. Das, S. S. Mullick, and P. N. Suganthan. Recent advances in differential evolution–an updated survey. *Swarm and Evolutionary Computation (SWEVO)*, 27:1–30, 2016. 35, 109, 110, 112

S. Das, S. Datta, and B. B. Chaudhuri. Handling data irregularities in classification: Foundations, trends, and future challenges. *Pattern Recognition*, 81:674–693, 2018. 3, 7, 11, 86

S. Daskalaki, I. Kopanas, and N. Avouris. Evaluation of classifiers for an uneven class distribution problem. *Applied artificial intelligence*, 20(5):381–417, 2006. 39

S. Datta and S. Das. Near-bayesian support vector machines for imbalanced data classification with equal or unequal misclassification costs. *Neural Networks*, 70:39 – 52, 2015. 19, 63

S. Datta and S. Das. Multiobjective support vector machines: Handling class imbalance with pareto optimality. *IEEE transactions on neural networks and learning systems*, 2018. 4

S. Datta, S. Nag, and S. Das. Boosting with lexicographic programming: Addressing class imbalance without cost tuning. *IEEE Transactions on Knowledge and Data Engineering*, 2019. 62, 63

J. Davis and M. Goadrich. The relationship between Precision-Recall and ROC curves. In *Proceedings of the 23rd international conference on Machine learning*, pages 233–240, 2006. 38, 39

R. F. de Morais and G. C. Vasconcelos. Boosting the performance of over-sampling algorithms through under-sampling the minority class. *Neurocomputing*, 343:3 – 18, 2019. 18

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *IEEE CVPR09*, pages 248–255, 2009. 41, 46, 62

J. Derrac, S. García, D. Molina, and F. Herrera. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *SWEVO*, 1(1):3 – 18, 2011. 107

J. Derrac, S. García, and F. Herrera. Fuzzy nearest neighbor algorithms: Taxonomy, experimental analysis and prospects. *Information Science*, 260:98–119, 2014. 71, 103

J. Derrac, F. Chiclana, S. García, and F. Herrera. Evolutionary fuzzy k-nearest neighbors algorithm using interval-valued fuzzy sets. *Information Sciences*, 329:144–163, 2016. 72, 107, 117

D. Devi, S. Biswas, and B. Purkayastha. Redundancy-driven modified tomek-link based undersampling: a solution to class imbalance. *Pattern Recognition Letters*, 93:3–12, 2017. 14

C. Domeniconi, J. Peng, and D. Gunopulos. Locally adaptive metric nearest-neighbor classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(9):1281–1285, 2002. 102

Q. Dong, S. Gong, and X. Zhu. Imbalanced deep learning by minority class incremental rectification. *IEEE transactions on pattern analysis and machine intelligence*, 41(6):1367–1381, 2018. 29, 123

G. Douzas and F. Bacao. Effective data generation for imbalanced learning using conditional generative adversarial networks. *Expert Systems with applications*, 91:464–471, 2018. 28, 123, 127

G. Douzas and F. Bacao. Geometric smote a geometrically enhanced drop-in replacement for smote. *Information Sciences*, 501:118–135, 2019. 16

G. Douzas, F. Bacao, and F. Last. Improving imbalanced learning through a heuristic oversampling method based on k-means and smote. *Information Sciences*, 465:1–20, 2018. 16

C. Drummond and R. C. Holte. Exploiting the cost (in) sensitivity of decision tree splitting criteria. In *ICML*, 2000. 19

D. Dua and C. Graff. UCI machine learning repository, 2017. URL `http://archive.ics.uci.edu/ml`. 91, 96, 116, 185

W. Duan, L. Jing, and X. Y. Lu. Imbalanced data classification using cost-sensitive support vector machine based on information entropy. In *Advanced Materials Research*, volume 989, pages 1756–1761. Trans Tech Publ, 2014. 19

H. Dubey and V. Pudi. Class based weighted k-nearest neighbor over imbalance dataset. In *Advances in Knowledge Discovery and Data Mining*, pages 305–316. Springer Berlin Heidelberg, 2013. 20, 74, 118

P. Ducange, B. Lazzerini, and F. Marcelloni. Multi-objective genetic fuzzy classifiers for imbalanced and cost-sensitive datasets. *Soft Computing*, 14(7):713–728, 2010. 22

P. Duda, M. Jaworski, and L. Rutkowski. On ensemble components selection in data streams scenario with reoccurring concept-drift. In *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–7. IEEE, 2017. 30

R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley, 2 edition, 2000. 2, 14, 71, 102

D. Elreedy and A. F. Atiya. A comprehensive analysis of synthetic minority oversampling technique (smote) for handling class imbalance. *Information Sciences*, 505:32–64, 2019. 15

S. Ertekin, J. Huang, L. Bottou, and L. Giles. Learning on the border: active learning in imbalanced data classification. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 127–136, 2007a. 12, 21

S. Ertekin, J. Huang, and C. L. Giles. Active learning for class imbalance problem. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 823–824, 2007b. 21

L. J. Eshelman. The chc adaptive search algorithm: How to have safe search when engaging in nontraditional genetic recombination. In *Foundations of Genetic Algorithms*, pages 265–283, 1991. 117

M. Ester, H.-P. Kriegel, J. Sander, X. Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231, 1996. 2, 14

S. Ezghari, A. Zahi, and K. Zenkouar. A new nearest neighbor classification method based on fuzzy set theory and aggregation operators. *Expert Systems with Applications*, 80:58–74, 2017. 72

W. Fan, S. J. Stolfo, J. Zhang, and P. K. Chan. Adacost: misclassification cost-sensitive boosting. In *Icml*, volume 99, pages 97–105, 1999. 23

W. Fan, F. Chu, H. Wang, and P. S. Yu. Pruning and dynamic scheduling of cost-sensitive ensembles. In *AAAI/IAAI*, pages 146–151, 2002. 23

Z. Ferdowsi, R. Ghani, and R. Settimi. Online active learning with imbalanced classes. In *2013 IEEE 13th International Conference on Data Mining*, pages 1043–1048. IEEE, 2013. 21

E. R. Fernandes and A. C. de Carvalho. Evolutionary inversion of class distribution in overlapping areas for multi-class imbalanced learning. *Information Sciences*, 494:141–154, 2019. 18

A. Fernández, M. J. del Jesus, and F. Herrera. Addressing overlapping in classification with imbalanced datasets: A first multi-objective approach for feature and instance selection. In *International Conference on Intelligent Data Engineering and Automated Learning*, pages 36–44. Springer, 2015a. 22

A. Fernández, M. Galar, J. A. Sanz, H. Bustince, O. Cordón, and F. Herrera. On the impact of distance-based relative competence weighting approach in one-vs-one classification for evolutionary fuzzy systems: Drcw-fh-gbml algorithm. In *2015 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 1–7. IEEE, 2015b. 104

A. Fernández, S. del Río, N. V. Chawla, and F. Herrera. An insight into imbalanced big data classification: outcomes and challenges. *Complex & Intelligent Systems*, 3(2):105–120, 2017. 9

A. Fernández, S. Garcia, F. Herrera, and N. V. Chawla. SMOTE for learning from imbalanced data: Progress and challenges, marking the 15-year anniversary. *Journal of Artificial Intelligence Research*, 61:863–905, 2018. 15, 123

C. Ferri, J. Hernndez-Orallo, and R. Modroiu. An experimental comparison of performance measures for classification. *Pattern Recognition Letters*, 30(1):27 – 38, 2009. 39

R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7:179–188, 1936. 103

E. Fix and J. L. Hodges Jr. Discriminatory analysis-nonparametric discrimination: consistency properties. Technical report, California Univ Berkeley, 1951. 2, 7, 30, 71

Y. Freund, R. Schapire, and N. Abe. A short introduction to boosting. *Journal-Japanese Society For Artificial Intelligence*, 14(771-780):1612, 1999. 23

H. Frigui and P. Gader. Detection and discrimination of land mines in ground-penetrating radar based on edge histogram descriptors and a possibilistic k-nearest neighbor classifier. *IEEE Transactions on Fuzzy Systems*, 17(1):185–199, 2009. 72, 149

Y. Fu, H. Zhang, Y. Bai, and W. Sun. An under-sampling method: Based on principal component analysis and comprehensive evaluation model. In *2016 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C)*, pages 414–415. IEEE, 2016. 14

M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera. A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42 (4):463–484, 2011. 24

M. Galar, A. Fernndez, E. Barrenechea, and F. Herrera. Eusboost: Enhancing ensembles for highly imbalanced data-sets by evolutionary undersampling. *Pattern Recognition*, 46(12): 3460 – 3471, 2013. 24

A.-J. Gallego, J. Calvo-Zaragoza, J. J. Valero-Mas, and J. R. Rico-Juan. Clustering-based k-nearest neighbor classification for large-scale data with neural codes representation. *Pattern Recognition*, 74:531 – 543, 2018. 71

F. Gao, W. Wang, M. Tan, L. Zhu, Y. Zhang, E. Fessler, L. Vermeulen, and X. Wang. Deepcc: a novel deep learning-based framework for cancer molecular subtype classification. *Oncogenesis*, 8(9):1–12, 2019. 152

S. García and F. Herrera. Evolutionary undersampling for classification with imbalanced datasets: Proposals and taxonomy. *Evolutionary computation*, 17(3):275–306, 2009. 14

S. García, J. R. Cano, A. Fernández, and F. Herrera. A proposal of evolutionary prototype selection for class imbalance problems. In *International Conference on Intelligent Data Engineering and Automated Learning*, pages 1415–1423. Springer, 2006a. 14

S. García, R. Aler, and I. M. Galván. Using evolutionary multiobjective techniques for imbalanced classification data. In *International Conference on Artificial Neural Networks*, pages 422–427. Springer, 2010. 22

S. García, A. Fernández, J. Luengo, and F. Herrera. Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. *Information Sciences*, 180:2044–2064, 2013. 92

V. García, R. Alejo, J. S. Sánchez, J. M. Sotoca, and R. A. Mollineda. Combined effects of class imbalance and class overlap on instance-based classification. In *International Conference on Intelligent Data Engineering and Automated Learning*, pages 371–378, 2006b. 7

N. Garcia-Pedrajas, J. A. D. Castillo, and G. Cerruela-Garcia. A proposal for local k values for k-nearest neighbor rule. *IEEE Transactions on Neural Network and Learning Systems*, 28(2):470–475, 2015. 71, 75, 118

A. Ghazikhani, R. Monsefi, and H. Sadoghi Yazdi. Ensemble of online neural networks for non-stationary and imbalanced data streams. *Neurocomputing*, 122:535 – 544, 2013a. 23

A. Ghazikhani, R. Monsefi, and H. S. Yazdi. Online cost-sensitive neural network classifiers for non-stationary and imbalanced data streams. *Neural computing and applications*, 23 (5):1283–1295, 2013b. 19

A. Ghosh, S. Das, S. S. Mullick, R. Mallipeddi, and A. K. Das. A switched parameter differential evolution with optional blending crossover for scalable numerical optimization. *Applied Soft Computing*, 57:329–352, 2017. 112, 150

A. K. Ghosh. On optimum choice of k in nearest neighbor classification. *Computational Statistics and Data Analysis*, 50:3113–3123, 2006. 102

J. D. Gibbson and S. Chakraborti. *Nonparametric Statistical Inference*. CRC Press, 2011. 107

D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., 1989. 17, 104

S. Gónzalez, S. García, M. Lázaro, A. R. Figueiras-Vidal, and F. Herrera. Class switching according to nearest enemy distance for learning from highly imbalanced data-sets. *Pattern Recognition*, 70:12–24, 2017. 18

I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014. 3, 28, 123, 127

I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. MIT press, 2016. 2

R. Guermazi, I. Chaabane, and M. Hammami. Aecid: Asymmetric entropy for classifying imbalanced data. *Information Sciences*, 467:373 – 397, 2018. 19

H. Guo and W. Wang. An active learning-based svm multi-class classification model. *Pattern recognition*, 48(5):1577–1597, 2015. 21

T. Guo, X. Zhu, Y. Wang, and F. Chen. Discriminative sample generation for deep imbalanced learning. In *IJCAI*, pages 2406–2412, 2019. 29

S. Gurumurthy, R. Kiran Sarvadevabhatla, and V. B. Radhakrishnan. DeLiGAN: generative adversarial networks for diverse and limited data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 166–174, 2017. 150

A. Gut. *Probability: A Graduate Course*. Springer-Verlag, 1 edition, 2005. 81, 83

A. Gutiérrez-López, F.-J. González-Serrano, and A. R. Figueiras-Vidal. Asymmetric label switching resists binary imbalance. *Information Fusion*, 60:20–24, 2020. 18

I. Guyon. Datasets for the agnostic learning vs. prior knowledge competition, 2006. URL `http://www.agnostic.inf.ethz.ch/datasets.php`. 97, 117

J. Ha and J.-S. Lee. A new under-sampling method using genetic algorithm for imbalanced data classification. In *Proceedings of the 10th International Conference on Ubiquitous Information Management and Communication*, pages 1–6, 2016. 14

G. Haixiang, L. Yijing, J. Shang, G. Mingyun, H. Yuanyue, and G. Bing. Learning from class-imbalanced data: Review of methods and applications. *Expert Systems with Applications*, 73:220 – 239, 2017. 6

P. Hall, B. U. Park, and R. J. Samworth. Choice of neighbour order in nearest-neighbour classification. *The Annals of Statistics*, 36(5):2135–2152, 2008. 102

H. Han, W.-Y. Wang, and B.-H. Mao. Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning. In *Advances in Intelligent Computing*, pages 878–887, 2005. 16, 124

Hanchuan Peng, Fuhui Long, and C. Ding. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1226–1238, 2005. 103

D. J. Hand and R. J. Till. A simple generalisation of the area under the ROC curve for multiple class classification problems. *Machine learning*, 45(2):171–186, 2001. 38, 39

P. Hart. The condensed nearest neighbor rule (corresp.). *IEEE transactions on information theory*, 14(3):515–516, 1968. 13

M. Havaei, A. Davy, D. Warde-Farley, A. Biard, A. Courville, Y. Bengio, C. Pal, P.-M. Jodoin, and H. Larochelle. Brain tumor segmentation with deep neural networks. *Medical Image Analysis*, 35:18 – 31, 2017. 27

M. Hayat, S. Khan, S. W. Zamir, J. Shen, and L. Shao. Gaussian affinity for max-margin class imbalanced learning. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. 29

S. S. Haykin. *Neural networks and learning machines*. Prentice Hall, 3 edition, 2009. 19

H. He and E. A. Garcia. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284, 2009. 6

H. He and Y. Ma. *Imbalanced Learning: Foundations, Algorithms, and Applications*. Wiley-IEEE Press, 1st edition, 2013. 6

H. He, Y. Bai, E. A. Garcia, and S. Li. ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In *IEEE International Joint Conference on Neural Networks*, pages 1322–1328, 2008. 16, 124

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on CVPR*, pages 770–778, 2016. 25

M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems 30*, pages 6626–6637, 2017. 144

S. Hido, H. Kashima, and Y. Takahashi. Roughly balanced bagging for imbalanced data. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 2(5-6):412–426, 2009. 24

G. E. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006. 19

T. K. Ho. Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, pages 278–282. IEEE, 1995. 18, 19

M. Hollander, D. A. Wolfe, and E. Chicken. *Nonparametric statistical methods*. John Wiley & Sons, 2013. 98

R. C. Holte, L. Acker, and B. Porter. Concept learning and the problem with small disjuncts. In *Proccedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 813–818, 1989. 15

J.-L. Hsu, P.-C. Hung, H.-Y. Lin, and C.-H. Hsieh. Applying under-sampling techniques and cost-sensitive learning methods on risk assessment of breast cancer. *Journal of medical systems*, 39(4):40, 2015. 25

B.-G. Hu, R. Gosine, L. Cao, and C. de Silva. Application of a fuzzy classification technique in computer grading of fish products. *IEEE transactions on fuzzy systems*, 6(1):144–152, 1998. 72

S. Hu, Y. Liang, L. Ma, and Y. He. Msmote: Improving classification performance when training data is imbalanced. In *2009 second international workshop on computer science and engineering*, volume 2, pages 13–17. IEEE, 2009. 16

X. Hu and C. Xie. Improving fuzzy k-nn by using genetic algorithm. *Journal of Computational Information Systems*, 2005. 104, 107, 117

C. Huang, Y. Li, C. Change Loy, and X. Tang. Learning deep representation for imbalanced classification. In *Proceedings of the IEEE Conference on CVPR*, pages 5375–5384, 2016. 25, 29, 30, 38, 39, 123, 134

U. Hwang, D. Jung, and S. Yoon. Hexagan: Generative adversarial nets for real world classification. In *Proceedings of the 36th International Conference on Machine Learning*, pages 2921–2930, 2019. 28

T. Imam, K. M. Ting, and J. Kamruzzaman. z-svm: An svm for improved classification of imbalanced data. In *Australasian Joint Conference on Artificial Intelligence*, pages 264–273. Springer, 2006. 12, 20

S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on International Conference on Machine Learning*, pages 448–456, 2015. 181, 182

H. Ishibuchi, K. Nozaki, and H. Tanaka. Distributed representation of fuzzy rules and its application to pattern classification. *Fuzzy sets and systems*, 52(1):21–32, 1992. 14

N. Jaccard, T. W. Rogers, E. J. Morton, and L. D. Griffin. Detection of concealed cars in complex cargo x-ray imagery using deep learning. *Journal of X-ray Science and Technology*, 25(3):323–339, 2017. 26, 27

N. Japkowicz. The class imbalance problem: Significance and strategies. In *Proceedings of the 2000 ICAI*, pages 111–117, 2000. 4, 12, 14, 20, 63, 97

N. Japkowicz. Why question machine learning evaluation methods (an illustrative review of the shortcomings of current methods). In *AAAI Workshop on Evaluation Methods for Machine Learning*, pages 6–11, 2006. 22, 23, 30, 38, 97

N. Japkowicz. *Assessment Metrics for Imbalanced Learning*, chapter 8, pages 187–206. John Wiley & Sons, Ltd, 2013. 39

M. Jaworski, P. Duda, and L. Rutkowski. On applying the restricted boltzmann machine to active concept drift detection. In *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–8. IEEE, 2017a. 30

M. Jaworski, P. Duda, L. Rutkowski, P. Najgebauer, and M. Pawlak. Heuristic regression function estimation methods for data streams with concept drift. In *International Conference on Artificial Intelligence and Soft Computing*, pages 726–737. Springer, 2017b. 30

M. Jaworski, P. Duda, and L. Rutkowski. Concept drift detection in streams of labelled data using the restricted boltzmann machine. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7. IEEE, 2018. 30

C. Jian, J. Gao, and Y. Ao. A new sampling method for classifying imbalanced data based on support vector machine ensemble. *Neurocomputing*, 193:115–122, 2016. 17

L. Jiang, C. Li, Z. Cai, and H. Zhang. Sampled bayesian network classifiers for class-imbalance and cost-sensitive learning. In *2013 IEEE 25th International Conference on Tools with Artificial Intelligence*, pages 512–517, 2013. 19

L. Jiang, C. Li, and S. Wang. Cost-sensitive bayesian network classifiers. *Pattern Recognition Letters*, 45:211 – 216, 2014. 19

T. Jo and N. Japkowicz. Class imbalances versus small disjuncts. *ACM Sigkdd Explorations Newsletter*, 6(1):40–49, 2004. 15

J. M. Johnson and T. M. Khoshgoftaar. Deep learning and thresholding with class-imbalanced big data. In *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, pages 755–762, 2019a. 29

J. M. Johnson and T. M. Khoshgoftaar. Deep learning and data sampling with imbalanced big data. In *2019 IEEE 20th International Conference on Information Reuse and Integration for Data Science (IRI)*, pages 175–183, 2019b. 27

J. M. Johnson and T. M. Khoshgoftaar. Survey on deep learning with class imbalance. *Journal of Big Data*, 6(1):27, 2019. 6, 25, 123

M. V. Joshi. On evaluating performance of classifiers for rare classes. In *2002 IEEE International Conference on Data Mining, 2002. Proceedings.*, pages 641–644, 2002. 39

M. V. Joshi, V. Kumar, and R. C. Agarwal. Evaluating boosting algorithms to classify rare classes: Comparison and improvements. In *Proceedings 2001 IEEE International Conference on Data Mining*, pages 257–264. IEEE, 2001. 23

N. Junsomboon and T. Phienthrakul. Combining over-sampling and under-sampling techniques for imbalance dataset. In *Proceedings of the 9th International Conference on Machine Learning and Computing*, pages 243–247, 2017. 18

F. Kamalov. Kernel density estimation based sampling for imbalanced class distribution. *Information Sciences*, 512:1192–1201, 2020. 17

P. Kang and S. Cho. Eus svms: Ensemble of under-sampled svms for data imbalance problems. In *International Conference on Neural Information Processing*, pages 837–846. Springer, 2006. 24

Q. Kang, X. Chen, S. Li, and M. Zhou. A noise-filtered under-sampling scheme for imbalanced classification. *IEEE transactions on cybernetics*, 47(12):4263–4274, 2016. 18

Q. Kang, L. Shi, M. Zhou, X. Wang, Q. Wu, and Z. Wei. A distance-based weighted undersampling scheme for support vector machines and its application to imbalanced classification. *IEEE transactions on neural networks and learning systems*, 29(9):4152–4165, 2017. 14

S. Katsumata and A. Takeda. Robust cost sensitive support vector machine. In *Artificial Intelligence and Statistics*, pages 434–443, 2015. 19

J. M. Keller, M. R. Gray, and J. A. Givens. A fuzzy k-nearest neighbor algorithm. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-15(4):580–585, 1985. 31, 32, 72, 92, 103, 104, 117

S. Khan, M. Hayat, S. W. Zamir, J. Shen, and L. Shao. Striking the right balance with uncertainty. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 29

S. H. Khan, M. Hayat, M. Bennamoun, F. A. Sohel, and R. Togneri. Cost-sensitive learning of deep feature representations from imbalanced data. *IEEE Transactions on Neural Networks and Learning Systems*, 29(8):3573–3587, 2018. 29, 123, 152

S. Khanchi, M. I. Heywood, and A. N. Zincir-Heywood. Properties of a gp active learning framework for streaming data with class imbalance. In *Proceedings of the genetic and evolutionary computation conference*, pages 945–952, 2017. 21

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980, presented at International Conference on Learning Representations (ICLR)*, 2015. 5, 178

D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 29, 142

M. Koziarski. Radial-based undersampling for imbalanced data classification. *Pattern Recognition*, 102:107262, 2020. 14

N. Kozlovskaia and A. Zaytsev. Deep ensembles for imbalanced classification. In *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 908–913, 2017. 24

B. Krawczyk. Cost-sensitive one-vs-one ensemble for multi-class imbalanced data. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 2447–2452, 2016. 24

B. Krawczyk. Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, 5(4):221–232, 2016. 6, 11, 122

B. Krawczyk and M. Woniak. Diversity measures for one-class classifier ensembles. *Neurocomputing*, 126:36 – 44, 2014. 20

B. Krawczyk, G. Schaefer, and M. Woniak. A cost-sensitive ensemble classifier for breast cancer classification. In *2013 IEEE 8th International Symposium on Applied Computational Intelligence and Informatics (SACI)*, pages 427–430, 2013. 23

B. Krawczyk, Ł. Jeleń, A. Krzyżak, and T. Fevens. One-class classification decomposition for imbalanced classification of breast cancer malignancy data. In *International Conference on Artificial Intelligence and Soft Computing*, pages 539–550. Springer, 2014a. 20

B. Krawczyk, M. Woźniak, and F. Herrera. Weighted one-class classification for different types of minority class examples in imbalanced data. In *2014 IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*, pages 337–344. IEEE, 2014b. 12, 20

B. Krawczyk, M. Woniak, and G. Schaefer. Cost-sensitive decision tree ensembles for effective imbalanced classification. *Applied Soft Computing*, 14:554 – 562, 2014c. 23

B. Krawczyk, G. Schaefer, and M. Woniak. A hybrid cost-sensitive ensemble for imbalanced breast thermogram classification. *Artificial Intelligence in Medicine*, 65(3):219 – 227, 2015. 23

B. Krawczyk, M. Koziarski, and M. Woźniak. Radial-based oversampling for multiclass imbalanced data classification. *IEEE transactions on neural networks and learning systems*, 2019. 17

E. Kriminger, , J. C. Principe, and C. Lakshminarayan. Nearest neighbor distributions for imbalanced classification. In *The 2012 International Joint Conference on Neural Networks*. IEEE Press, 2012. 74, 118

A. Krizhevsky. Learning multiple layers of features from tiny images. Technical Report TR-2009, University of Toronto, 2009. 137

A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 25

M. Kubat, S. Matwin, et al. Addressing the curse of imbalanced training sets: one-sided selection. In *ICML*, volume 97, pages 179–186, 1997. 4, 13, 22, 30, 38, 39, 40, 47, 74, 134

M. Kukar, I. Kononenko, et al. Cost-sensitive learning with neural networks. In *ECAI*, volume 98, pages 445–449, 1998. 18, 19

A. Kumar, P. Sattigeri, and T. Fletcher. Semi-supervised learning with gans: Manifold invariance with improved inference. In *Advances in Neural Information Processing Systems*, pages 5534–5544, 2017. 124, 127

Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989. 25

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 26, 136

Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436–444, 2015. 2

H. Lee, M. Park, and J. Kim. Plankton classification on imbalanced large scale database via convolutional neural networks with transfer learning. In *2016 IEEE international conference on image processing (ICIP)*, pages 3713–3717. IEEE, 2016. 27

J. Lee, Y. Wu, and H. Kim. Unbalanced data classification using support vector machines with active learning on scleroderma lung disease patterns. *Journal of Applied Statistics*, 42(3):676–689, 2015. 13

S. S. Lee. Noisy replication in skewed binary classification. *Computational statistics & data analysis*, 34(2):165–191, 2000. 15

J. L. Leevy, T. M. Khoshgoftaar, R. A. Bauder, and N. Seliya. A survey on addressing high-class imbalance in big data. *Journal of Big Data*, 5(1):42, 2018. 7

J. Leskovec and J. Shawe-Taylor. Linear programming boosting for uneven datasets. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 456–463, 2003. 23

G. Levi and T. Hassner. Age and gender classification using convolutional neural networks. In *Proceedings of the IEEE conference on CVPR workshops*, pages 34–42, 2015. 25, 27

A. Li, T. Luo, T. Xiang, W. Huang, and L. Wang. Few-shot learning with global class representations. In *The IEEE International Conference on Computer Vision (ICCV)*, 2019. 6

J. Li, S. Fong, R. K. Wong, and V. W. Chu. Adaptive multi-objective swarm fusion for imbalanced data classification. *Information Fusion*, 39:1 – 24, 2018a. 12

J. Li, S. Fong, R. K. Wong, and V. W. Chu. Adaptive multi-objective swarm fusion for imbalanced data classification. *Information Fusion*, 39:1–24, 2018b. 22

Y. Li and X. Zhang. Improving k nearest neighbor with exemplar generalization for imbalanced classification. In *Fifteenth Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining*, pages 321–332. Springer-Verlag, 2011. 20, 74

Y. Li, T. Wang, B. Kang, S. Tang, C. Wang, J. Li, and J. Feng. Overcoming classifier imbalance for long-tail object detection with balanced group softmax. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10991–11000, 2020. 29

Y.-F. Li, J. T. Kwok, and Z.-H. Zhou. Cost-sensitive semi-supervised support vector machine. In *Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010. 19

G. Liang and A. G. Cohn. An effective approach for imbalanced classification: unevenly balanced bagging. In *Twenty-Seventh AAAI Conference on Artificial Intelligence*, 2013. 24

J. J. Liang, B.-Y. Qu, P. N. Suganthan, and A. G. Hernndez-Daz. Problem definitions and evaluation criteria for the cec 2013 special session and competition on real-parameter optimization. Technical report, NTU, Singapore, 2013. 109, 111, 112

P. Lim, C. K. Goh, and K. C. Tan. Evolutionary cluster-based synthetic oversampling ensemble (eco-ensemble) for imbalance learning. *IEEE Transactions on Cybernetics*, 47(9): 2850–2861, 2017. 24

C. H. Lin, M. Mausam, and D. S. Weld. Active learning with unbalanced classes and example-generation queries. In *Sixth AAAI Conference on Human Computation and Crowdsourcing*, 2018. 21

M. Lin, K. Tang, and X. Yao. Dynamic sampling approach to training neural networks for multiclass imbalance classification. *IEEE Transactions on Neural Networks and Learning Systems*, 24(4):647–660, 2013. 124

T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE ICCV*, pages 2980–2988, 2017a. 7, 25, 29, 123, 149, 152

W.-C. Lin, C.-F. Tsai, Y.-H. Hu, and J.-S. Jhang. Clustering-based undersampling in class-imbalanced data. *Information Sciences*, 409:17–26, 2017b. 14

Y. Lin, Y. Lee, and G. Wahba. Support vector machines for classification in nonstandard situations. *Machine learning*, 46(1-3):191–202, 2002. 20

C. X. Ling and V. S. Sheng. Cost-sensitive learning and the class imbalance problem, 2008. 12

A. Liu, J. Ghosh, and C. Martin. A framework for analyzing skew in evaluation metrics. In *AAAI Workshop on Evaluation Methods for Machine Learning II*, pages 1–6, 2007. 39

C.-L. Liu and M. Nakagawa. Evaluation of prototype learning algorithms for nearest-neighbor classifier in application to handwritten character recognition. *Pattern Recognition*, 34:601–615, 2001. 20

L. Liu, M. Muelly, J. Deng, T. Pfister, and L.-J. Li. Generative modeling for small-data object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6073–6081, 2019. 152

Q. Liu and C. Liu. A novel locally linear knn method with applications to visual recognition. *IEEE Transactions on Neural Network and Learning Systems*, 2016. doi: 10.1109/TNNLS.2016.2572204. 102

S. Liu, P. Zhu, and S. Qin. An improved weighted knn algorithm for imbalanced data classification. In *2018 IEEE 4th International Conference on Computer and Communications (ICCC)*, pages 1814–1819, 2018. 20

S. Liu, G. Lin, Q. Han, S. Wen, J. Zhang, and Y. Xiang. Deepbalance: Deep-learning and fuzzy oversampling for vulnerability detection. *IEEE Transactions on Fuzzy Systems*, pages 1–1, 2019. 28

W. Liu and S. Chawla. Class confidence weighted knn algorithms for imbalanced data sets. In *Fifteenth Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining*, pages 345–356. Springer-Verlag, 2011. 74

W. Liu, S. Chawla, D. A. Cieslak, and N. V. Chawla. A robust decision tree algorithm for imbalanced data sets. In *Proceedings of the 2010 SIAM International Conference on Data Mining*, pages 766–777. SIAM, 2010. 19

Y. Liu, A. An, and X. Huang. Boosting prediction accuracy on imbalanced datasets with svm ensembles. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 107–118. Springer, 2006. 24

Y. Liu, Y. Zhou, X. Liu, F. Dong, C. Wang, and Z. Wang. Wasserstein gan-based small-sample augmentation for new-generation artificial intelligence: A case study of cancer-staging data in biology. *Engineering*, 5(1):156 – 163, 2019. 28, 123

S. Lomax and S. Vadera. A survey of cost-sensitive decision tree induction algorithms. *ACM Computing Surveys (CSUR)*, 45(2):1–35, 2013. 19

V. López, A. Fernández, J. G. Moreno-Torres, and F. Herrera. Analysis of preprocessing vs. cost-sensitive learning for imbalanced classification. open problems on intrinsic data characteristics. *Expert Systems with Applications*, 39(7):6585–6608, 2012. 25

V. López, A. Fernández, S. García, V. Palade, and F. Herrera. An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. *Information Sciences*, 250:113 – 141, 2013. 7

V. López, A. Fernández, and F. Herrera. On the importance of the validation technique for classification with imbalanced datasets: Addressing covariate shift when data is skewed. *Information Sciences*, 257:1 – 13, 2014. 41

A. Luque, A. Carrasco, A. Martn, and A. de las Heras. The impact of class imbalance in classification performance metrics based on the binary confusion matrix. *Pattern Recognition*, 91:216 – 231, 2019. 40

Y. Ma, M. Kan, S. Shan, and X. Chen. Learning deep face representation with long-tail data: An aggregate-and-disperse approach. *Pattern Recognition Letters*, 133:48 – 54, 2020. 29

T. Maciejewski and J. Stefanowski. Local neighbourhood extension of smote for mining imbalanced data. In *2011 IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*, pages 104–111. IEEE, 2011. 16

Y. Mack and M. Rosenblatt. Multivariate k-nearest neighbor density estimates. *Journal of Multivariate Analysis*, 9:1–15, 1979. 78

J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, pages 281–297, 1967. 2

G. Madjarov, D. Kocev, D. Gjorgjevikj, and S. Džeroski. An extensive experimental comparison of methods for multi-label learning. *Pattern recognition*, 45(9):3084–3104, 2012. 149

D. Mahajan, R. Girshick, V. Ramanathan, K. He, M. Paluri, Y. Li, A. Bharambe, and L. van der Maaten. Exploring the limits of weakly supervised pretraining. In *European Conference on Computer Vision (ECCV)*, pages 185–201, 2018. 62

S. Maheshwari, J. Agrawal, and S. Sharma. New approach for classification of highly imbalanced datasets using evolutionary algorithms. *Int. J. Sci. Eng. Res*, 2(7):1–5, 2011. 17

H. H. Maheta and V. K. Dabhi. Classification of imbalanced data sets using multi objective genetic programming. In *2015 International Conference on Computer Communication and Informatics (ICCCI)*, pages 1–6. IEEE, 2015. 22

J. Maillo, J. Luengo, S. Garca, F. Herrera, and I. Triguero. Exact fuzzy k-nearest neighbor classification for big datasets. In *2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 1–6. IEEE, 2017. 72

I. Mani and I. Zhang. knn approach to unbalanced data distributions: a case study involving information extraction. In *Proceedings of workshop on learning from imbalanced datasets*, volume 126, 2003. 20

X. Mao, Q. Li, H. Xie, R. Y. K. Lau, Z. Wang, and S. Paul Smolley. Least squares generative adversarial networks. In *Proceedings of the IEEE ICCV*, pages 2794–2802, 2017. 28, 133, 139

A. Maratea and A. Petrosino. Asymmetric kernel scaling for imbalanced data classification. In *International Workshop on Fuzzy Logic and Applications*, pages 196–203. Springer, 2011. 12, 18, 21

A. Maratea, A. Petrosino, and M. Manzo. Adjusted f-measure and kernel scaling for imbalanced data learning. *Information Sciences*, 257:331–341, 2014. 12, 21

G. Mariani, F. Scheidegger, R. Istrate, C. Bekas, and C. Malossi. Bagan: Data augmentation with balancing gan. *arXiv preprint arXiv:1803.09655*, 2018. 28, 124

R. T. Marler and J. S. Arora. Survey of multi-objective optimization methods for engineering. *Structural and multidisciplinary optimization*, 26(6):369–395, 2004. 22

H. Masnadi-Shirazi and N. Vasconcelos. Asymmetric boosting. In *Proceedings of the 24th international conference on Machine learning*, pages 609–619, 2007. 23

H. Masnadi-Shirazi and N. Vasconcelos. Risk minimization, probability elicitation, and cost-sensitive svms. In *ICML*, pages 759–766, 2010. 19, 23

D. Mateos-García, J. García-Gutiérrez, and J. C. Riquelme-Santos. On the evolutionary optimization of k-nn by label-dependent feature weighting. *Pattern Recognition Letters*, 33 (16):2232 – 2238, 2012. 103, 107, 117

J. Mathew, M. Luo, C. K. Pang, and H. L. Chan. Kernel-based smote for svm classification of imbalanced datasets. In *IECON 2015 - 41st Annual Conference of the IEEE Industrial Electronics Society*, pages 001127–001132, 2015. 16

J. Mathew, C. K. Pang, M. Luo, and W. H. Leong. Classification of imbalanced data by oversampling in kernel space of support vector machines. *IEEE Transactions on Neural Networks and Learning Systems*, 29(9):4065–4076, 2018. 16

D. Mease, A. Wyner, , and A. Buja. Cost-weighted boosting with jittering and over/under-sampling: Jous-boost. *J. Machine Learning Research*, 8:409–439, 2007. 24

G. Menardi and N. Torelli. Training and assessing classification rules with imbalanced data. *Data Mining and Knowledge Discovery*, 28(1):92–122, 2014. 16

J. Mendal and R. B. John. Type-2 fuzzy set made simple. *IEEE Transaction on Fuzzy system*, 10(2):1475–1488, 2002. 103

L. Metz, B. Poole, D. Pfau, and J. Sohl-Dickstein. Unrolled generative adversarial networks. *arXiv preprint arXiv:1611.02163*, 2016. 28

M. Mirza and S. Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014. 28, 123, 125, 178

M. Muja and D. G. Lowe. Scalable nearest neighbor algorithms for high dimensional data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(11):2227–2240, 2014. 71

R. Munoz-Salinas, E. Aguirre, O. Cordon, and M. Garcia-Silvente. Automatic tuning of a fuzzy visual system using evolutionary algorithms: Single-objective versus multiobjective approaches. *IEEE Transactions on Fuzzy Systems*, 16(2):485–501, 2008. 104

L. Nanni, S. Ghidoni, and S. Brahnam. Handcrafted vs. non-handcrafted features for computer vision classification. *Pattern Recognition*, 71:158 – 172, 2017. 62

Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, page 5, 2011. 137

W. W. Ng, J. Hu, D. S. Yeung, S. Yin, and F. Roli. Diversified sensitivity-based undersampling for imbalance classification problems. *IEEE transactions on cybernetics*, 45(11): 2402–2412, 2014. 14

W. W. Ng, G. Zeng, J. Zhang, D. S. Yeung, and W. Pedrycz. Dual autoencoders features for imbalance classification problem. *Pattern Recognition*, 60:875–889, 2016. 29

W. W. Ng, J. Zhang, C. S. Lai, W. Pedrycz, L. L. Lai, and X. Wang. Cost-sensitive weighting and imbalance-reversed bagging for streaming imbalanced and concept drifting in electricity pricing classification. *IEEE Transactions on Industrial Informatics*, 15(3):1588–1597, 2018. 30

J. Nie, R. M. Anwer, H. Cholakkal, F. S. Khan, Y. Pang, and L. Shao. Enriched feature guided refinement network for object detection. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. 29

N. Nikolaou and G. Brown. Calibrating adaboost for asymmetric learning. In *International Workshop on Multiple Classifier Systems*, pages 112–124. Springer, 2015. 23, 24

N. Nikolaou, N. Edakunni, M. Kull, P. Flach, and G. Brown. Cost-sensitive boosting algorithms: Do we really need them? *Machine Learning*, 104(2-3):359–384, 2016. 23

K. Niu, Z. Zhang, Y. Liu, and R. Li. Resampling ensemble model based on data distribution for imbalanced credit risk evaluation in p2p lending. *Information Sciences*, 536:120 – 134, 2020. 24

H. Núñez, L. Gonzalez-Abril, and C. Angulo. Improving svm classification on imbalanced datasets by introducing a new bias. *Journal of Classification*, 34(3):427–443, 2017. 40

A. Odena, C. Olah, and J. Shlens. Conditional image synthesis with auxiliary classifier gans. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2642–2651. JMLR. org, 2017. 123, 124, 127

N. Ofek, L. Rokach, R. Stern, and A. Shabtai. Fast-cbus: A fast clustering-based undersampling method for addressing the class imbalance problem. *Neurocomputing*, 243:88–102, 2017. 14

S.-H. Oh. Error back-propagation algorithm for classification of imbalanced data. *Neurocomputing*, 74(6):1058 – 1061, 2011. 18, 19

K. Oksuz, B. C. Cam, S. Kalkan, and E. Akbas. Imbalance problems in object detection: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. doi: 10.1109/TPAMI.2020.2981890. 4, 25

W. Ouyang, X. Wang, C. Zhang, and X. Yang. Factors in finetuning deep model for object detection with long-tail distribution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 864–873, 2016. 29

A. Palacios, K. Trawiński, O. Cordón, and L. Sánchez. Cost-sensitive learning of fuzzy rules for imbalanced classification problems using furia. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 22(05):643–675, 2014. 18

T. Pan, J. Zhao, W. Wu, and J. Yang. Learning imbalanced datasets based on smote and gaussian distribution. *Information Sciences*, 512:1214–1233, 2020. 16

R. Paredes and E. Vidal. Learning weighted metrics to minimize nearest-neighbor classification error. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 28(7): 1100–1110, 2006. 104, 105, 107, 117

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988. 19

P. C. Pendharkar. A threshold varying bisection method for cost sensitive learning in neural networks. *Expert Systems with Applications*, 34(2):1456 – 1464, 2008. 19

L. Peng, Y. Xiao-Yang, B. Ting-Ting, and H. Jiu-Ling. Imbalanced data svm classification method based on cluster boundary sampling and dt-knn pruning. *International Journal of Signal Processing, Image Processing and Pattern Recognition*, 7(2):61–68, 2014. 14

M. Peng, Q. Zhang, X. Xing, T. Gui, X. Huang, Y.-G. Jiang, K. Ding, and Z. Chen. Trainable undersampling for class-imbalance learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4707–4714, 2019. 15

P. Peng, W. Zhang, Y. Zhang, Y. Xu, H. Wang, and H. Zhang. Cost sensitive active learning using bidirectional gated recurrent neural networks for imbalanced fault diagnosis. *Neurocomputing*, 407:232 – 245, 2020. 13, 21

Y. Peng. Adaptive sampling with optimal cost for class-imbalance learning. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015. 24

M. Pérez-Ortiz, P. A. Gutiérrez, P. Tino, and C. Hervás-Martínez. Oversampling the minority class in the feature space. *IEEE transactions on neural networks and learning systems*, 27 (9):1947–1961, 2015. 17

B. B. Pineda-Bautista, J. A. Carrasco-Ochoa, and J. F. Martınez-Trinidad. General framework for class-specific feature selection. *Expert Systems with Applications*, 38(8):10018–10024, 2011. 104

M. H. Popel, K. M. Hasib, S. Ahsan Habib, and F. Muhammad Shah. A hybrid undersampling method (husboost) to classify imbalanced data. In *2018 21st International Conference of Computer and Information Technology (ICCIT)*, pages 1–7, 2018. 24

S. Pouyanfar, Y. Tao, A. Mohan, H. Tian, A. S. Kaseb, K. Gauen, R. Dailey, S. Aghajanzadeh, Y.-H. Lu, S.-C. Chen, et al. Dynamic sampling in convolutional neural networks for imbalanced data classification. In *2018 IEEE conference on multimedia information processing and retrieval (MIPR)*, pages 112–117. IEEE, 2018. 27

A. D. Pozzolo, O. Caelen, Y.-A. L. Borgne, S. Waterschoot, and G. Bontempi. Learned lessons in credit card fraud detection from a practitioner perspective. *Expert Systems with Applications*, 41(10):4915 – 4928, 2014. 3

W. Prachuabsupakij. Clus: A new hybrid sampling classification for imbalanced data. In *2015 12th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, pages 281–286. IEEE, 2015. 17

R. C. Prati, G. E. Batista, and M. C. Monard. Class imbalances versus class overlapping: an analysis of a learning system behavior. In *Mexican international conference on artificial intelligence*, pages 312–321, 2004. 7

J. D. Prusa, T. M. Khoshgoftaar, and N. Seliya. Enhancing ensemble learners with data sampling on high-dimensional imbalanced tweet sentiment data. In *The twenty-ninth international flairs conference*, 2016. 24

J. R. Quinlan. Simplifying decision trees. *International journal of man-machine studies*, 27 (3):221–234, 1987. 18, 19

A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 123

E. Ramentol, Y. Caballero, R. Bello, and F. Herrera. Smote-rsb*: a hybrid preprocessing approach based on oversampling and undersampling for high imbalanced data-sets using smote and rough sets theory. *Knowledge and information systems*, 33(2):245–265, 2012a. 17

E. Ramentol, N. Verbiest, R. Bello, Y. Caballero, C. Cornelis, and F. Herrera. Smote-frst: a new resampling method using fuzzy rough set theory. In *Uncertainty Modeling in Knowledge Engineering and Decision Making*, pages 800–805. World Scientific, 2012b. 16

B. Raskutti and A. Kowalczyk. Extreme re-balancing for svms: a case study. *ACM Sigkdd Explorations Newsletter*, 6(1):60–69, 2004. 20

G. Rätsch. IDA benchmark repository, 2001. URL http://ida.first.fhg.de/projects/bench/benchmarks.htm. 96, 117

G. Rätsch, T. Onoda, and K.-R. Müller. Soft margins for adaboost. *Machine learning*, 42 (3):287–320, 2001. 23

M. L. Raymer, , W. F. Punch, E. D. Goodman, L. A. Kuhn, and A. K. Jain. Dimensionality reduction using genetic algorithms. *IEEE Transactions on Evolutionary Computation*, 2 (2):164–171, 2000. 104

R. Razavi-Far, M. Farajzadeh-Zanjani, B. Wang, M. Saif, and S. Chakrabarti. Imputation-based ensemble techniques for class imbalance learning. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–1, 2019. 24

A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. Cnn features off-the-shelf: An astounding baseline for recognition. In *2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 512–519, 2014. 62

D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, pages 1278–1286, 2014. 142

B. Richhariya and M. Tanveer. A reduced universum twin support vector machine for class imbalance learning. *Pattern Recognition*, page 107150, 2020. 17

S. d. Rıo, V. López, J. M. Benítez, and F. Herrera. On the use of mapreduce for imbalanced big data using random forest. *Information Sciences*, 285:112 – 137, 2014. Processing and Mining Complex Data Streams. 24

A. Roy, R. M. Cruz, R. Sabourin, and G. D. Cavalcanti. A study on combining dynamic selection and data preprocessing for imbalance learning. *Neurocomputing*, 286:179 – 192, 2018. 24

E. M. Rudd, L. P. Jain, W. J. Scheirer, and T. E. Boult. The extreme value machine. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(3):762–768, 2018. 30, 40, 149, 153

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533, 1986. 3, 63

P. Sadhukhan. Learning minority class prior to minority oversampling. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2019. 17

P. Sadhukhan and S. Palit. Adaptive learning of minority class prior to minority oversampling. *Pattern Recognition Letters*, 2020. 17

J. A. Sáez, J. Luengo, J. Stefanowski, and F. Herrera. Smote–ipf: Addressing the noisy and borderline examples problem in imbalanced classification by a re-sampling method with filtering. *Information Sciences*, 291:184–203, 2015. 16

J. A. Sáez, B. Krawczyk, and M. Woźniak. Analyzing the oversampling of different classes and types of examples in multi-class imbalanced datasets. *Pattern Recognition*, 57:164–178, 2016. 15

A. Saha and S. Das. Geometric divergence based fuzzy clustering with strong resilience to noise features. *Pattern Recognition Letters*, 79:60–67, 2016. 149

Y. Sahin, S. Bulkan, and E. Duman. A cost-sensitive decision tree approach for fraud detection. *Expert Systems with Applications*, 40(15):5916–5923, 2013. 19

T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. In *Advances in neural information processing systems*, pages 2234–2242, 2016. 124, 127

R. J. Samworth. Optimal weighted nearest neighbour classifiers. *The Annals of Statistics*, 40 (5):2733–2763, 2012. 103

S. Santurkar, L. Schmidt, and A. Madry. A classification-based study of covariate shift in gan distributions. In *International Conference on Machine Learning*, pages 4487–4496, 2018. 29, 123

M. Sarkar. Fuzzy-rough nearest neighbor algorithms in classification. *Fuzzy Sets and Systems*, 158(19):2134–2152, 2007. 72

P. Sarkar, V. Davoodnia, and A. Etemad. Computer-aided diagnosis using class-weighted deep neural network. In *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, pages 410–413, 2019. 29

R. E. Schapire. The strength of weak learnability. *Machine learning*, 5(2):197–227, 1990. 23

C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse, and A. Napolitano. Building useful models from imbalanced data with sampling and boosting. In *FLAIRS conference*, pages 306–311, 2008. 23

C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse, and A. Napolitano. RUSBoost: A hybrid approach to alleviating class imbalance. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 40(1):185–197, 2010. 13, 24, 63

T. R. Shaham, T. Dekel, and T. Michaeli. Singan: Learning a generative model from a single natural image. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4570–4580, 2019. 28

V. S. Sheng and C. X. Ling. Thresholding for making classifiers cost-sensitive. In *AAAI*, pages 476–481, 2006. 19

C. Shorten and T. M. Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):60, 2019. 28

M. J. Siers and M. Z. Islam. Novel algorithms for cost-sensitive classification and knowledge discovery in class imbalanced datasets with an application to nasa software defects. *Information Sciences*, 459:53 – 70, 2018. 19

N. F. F. D. Silva, L. F. Coletta, and E. R. Hruschka. A survey and comparative study of tweet sentiment analysis via semi-supervised learning. *ACM Computing Surveys (CSUR)*, 49(1):1–26, 2016. 2

B. W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman and Hall/CRC, 1988. 72, 92

P. Sobhani, H. Viktor, and S. Matwin. Learning from imbalanced data using ensemble methods and cluster-based undersampling. In *International Workshop on New Frontiers in Mining Complex Patterns*, pages 69–83. Springer, 2014. 24

P. Soda. A multi-objective optimisation approach for class imbalance learning. *Pattern Recognition*, 44(8):1801–1810, 2011. 12, 22

M. Sokolova and G. Lapalme. Performance measures in classification of human communications. In *Advances in Artificial Intelligence*, pages 159–170. Springer Berlin Heidelberg, 2007. 39

M. Sokolova and G. Lapalme. A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, 45(4):427 – 437, 2009. 39, 45, 48, 134

M. Sokolova, N. Japkowicz, and S. Szpakowicz. Beyond accuracy, f-score and roc: A family of discriminant measures for performance evaluation. In *Australian Conference on Artificial Intelligence*, pages 1015–1021, 2006. 23, 38, 97

R. Soleymani, E. Granger, and G. Fumera. Progressive boosting for class imbalance and its application to face re-identification. *Expert Systems with Applications*, 101:271 – 291, 2018. 24

J. Song, X. Lu, and X. Wu. An improved adaboost algorithm for unbalanced classification data. In *2009 Sixth International Conference on Fuzzy Systems and Knowledge Discovery*, volume 1, pages 109–113. IEEE, 2009. 23

J. T. Springenberg. Unsupervised and semi-supervised learning with categorical generative adversarial networks. *arXiv preprint arXiv:1511.06390*, 2015. 124, 127

A. Srivastava, L. Valkov, C. Russell, M. U. Gutmann, and C. Sutton. Veegan: Reducing mode collapse in gans using implicit variational learning. In *Advances in Neural Information Processing Systems 30*, pages 3308–3318, 2017. 29, 124

R. Stecking and K. B. Schebesch. Classification of large imbalanced credit client data with cluster based svm. In *Challenges at the Interface of Data Analysis, Computer Science, and Optimization*, pages 443–451. Springer, 2012. 14

J. Stefanowski and S. Wilk. Improving rule based classifiers induced by modlem by selective pre-processing of imbalanced data. In *Proc. of the RSKD Workshop at ECML/PKDD, Warsaw*, pages 54–65, 2007. 18

J. Stefanowski and S. Wilk. Selective pre-processing of imbalanced data for improving classification performance. In *International Conference on Data Warehousing and Knowledge Discovery*, pages 283–292. Springer, 2008. 18

R. Storn and K. Price. Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11(4):341–359, 1997. 104

J. Sun, J. Lang, H. Fujita, and H. Li. Imbalanced enterprise credit evaluation with dte-sbd: Decision tree ensemble based on smote and bagging with differentiated sampling rates. *Information Sciences*, 425:76 – 91, 2018. 24

J. Sun, H. Li, H. Fujita, B. Fu, and W. Ai. Class-imbalanced dynamic financial distress prediction based on adaboost-svm ensemble combined with smote and time weighting. *Information Fusion*, 54:128 – 144, 2020. 4

Y. Sun, M. S. Kamel, A. K. Wong, and Y. Wang. Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognition*, 40(12):3358 – 3378, 2007. 23

Z. Sun, Q. Song, X. Zhu, H. Sun, B. Xu, and Y. Zhou. A novel ensemble method for classifying imbalanced data. *Pattern Recognition*, 48(5):1623 – 1637, 2015. 24

C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on CVPR*, pages 1–9, 2015. 25

C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016. 25, 63

M. H. Tahan and S. Asadi. Emdid: Evolutionary multi-objective discretization for imbalanced datasets. *Information Sciences*, 432:442–461, 2018. 22

A. Taherkhani, G. Cosma, and T. McGinnity. Adaboost-cnn: An adaptive boosting algorithm for convolutional neural networks to classify multi-class imbalanced datasets using transfer learning. *Neurocomputing*, 404:351 – 366, 2020. 30

A. C. Tan, D. Gilbert, and Y. Deville. Multi-class protein fold classification using a new ensemble machine learning approach. *Genome Informatics*, 14:206–217, 2003. 23

S. Tan. Neighbor-weighted k-nearest neighbor for unbalanced text corpus. *Expert Systems with Applications*, 28:667–671, 2005. 19, 20, 73, 74, 87, 97, 103

R. Tanabe and A. Fukunaga. Success-history based parameter adaptation for differential evolution. In *IEEE CEC*, pages 71–78, 2013. 35, 107, 110, 112

B. Tang and H. He. ENN: Extended nearest neighbor method for pattern recognition. *IEEE Computational Intelligence Magazine*, 10(3):52–60, 2015. 107, 117

B. Tang and H. He. Gir-based ensemble sampling approaches for imbalanced learning. *Pattern Recognition*, 71:306–319, 2017. 24

Y. Tang, Y.-Q. Zhang, N. V. Chawla, and S. Krasser. Svms modeling for highly imbalanced classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(1):281–288, 2008. 24

X. Tao, Q. Li, W. Guo, C. Ren, C. Li, R. Liu, and J. Zou. Self-adaptive cost weights-based support vector machine cost-sensitive ensemble for imbalanced data classification. *Information Sciences*, 487:31 – 56, 2019. 23

X. Tao, Q. Li, W. Guo, C. Ren, Q. He, R. Liu, and J. Zou. Adaptive weighted over-sampling for imbalanced datasets based on density peaks clustering with heuristic filtering. *Information Sciences*, 519:43 – 73, 2020. 15

A. S. Tarawneh, A. B. A. Hassanat, K. Almohammadi, D. Chetverikov, and C. Bellinger. Smotefuna: Synthetic minority over-sampling technique based on furthest neighbour algorithm. *IEEE Access*, 8:59069–59082, 2020. 16

P. Thanathamathee and C. Lursinsap. Handling imbalanced data sets with synthetic boundary data generation using bootstrap re-sampling and adaboost techniques. *Pattern Recognition Letters*, 34(12):1339 – 1347, 2013. 24

K. M. Ting. The problem of small disjuncts: its remedy in decision trees. In *Proceedings of the Tenth Canadian Conference on Artificial Intelligence*, pages 91–97, 1994. 15

K. M. Ting. A comparative study of cost-sensitive boosting algorithms. In *In Proceedings of the 17th International Conference on Machine Learning*, 2000. 23

I. Tomek. Two modifications of cnn. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-6(11):769–772, 1976. 13

K. Trawiński, O. Cordón, A. Quirin, and L. Sánchez. Multiobjective genetic classifier selection for random oracles fuzzy rule-based classifier ensembles: How beneficial is the additional diversity? *Knowledge-Based Systems*, 54:3–21, 2013. 104

K. Trawiński, O. Cordón, and A. Quirin. Embedding evolutionary multiobjective optimization into fuzzy linguistic combination method for fuzzy rule-based classifier ensembles. In *2014 IEEE international conference on fuzzy systems (FUZZ-IEEE)*, pages 1968–1975. IEEE, 2014. 104

I. Triguero, S. Gonzlez, J. M. Moyano, S. Garca, J. Alcal-Fdez, J. Luengo, A. Fernndez, M. J. del Jess, L. Snchez, and F. Herrera. KEEL 3.0: an open source software for multi-stage analysis in data mining. *International Journal of Computational Intelligence Systems*, 10: 1238–1249, 2011. 91, 96, 117, 185

S. Tripathi, S. Chandra, A. Agrawal, A. Tyagi, J. M. Rehg, and V. Chari. Learning to generate synthetic data via compositing. In *Proceedings of the IEEE Conference on CVPR*, pages 461–470, 2019. 28, 152

C.-F. Tsai, W.-C. Lin, Y.-H. Hu, and G.-T. Yao. Under-sampling class imbalanced datasets by combining clustering analysis and instance selection. *Information Sciences*, 477:47–54, 2019. 14

N. Verbiest, E. Ramentol, C. Cornelis, and F. Herrera. Improving smote with fuzzy rough prototype selection to detect noise in imbalanced classification data. In *Ibero-American Conference on Artificial Intelligence*, pages 169–178. Springer, 2012. 16

N. Verbiest, E. Ramentol, C. Cornelis, and F. Herrera. Preprocessing noisy imbalanced datasets using smote enhanced with fuzzy rough prototype selection. *Applied Soft Computing*, 22:511 – 517, 2014. 16

K. Veropoulos, C. Campbell, N. Cristianini, et al. Controlling the sensitivity of support vector machines. In *Proceedings of the international joint conference on AI*, volume 55, page 60, 1999. 19

P. Villar, R. Montes, A. M. Sanchez, and F. Herrera. Fuzzy-citation-knn: A fuzzy nearest neighbor approach for multi-instance classification. In *2016 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 946–952, 2016. 72

P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(Dec):3371–3408, 2010. 17

P. Viola and M. Jones. Fast and robust classification using asymmetric adaboost and a detector cascade. In *Advances in neural information processing systems*, pages 1311–1318, 2002. 23

P. Vuttipittayamongkol and E. Elyan. Neighbourhood-based undersampling approach for handling imbalanced and overlapped data. *Information Sciences*, 509:47–70, 2020. 14

T. Wagner. Convergence of the nearest neighbor rule. *IEEE Transactions on Information Theory*, 17(5):566–571, 1971. 74

N. Wahab, A. Khan, and Y. S. Lee. Two-phase deep convolutional neural network for reducing class skewness in histopathological images based breast cancer detection. *Computers in Biology and Medicine*, 85:86 – 97, 2017. 4

Z. Wan, Y. Zhang, and H. He. Variational autoencoder based synthetic data generation for imbalanced learning. In *2017 IEEE symposium series on computational intelligence (SSCI)*, pages 1–7. IEEE, 2017. 29

B. X. Wang and N. Japkowicz. Boosting support vector machines for imbalanced data sets. *Knowledge and information systems*, 25(1):1–20, 2010. 12, 23

H. Wang, Z. Cui, Y. Chen, M. Avidan, A. B. Abdallah, and A. Kronzer. Predicting hospital readmission via cost-sensitive deep learning. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 15(6):1968–1978, 2018. 29

J. Wang, P. Neskovic, and L. N. Cooper. Improving nearest neighbor rule with a simple adaptive distance measure. *Pattern Recognition Letters*, 28:207–213, 2007. 102

L. Wang, L. Khan, and B. Thuraisingham. An effective evidence theory based k-nearest neighbor (kNN) classification. In *Proceedings of IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, pages 797–801. IEEE, 2008. 20, 74

Q. Wang. A hybrid sampling svm approach to imbalanced data classification. In *Abstract and Applied Analysis*, volume 2014. Hindawi, 2014. 17

Q. Wang, X. Zhou, C. Wang, Z. Liu, J. Huang, Y. Zhou, C. Li, H. Zhuang, and J. Cheng. Wgan-based synthetic minority over-sampling technique: Improving semantic fine-grained classification for lung nodules in ct images. *IEEE Access*, 7:18450–18463, 2019. 28

S. Wang and X. Yao. Diversity analysis on imbalanced data sets by using ensemble models. In *2009 IEEE Symposium on Computational Intelligence and Data Mining*, pages 324–331. IEEE, 2009. 24

S. Wang, Z. Li, W. Chao, and Q. Cao. Applying adaptive over-sampling technique based on data density and cost-sensitive svm to imbalanced learning. In *The 2012 IJCNN*, pages 1–8. IEEE, 2012. 25

S. Wang, L. L. Minku, and X. Yao. A multi-objective ensemble method for online class imbalance learning. In *2014 International Joint Conference on Neural Networks (IJCNN)*, pages 3311–3318. IEEE, 2014. 22

S. Wang, W. Liu, J. Wu, L. Cao, Q. Meng, and P. J. Kennedy. Training deep neural networks on imbalanced data sets. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 4368–4374. IEEE, 2016. 123

X. Wang, A. Shrivastava, and A. Gupta. A-fast-rcnn: Hard positive generation via adversary for object detection. In *Proceedings of the IEEE conference on CVPR*, pages 2606–2615, 2017a. 28

X. Wang, Y. Lyu, and L. Jing. Deep generative model for robust imbalance classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14124–14133, 2020. 28

Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM Computing Surveys (CSUR)*, 2019. 153

Y.-X. Wang, D. Ramanan, and M. Hebert. Learning to model the tail. In *Advances in Neural Information Processing Systems*, pages 7029–7039, 2017b. 29, 123, 134

K. Q. Weinberger, J. Blitzer, and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. In *Advances in neural information processing systems*, pages 1473–1480, 2005. 102

G. M. Weiss. The impact of small disjuncts on classifier learning. *Annals of Information Systems*, 8:193–226, 2010. 15, 152

G. M. Weiss and H. Hirsh. A quantitative study of small disjuncts. *AAAI/IAAI*, 2000: 665–670, 2000. 15, 152

D. Wettschereck and T. G. Dietterich. Locally adaptive nearest neighbor algorithms. In *Advances in Neural Information Processing Systems 6*, pages 184–191. Morgan Kaufmann, 1994. 75

G. Y. Wong, F. H. Leung, and S.-H. Ling. An under-sampling method based on fuzzy logic for large imbalanced dataset. In *2014 ieee International Conference on Fuzzy Systems (fuzz-ieee)*, pages 1248–1252. IEEE, 2014. 14

M. L. Wong, K. Seng, and P. K. Wong. Cost-sensitive ensemble of stacked denoising autoencoders for class imbalance problems in business domain. *Expert Systems with Applications*, 141:112918, 2020. 23

C. Wu, L. Herranz, X. Liu, J. van de Weijer, B. Raducanu, et al. Memory Replay GANs: learning to generate new categories without forgetting. In *Advances in Neural Information Processing Systems*, pages 5966–5976, 2018. 150

G. Wu and E. Y. Chang. Adaptive feature-space conformal transformation for imbalanced-data learning. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 816–823, 2003. 21

G. Wu and E. Y. Chang. Aligning boundary in kernel space for learning imbalanced dataset. In *Fourth IEEE International Conference on Data Mining (ICDM'04)*, pages 265–272. IEEE, 2004. 21

G. Wu and E. Y. Chang. Kba: Kernel boundary alignment considering imbalanced data distribution. *IEEE Transactions on knowledge and data engineering*, 17(6):786–795, 2005. 21

P. Xenopoulos. Introducing deepbalance: Random deep belief network ensembles to address class imbalance. In *2017 IEEE International Conference on Big Data (Big Data)*, pages 3684–3689, 2017. 24

H. Xiao, K. Rasul, and R. Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017. 26, 137

J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *Computer vision and pattern recognition (CVPR), 2010 IEEE conference on*, pages 3485–3492, 2010. 29, 137

J. Xiao, L. Xie, C. He, and X. Jiang. Dynamic classifier ensemble model for customer classification with imbalanced class distribution. *Expert Systems with Applications*, 39(3): 3668–3675, 2012. 23

T. Xiao, Y. Xu, K. Yang, J. Zhang, Y. Peng, and Z. Zhang. The application of two-level attention models in deep convolutional neural network for fine-grained image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 842–850, 2015. 153

S. Xie and Z. Tu. Holistically-nested edge detection. In *Proceedings of the IEEE ICCV*, pages 1395–1403, 2015. 123

S. Xie and Z. Tu. Holistically-nested edge detection. *International Journal of Computer Vision*, 125(1-3):3–18, 2017. 123

K. Yamazaki, M. Kawanabe, S. Watanabe, M. Sugiyama, and K. Muller. Asymptotic bayesian generalization error when training and test distributions are different. In *24th International Conference on Machine learning*, pages 1079–1086, 2007. 40, 41

Y. Yan, M. Chen, M.-L. Shyu, and S.-C. Chen. Deep learning for imbalanced multimedia data classification. In *2015 IEEE International Symposium on Multimedia (ISM)*, pages 483–488. IEEE, 2015. 123

C.-Y. Yang, J.-S. Yang, and J.-J. Wang. Margin calibration in svm class-imbalanced learning. *Neurocomputing*, 73(1-3):397–411, 2009. 12, 24

K. Yang, Z. Yu, X. Wen, W. Cao, C. L. P. Chen, H. Wong, and J. You. Hybrid classifier ensemble for imbalanced data. *IEEE Transactions on Neural Networks and Learning Systems*, 31(4):1387–1400, 2020. 22

M.-S. Yang and C.-H. Chen. On the edited fuzzy k-nearest neighbor rule. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 28(3):461–466, 1998. 72

M.-S. Yang and C.-T. Chen. On strong consistency of the fuzzy generalized nearest neighbor rule. *Fuzzy sets and systems*, 60:273–281, 1991. 34, 74

X. Ye, H. Li, A. Imakura, and T. Sakurai. An oversampling framework for imbalanced classification based on laplacian eigenmaps. *Neurocomputing*, 399:107 – 116, 2020. 16

S.-J. Yen and Y.-S. Lee. Under-sampling approaches for improving prediction of the minority class in an imbalanced dataset. In *Intelligent Control and Automation*, pages 731–740. Springer, 2006. 14

S.-J. Yen and Y.-S. Lee. Cluster-based under-sampling approaches for imbalanced data distributions. *Expert Systems with Applications*, 36(3):5718–5727, 2009. 14

C. You, C. Li, D. P. Robinson, and R. Vidal. Scalable exemplar-based subspace clustering on class-imbalanced data. In *The European Conference on Computer Vision (ECCV)*, September 2018. 27

X. You, R. Wang, and D. Tao. Diverse expected gradient active learning for relative attributes. *IEEE transactions on image processing*, 23(7):3203–3217, 2014. 21

F. Yu, A. Seff, Y. Zhang, S. Song, T. Funkhouser, and J. Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015. 41, 137

H. Yu, J. Ni, and J. Zhao. Acosampling: An ant colony optimization-based undersampling method for classifying imbalanced dna microarray data. *Neurocomputing*, 101:309–318, 2013. 14

X. Yuan, L. Xie, and M. Abouelenien. A regularized ensemble framework of deep learning for cancer detection from multi-class, imbalanced training data. *Pattern Recognition*, 77: 160 – 172, 2018. 30

L. Zadeh. Fuzzy sets. *Information and Control*, 8(3):338–353, 1965. 71

C. Zhang, K. C. Tan, and R. Ren. Training cost-sensitive deep belief networks on imbalance data problems. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 4362–4367, 2016. 19

C. Zhang, K. C. Tan, H. Li, and G. S. Hong. A cost-sensitive deep belief network for imbalanced classification. *IEEE transactions on neural networks and learning systems*, 30 (1):109–122, 2018a. 19

C. Zhang, Y. Zhou, Y. Chen, Y. Deng, X. Wang, L. Dong, and H. Wei. Over-sampling algorithm based on vae in imbalanced classification. In *International Conference on Cloud Computing*, pages 334–344. Springer, 2018b. 29

C. Zhang, Y. Zhou, and Y. Deng. Vcos: A novel synergistic oversampling algorithm in binary imbalance classification. *IEEE Access*, 7:145435–145443, 2019. 17

H. Zhang and M. Li. Rwo-sampling: A random walk over-sampling approach to imbalanced data classification. *Information Fusion*, 20:99 – 116, 2014. 12

J. Zhang and I. Mani. Knn approach to unbalanced data distributions: A case study involving information extraction. In *Proceedings of the ICML'2003 Workshop on Learning from Imbalanced Datasets*, pages 1–7, 2003. 74

J. Zhang and A. C. Sanderson. Jade: Adaptive differential evolution with optional external archive. *IEEE Transactions on Evolutionary Computation*, 13(5):945–958, 2009. 110

J. Zhang, X. Wu, and V. S. Shengs. Active learning with imbalanced multiple noisy labeling. *IEEE transactions on cybernetics*, 45(5):1095–1107, 2014a. 21

W. Zhang, R. Ramezani, and A. Naeim. Wotboost: Weighted oversampling technique in boosting for imbalanced learning. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 2523–2531, 2019. 24

X. Zhang and Y. Li. A positive-biased nearest neighbour algorithm for imbalanced classification. In *Advances in Knowledge Discovery and Data Mining*, pages 293–304. Springer Berlin Heidelberg, 2013. 20, 74

X. Zhang, T. Yang, and P. Srinivasan. Online asymmetric active learning with imbalanced data. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 2055–2064, 2016a. 21

X. Zhang, Y. Zhuang, W. Wang, and W. Pedrycz. Transfer boosting with synthetic instances for class imbalanced object recognition. *IEEE transactions on cybernetics*, 48(1):357–370, 2016b. 4

X. Zhang, Y. Li, R. Kotagiri, L. Wu, Z. Tari, and M. Cheriet. Krnn: k rare-class nearest neighbour classification. *Pattern Recognition*, 62:33–44, 2017. 20, 74

Y. Zhang, P. Fu, W. Liu, and G. Chen. Imbalanced data classification based on scaling kernel-based support vector machine. *Neural Computing and Applications*, 25(3-4):927–935, 2014b. 21

Z.-L. Zhang and M.-L. Zhang. Multi-instance multi-label learning with application to scene classification. In *Advances in neural information processing systems*, pages 1609–1616, 2007. 153

Z.-L. Zhang, X.-G. Luo, S. Gonzlez, S. Garca, and F. Herrera. Drcw-aseg: One-versus-one distance-based relative competence weighting with adaptive synthetic example generation for multi-class imbalanced datasets. *Neurocomputing*, 285:176 – 187, 2018c. 24

J. Zhao, M. Mathieu, and Y. LeCun. Energy-based generative adversarial network. *arXiv preprint arXiv:1609.03126*, 2016. 28

P. Zhao and S. C. Hoi. Cost-sensitive online active learning with application to malicious url detection. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 919–927, 2013. 13, 21

Z. Zhao, P. Zhong, and Y. Zhao. Learning svm with weighted maximum margin criterion for classification of imbalanced data. *Mathematical and Computer Modelling*, 54(3-4):1093–1099, 2011. 21

W. Zheng and M. Jin. The effects of class imbalance and training data size on classifier learning: An empirical study. *SN Computer Science*, 2020. doi: https://doi.org/10.1007/s42979-020-0074-0. 7

Zhi-Hua Zhou and Xu-Ying Liu. Training cost-sensitive neural networks with methods addressing the class imbalance problem. *IEEE Transactions on Knowledge and Data Engineering*, 18(1):63–77, 2006. 19

Z.-H. Zhou. A brief introduction to weakly supervised learning. *National Science Review*, 5 (1):44–53, 2018. 3, 153

J. Zhu and E. Hovy. Active learning for word sense disambiguation with methods for addressing the class imbalance problem. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 783–790, 2007. 21

M. Zhu, J. Xia, X. Jin, M. Yan, G. Cai, J. Yan, and G. Ning. Class weights random forest algorithm for processing class imbalanced medical data. *IEEE Access*, 6:4641–4652, 2018. 19

T. Zhu, Y. Lin, and Y. Liu. Synthetic minority oversampling technique for multiclass imbalance problems. *Pattern Recognition*, 72:327–340, 2017. 17

K. Zielinski, D. Peters, and R. Laur. Run time analysis regarding stopping criteria for differential evolution and particle swarm optimization. In *Int. Conf. Experiments/Process/Syst. Modelling/Simulation/Optimization*, 2005. 111

D. Zimbra, A. Abbasi, D. Zeng, and H. Chen. The state-of-the-art in twitter sentiment analysis: A review and benchmark evaluation. *ACM Trans. Manage. Inf. Syst.*, 9(2): 5:1–5:29, 2018. 149