INDIAN STATISTICAL INSTITUTE, KOLKATA

DISSERTATION

Exploring Neural Networks for Gesture Recognition

Author:
Shaunak Gupta

Supervisor:
Prof. Utpal Garain

A dissertation submitted in partial fulfillment of the requirements for the degree of Master of Technology in Computer Science

in the

Computer Vision and Pattern Recognition Unit
Indian Statistical Institute, Kolkata

July 16, 2017

Declaration of Authorship

I, Shaunak Gupta, declare that this dissertation titled, "Exploring Neural Networks for Gesture Recognition" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.

Signed:

Date:

CERTIFICATE

This is to certify that the dissertation entitled **"Exploring Neural Networks for Gesture Recognition"** submitted by **Shaunak Gupta** to Indian Statistical Institute, Kolkata, in partial fulfillment for the award of the degree of **Master of Technology in Computer Science** is a bonafide record of work carried out by him under my supervision and guidance. The dissertation has fulfilled all the requirements as per the regulations of this institute and, in my opinion, has reached the standard needed for submission.

Utpal Garain
Computer Vision and Pattern Recognition Unit,
Indian Statistical Institute, Kolkata,
700108, INDIA.

Abstract

Gesture recognition is the task of recognizing human gestures from video data. In this report, we discuss the ChaLearn LAP Isolated Gesture Dataset and different methods used for gesture recognition from RGB-D videos. We also propose three new methods each involving neural networks in some capacity. The first method uses a pre-trained 3D ConvNet model for feature extraction. The second method uses spatio-temporal interest points and unsupervised learning followed by an LSTM network for prediction. The third method describes the generation of composite difference images that represent the video and then uses 2D ConvNets for prediction. We discuss merits and demerits of each method.

Acknowledgements

I would like to express my heartfelt gratitude to my supervisor Prof. Utpal Garain for encouraging me to pursue research in neural networks and for his constant guidance and support.

My sincere thanks to Akshay Chaturvedi for his valuable suggestions and discussions. I would also like to thank Suraj Agrawal and Rajeev Baditha for their suggestions and ideas.

I am also thankful to my parents, friends, and family for all their support and encouragement.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Gesture Recognition

Gestures have always been an integral part of human communication. Fine motor skills allow us to express ourselves with a myriad of gestures. Some gestures are universal and understood by a vast majority of people whereas others are restricted to certain cultures. In addition to people complementing their words with gestures, the sign language uses gestures as the sole tool to convey meaning.

Humans, from an early age, are very good at recognizing gestures; thus explaining the prevalence of gestures in everyday communication. There have been numerous studies researching different aspects of non-verbal communication and hand gestures (Krauss, Chen, and Chawla, 1996).

Attempts to build automated gesture recognition have met with varied success. With the advent and relative success of neural networks in domains as diverse as natural language processing to computer vision in the past decade, in this report, we discuss and propose some techniques that may be used for gesture recognition.

1.2 The Dataset

ChaLearn Looking at People is a group at Computer Vision Center, University of Barcelona, interested in computer vision research especially involving human gestures, actions, emotions, etc.. Large Scale Isolated Gesture Recognition Challenge is a competition inviting researchers to submit their solutions for the gesture recognition task.

The dataset, known as IsoGD dataset [1], is derived from Chalearn Gesture Dataset (2011) released for one-shot learning challenge. It consists of nearly 50,000 RGB and depth videos each of duration ranging from 1 to 40 seconds. Each video clip contains an isolated gesture performed by a single performer in an indoor space. There are 249 gesture labels and the gestures are performed by 21 individuals.

The complete dataset is divided into three mutually exclusive sets as explained in Table 1.1. The challenge is designed for large scale, user independent learning. For example, the performers that appear in test and validation sets do not appear in the training set. Further dataset details may be found in the overview paper (Wan et al., 2016).

[1] Please refer to http://chalearnlap.cvc.uab.es/dataset/21/description/ for more details on the dataset

TABLE 1.1: Details of training, validation and testing sets

Sets	# of Labels	# of Gestures	# of Performers
Training	249	35878	17
Validation	249	5784	2
Testing	249	6271	2

1.2.1 RGB-D Videos

While the early research on gesture recognition was mostly focused on predicting the gesture from RGB videos, there are now affordable sophisticated products (such as Microsoft Kinect) that capture the depth information in addition to the RGB data. In the IsoGD dataset, there exist 2 videos for every data point in the set. In addition to the 3-channel RGB video, there is also a single channel depth video with pixel intensity proportional to the distance from the camera. Both videos are shot for the same duration with the same frame-rate and have a one-to-one frame correspondence. Figure 1.1 shows some sample frames from a depth and RGB video in the dataset.

FIGURE 1.1: Some sample frames from an RGB and a depth video

1.2.2 Internal validation set

For evaluating recognition system performance, we cannot use the provided validation set repeatedly for the risk of introducing bias in our model. Also, since the system should be user and setting independent, a random split of training dataset would not be useful. Hence, out of the 17 performers in training dataset, 2 performers were chosen and their videos selected as the internal validation set for evaluating performance. The original training dataset of 35878 videos was thus split into a new training set with 30976 videos and an internal validation set with 4902 videos.

1.3 Challenges

This section briefly explains some of the challenges associated with gesture recognition in general as well as with the IsoGD dataset in particular.

Videos have very high dimensionality and a universally good compact feature representation of a video requires further research. Gestures may be performed at various speeds in several different ways. Because of high dimensionality and variability, most recognition systems need extremely large training sets to generalize well. Even though the IsoGD training dataset boasts of around 35,000 RGB-D videos, the average number of training videos per class is less than 150. Hence, any recognition model with a large number of tunable parameters is prone to overfitting. In addition, the within-class variance of training dataset is quite small as there are several instances where the same performer performs a gesture multiple times in the training dataset.

The depth video, though providing useful information, is quite noisy as seen in the sample frames in Figure 1.2 and needs to be preprocessed to be useful. Additionally, even though the frames of RGB and depth videos have a one-to-one correspondence, the pixels do not. Thus, the two videos capture data from different viewpoints and are not perfectly aligned.

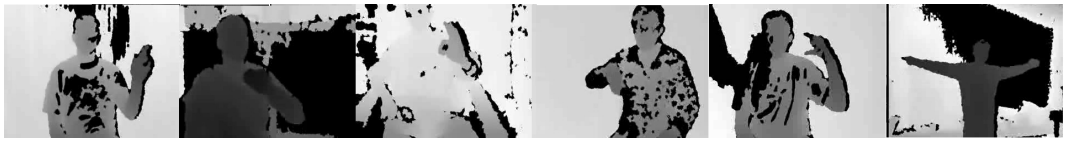

FIGURE 1.2: Example frames from noisy depth videos

1.4 Outline

The rest of this report is organized as follows. Chapter 2 discusses three different approaches to the gesture recognition problem. The first method uses traditional spatio-temporal features to discriminate videos. The second method generates an image representation of a video and then uses a 2D ConvNet to predict gesture labels. The third method uses a 3D ConvNet for an action recognition task. Chapter 3 describes our approaches to the problem. We first present pre-processing methods to remove background and noise from the videos. The first method uses bottleneck features from a pre-trained 3D ConvNet and a small LSTM network for predictions. In the second method, we model a gesture as a sequence of key frames and try to predict the gesture label using an LSTM neural network. The last method generates representative feature images by combining RGB and depth videos and then uses a 2D ConvNet to predict the labels. The final chapter compares the results of various techniques and discusses the future directions of this work.

Chapter 2

Related Work

2.1 Mixed Features around Sparse Keypoints

Mixed Features around Sparse Keypoints (or MFSK) (Wan, Guo, and Li, 2016) is a bag of visual words (BoVW) approach proposed as the baseline method in the overview paper describing the IsoGD dataset (Wan et al., 2016).

The method involves finding keypoints in the video and then calculating feature descriptors for the regions surrounding each keypoint. SURF detector is used for keypoint detection around motion regions.

2.1.1 Spatial Pyramid Building

The first stage of the algorithm builds spatial pyramids for each frame in the RGB-D videos. The pyramids are built by downsampling the image in the spatial domain such that at each level in the pyramid, the image size decreases exponentially. These pyramids are then used to find robust keypoints in the RGB and depth frames.

2.1.2 Keypoint Detection

The initial set of keypoints are extracted by using the SURF feature detector (Bay et al., 2008) in each spatial scale. The algorithm calculates the approximate Hessian matrix for a region surrounding each pixel. The maxima of the determinant of the Hessian matrix is then found in a pre-determined neighborhood. These maxima are chosen as initial keypoints and interpolated in the scale space. An example of extracted keypoints is shown in Figure 2.1.

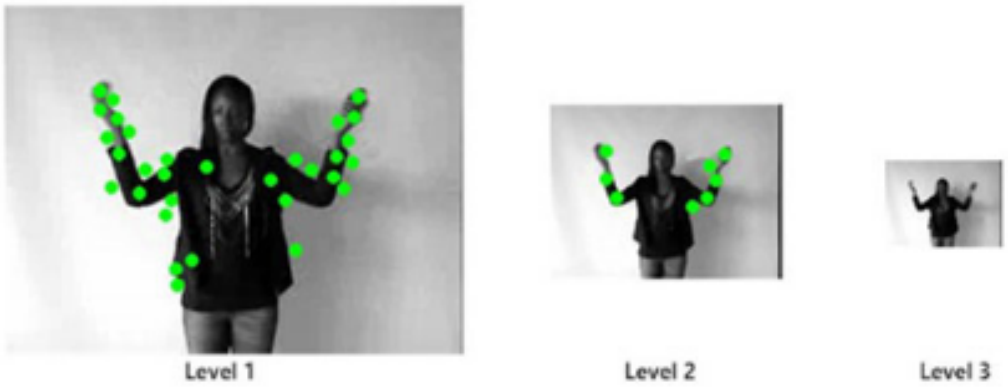

FIGURE 2.1: Keypoints detected at different spatial scales

After this optical flow is calculated for each initially chosen keypoint using the Lucas-Kanade method (Lucas and Kanade, 1981) on successive frames. Keypoints with low velocity are then discarded.

2.1.3 Feature Descriptors

The following four feature descriptors are calculated for the region surrounding each keypoint:

- 3D Sparse Motion SIFT (Wan et al., 2014)
- Histogram of Oriented Gradients (Dalal and Triggs, 2005)
- Histogram of Optical Flows
- Motion Boundary Histograms (Dalal, Triggs, and Schmid, 2006)

These features are then concatenated to form the final feature descriptor. An SVM classifier with a linear kernel is trained to predict the labels.

2.2 Dynamic Images

This method (Wang et al., 2016) uses multiple image representations of a video, called Dynamic Depth Images (DDI), Dynamic Depth Normal Images (DDNI) and Dynamic Depth Motion Normal Images (DDMNI) by using rank pooling (Bilen et al., 2016) to generate the so-called dynamic images from the depth videos. Unlike other methods, the proposed method only used the depth videos to train and predict the gestures (and not using the RGB video at all).

These images are then fed to a deep 2D convolutional neural network with pre-trained weights which are fine-tuned to fit the training data. In the preprocessing step, they use a histogram based approach to remove the background from the depth videos. We also borrow the same method for background removal (described in the next chapter).

2.2.1 Rank Pooling

Rank pooling [1] is a method proposed by Bilen et al. that obtains a dynamic image by encoding the temporal evaluation from the frames of a video. One of the advantages of this method is that since the compact feature representation is a new RGB image, it may be processed by CNN architecture trained on other real world images.

The dynamic image is obtained as a ranking classifier that sorts video frames temporally. Let $I_1, \ldots, I_T$ denote the frames of the video. Let $\psi(I_t) \in \mathbb{R}^d$ denote a representation of the frame I_t. Let $V_t = \frac{1}{t}\sum_{\tau=1}^{t} \psi(I_\tau)$ be the time average of these features up to time t. The ranking function associates to each time t a score $S(t|d) = \langle d, V_t \rangle$, where $d \in \mathbb{R}^d$ is a vector of parameters. The function parameters d are learned so that the scores reflect the rank of the frames in the video. Therefore, later times are associated with larger scores, i.e. $\boldsymbol{q > t \Rightarrow S(q|d) > S(t|d)}$. Learning d is posed as a convex optimization problem using the RankSVM formulation:

$$\boldsymbol{d^* = \rho(I_1, \ldots, I_T; \psi) = \underset{d}{\operatorname{argmin}}\, E(d),}$$

[1] Content taken from the original paper (Bilen et al., 2016)

$$E(d) = \frac{\lambda}{2}\|d\|^2 + \frac{2}{T(T-1)} \times \sum_{q>t} max\{0, 1 - S(q|d) + S(t|d)\}$$

In their construction, $\psi(I_t)$ is simply the RGB pixel representation of the image flattened to a vector. Because the dimensionality of d is same as that of frame representation, the descriptor given by RankSVM d* also has the same dimensionality. Thus, the descriptor d* may be reshaped to form an image. This optimization step is called forward rank pooling. Rank pooling may also be applied in the reverse temporal direction to generate a backward dynamic image.

2.2.2 Normal and Motion Normal Images

In addition to applying the rank pool operation to depth frames, the operation is also applied to normal images where each pixel contains the surface normals (gradients) in the X, Y, and temporal dimension. To create motion normal images, a gaussian mixture model is used to model moving foreground for better background removal. Figure 2.2 shows forward and backward dynamic images for one of the training depth videos.

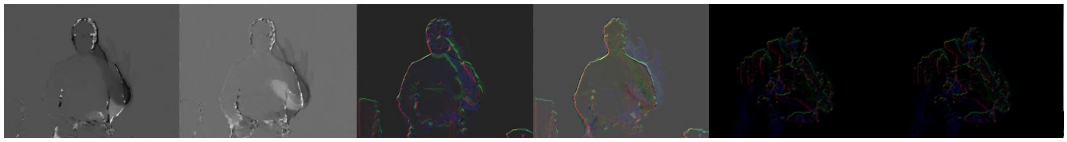

FIGURE 2.2: Extracted dynamic images from a depth video. From left to right - (a) Forward Depth Dynamic Image (b) Backward Depth Dynamic Image (c) Forward Depth Dynamic Normal Image (d) Backward Depth Dynamic Normal Image (e) Forward Depth Dynamic Motion Normal Image (f) Backward Depth Dynamic Motion Normal Image

2.2.3 Training and Network Architecture

A total of six dynamic images are generated from each depth video. These six image representations were trained separately on six different Convolutional Neural Networks. Pretrained VGG16 models were adopted for fine-tuning. Finally, a combination of all neural networks was used to predict the label on the test data.

2.3 C3D - 3D convolutional networks

2D CNNs are extremely good at image classification tasks, therefore a natural extension is to add the temporal dimension to a 2D CNN and use 3D CNNs for video classification. In their paper (Tran et al., 2015), Tran et al. explore using a deep 3D CNN architecture to learn discriminative spatio-temporal features.

Because of spatio-temporal convolution and pooling operations, the temporal information in the input video is preserved. Different kernel sizes are explored and it is concluded that a deep architecture with all kernels of size 3x3x3 is the best option for 3D ConvNets.

The selected design has 8 convolution layers, 5 max-pooling layers, 2 fully connected layers, and a final softmax layer. The network is also known by the short name C3D.

The C3D network is trained on the Sports-1M dataset (Karpathy et al., 2014a) which consists of 1.1 million sports videos. The challenge is to determine the label of a video from among 487 sports categories.

Chapter 3

Proposed Methods

3.1 Video Preprocessing

First, we explain the pre-processing steps performed before the actual model is trained.

3.1.1 Background Removal

The first step in preprocessing is to remove the background from the videos. Because of the diverse backgrounds in the RGB videos, it is much simpler to use the depth video for background removal. In a depth image, a higher pixel intensity implies that the object represented by the pixel is farther from the sensor and vice-versa.

A histogram based approach (Wang et al., 2016) is used to determine the background in depth videos. It is a reasonable assumption that most of the background will be a flat or nearly flat surface and will have higher depth intensity values compared to the foreground. We use this assumption to set a threshold depth value.

Specifically, a histogram of intensity values in the depth video is calculated. The threshold for background depth is set as the location of the peak in the latter half of the histogram bin locations minus a fixed tolerance. A tolerance value of 0.1 yielded good results. Only the first frame of the video is used to set this threshold. Once the threshold is set, all pixels having a higher intensity value in any of the frames are set to background intensity of 1. Figure 3.1 compares some frames before and after background removal.

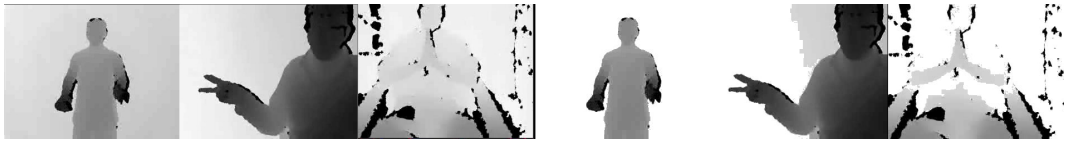

FIGURE 3.1: The first three images on the left are the original extracted depth frames. The latter three images have the background removed from them.

3.1.2 Denoising depth videos

The depth videos are much more susceptible to noise compared to RGB videos. Some of the data is not recorded accurately by the sensor, and as a result, some pixels show as black pixels in the video. To correct this, we use a combination of filters and connected components to regard these pixels as background.

In many frames of depth videos, a large chunk of the image is blackened due to unregistered data from the sensor. To correct this, we find all the connected components in the

depth image with zero depth intensity. Our hypothesis is that a large connected component of black pixels is much more likely to be noise than signal. Thus, every such connected component covering more pixels than a set threshold is set to the background value of 1. A threshold value of 5% of the total pixels in the image generated good results.

We then smooth the image using a gaussian filter with a standard deviation of 0.7 in both the dimensions. Because we'd like the background to have zero intensity pixels, we invert the image thus getting zero intensity pixels as the background and higher intensity pixels for the foreground.

The results at different stages of preprocessing can be seen in Figure 3.2 below.

FIGURE 3.2: From left to right: (a) Original frame from video (b) After background removal (c) After using connected components to filter noise (d) After applying gaussian filter (e) Final inverted image

3.1.3 Aligning Depth videos with RGB videos

As explained in Chapter 1, while there is a one-to-one frame correspondence for RGB and depth videos, the pixels are not aligned perfectly. The next preprocessing step is to transform the depth video such that both videos are aligned.

For this purpose, we assume that the transformation is linear and involves only translation and scaling (i.e. no shear or rotation). Intuitively, this seems to be a reasonable assumption. Concretely, a pixel at position (x, y) in the depth video is transformed to a pixel at position (x', y') in the RGB video. Under our assumptions, the transformation may be written as:

$$x' = \alpha_x + \beta_x * x$$

$$y' = \alpha_y + \beta_y * y$$

Here α_x, β_x are the translation and scaling parameters respectively in x dimension and α_y, β_y are the translation and scaling parameters respectively in y dimension. To estimate the parameters, 10 points were marked manually on the first frame of both RGB and depth video of a randomly selected datapoint. All the 10 points must follow the same transformation and therefore a linear curve fit that minimizes the mean squared error provides us with the estimated values of parameters. Incidentally, in this case, the transformation parameters in X and Y dimensions are almost equal. Therefore, we take the mean of parameters in both dimensions to maintain symmetry.

Now that both the videos are pixelwise aligned, we may find the background pixels in depth video and set the corresponding pixels in RGB video as background. Some examples of background removal in RGB frames are shown in Figure 3.3.

FIGURE 3.3: Some examples of background removal in RGB frames using aligned depth video

3.2 Using Sports1M C3D bottleneck features

The first attempted approach was to use pre-trained bottleneck features extracted by a 3D convolutional network. C3D features (Tran et al., 2015) are spatio-temporal features extracted by training a deep 3-dimensional convolutional neural network on the Sports-1M dataset (Karpathy et al., 2014b).

3.2.1 Motivation

Deep 2D convolutional neural networks trained on very large datasets are able to learn complex features that consistently beat hand-crafted features for image recognition tasks. 3D CNNs are an extension to 2D CNNs to process video data where the third dimension is temporal. Since we do not have a large dataset to train a deep 3D ConvNet, instead we leverage an existing model pre-trained on a large dataset. The intuition is that such a network would have already learned a lot of low-level features necessary to perform a variety of recognition tasks. Moreover, both Sports 1M and IsoGD are action recognition tasks with large number of classes. C3D's prediction accuracy of over 60% on the Sports 1M dataset shows that the model has the capacity to learn complex features to discriminate among a large number of classes.

3.2.2 Design

The original C3D network architecture contains 5 groups of convolutional layers that in total include 8 convolutional layers, each with a kernel size of (3,3,3), 5 max-pooling layers, each with a pooling size of (2,2,2) (except the first pooling layer). The network accepts 3-channel videos of resolution $\mathbf{112 \times 112}$ pixels and a fixed temporal dimension of 16. Table 3.1 describes the ConvNet model.

After loading pre-trained weights, fully connected layers are removed from the network up to the last densely connected layer of size 4096. Thus, the network outputs a 4096 sized vector as feature representation of an $\mathbf{112 \times 112 \times 16 \times 3}$ sized input.

The RGB videos in the IsoGD dataset have a resolution of $\mathbf{240 \times 320}$ pixels which can be resized using interpolation to match the required resolution. However, the videos are of variable duration and therefore cannot be fed directly to the ConvNet. Therefore, we group the video frames in blocks of 16 (after necessary zero-padding such that 16 divides the total number of frames) and then feed these blocks to the modified C3D model to get a feature representation of each block.

We now have each data label represented as a variable length sequence of 4096 sized vectors. Long Short Term Memory (LSTM) networks are powerful tools that may be used to model sequences. We design a small LSTM network with 128 units connected to a fully connected softmax layer to give output probabilities of gesture labels.

Layer	# of units	Kernel/Pooling Size	Stride	# of parameters
Convolution3D	64	(3,3,3)	(1,1,1)	5248
MaxPooling3D	-	(1,2,2)	(1,2,2)	0
Convolution3D	128	(3,3,3)	(1,1,1)	221312
MaxPooling3D	-	(2,2,2)	(2,2,2)	0
Convolution3D	256	(3,3,3)	(1,1,1)	884992
Convolution3D	256	(3,3,3)	(1,1,1)	1769728
MaxPooling3D	-	(2,2,2)	(2,2,2)	0
Convolution3D	512	(3,3,3)	(1,1,1)	3539456
Convolution3D	512	(3,3,3)	(1,1,1)	7078400
MaxPooling3D	-	(2,2,2)	(2,2,2)	0
Convolution3D	512	(3,3,3)	(1,1,1)	7078400
Convolution3D	512	(3,3,3)	(1,1,1)	7078400
ZeroPadding3D	-	(0,1,1)	(1,1,1)	0
MaxPooling3D	-	(2,2,2)	(2,2,2)	0
Flatten	-	-	-	0
Fully Connected	4096	-	-	33558528
Dropout	-	-	-	0
Fully Connected	4096	-	-	16781312
Dropout	-	-	-	0
Fully Connected	487	-	-	1995239

TABLE 3.1: The C3D Model for feature extraction.

3.3 Extracting key frames and frame clustering

The second attempted approach was to use unsupervised learning to cluster similar frames together and then model the video (and gesture) as a sequence of clusters representations.

3.3.1 Motivation

One of the shortcomings of earlier approaches was a lack of sufficient data to prevent overfitting. To overcome this, we decided to extract relevant frames from the videos (pre-processed depth frames with background removed). The total number of frames is rather large, and even though they are not associated with a label (as there may be multiple gestures that include a similar looking posture), we may perform unsupervised clustering to cluster them into relevant classes. The video can then be modeled as a sequence of clusters or frame representations to train and predict the gesture labels.

Similar methods have been used for modeling of human actions (Li, Zhang, and Liu, 2008; Wang, 2011). Our approach was to extract the key frames from the video and then use a translation and rotation invariant feature descriptor to represent the postures. These postures can then be clustered using an unsupervised clustering scheme.

3.3.2 Spatio-temporal interest points and feature descriptors

To extract relevant key frames, we determine the spatio-temporal interest points (Dollár et al., 2005) after removing background from the video. The key points in frames are detected using a periodic response filter.

Any video frame that has above a certain threshold of interest points is deemed a key frame. Then, for each interest point in a key frame, a spatio-temporal neighborhood of size

$\mathbf{5} \times \mathbf{5} \times \mathbf{5}$ is chosen and 3D gradients (in X, Y, and temporal dimension) are calculated for each pixel in this window. The gradients are then mapped to spherical coordinates and a histogram is made with azimuth and elevation angles. 10 bins were chosen for azimuth and elevation resulting in 100 combinations of bins. This vector of size 100 was chosen as a feature descriptor for the posture. This descriptor resembles histogram of ordinated gradients (HOG) (Dalal and Triggs, 2005), but instead of calculating histogram for all regions of the image, we only do so in a window surrounding the interest points. The feature descriptor is normalized by the L2 norm.

3.3.3 Clustering and Cluster embedding

The frames were clustered into 1000 clusters using *MiniBatchKMeans* algorithm. The gesture video can now be thought of as a sequence of key frames. This is analogous to a language model where a cluster label corresponds to a word and a gesture video corresponds to a sentence. Notice that the postures are not generally independent in a gesture, and there lies some context in which a posture appears. Further, the clustering of postures may not be perfect and to remedy this we choose an embedding for a cluster label that is low dimension and preserves context. Figure 3.4 shows some sample frames from a cluster. Notice that though there are some frames with a similar posture, there isn't a clear pattern to all the frames. Thus implying that our feature representation is not suited for posture clustering.

Again, we borrow the word2vec (Mikolov et al., 2013) algorithm from natural language processing for choosing cluster embedding. A vector of size 50 was generated for each cluster using the continuous bag of words (CBOW) method. The sequences were fed to a 128 unit LSTM network connected to a 249 densely connected softmax activated neuron layer to predict the gesture label.

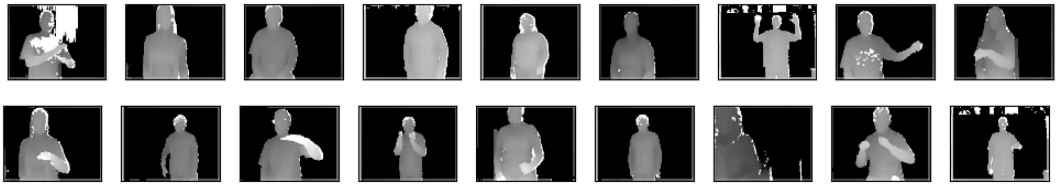

FIGURE 3.4: Some sample frames from a cluster

3.4 Using composite difference images

The third attempted method extracts two images as a feature representation of the gesture video. These images are then fed to deep 2-dimensional convolutional neural networks with image augmentation to predict the gesture labels. In the following sections, we describe the motivation and implementation of the algorithm.

3.4.1 Motivation

The idea to use an image as a feature descriptor was inspired by motion blur images. These images capture the important spatio-temporal features and reduce the emphasis on the background. We have previously discussed the rank pooling method (Bilen et al., 2016) to generate dynamic images. Our approach is to generate two 4-channel RGBA images using a

combination of depth and RGB videos, primarily concentrating on the pixels that change significantly.

3.4.2 Extracting the difference image

The extracted difference images capture the pixels that change significantly during the course of the video and tries to remove the parts that do not do so. Therefore, it filters out the non-discriminative information such as the background, the performer, etc..

We start with the preprocessing steps explained in Section 3.1. This includes aligning depth video with RGB, background removal and denoising of depth video. We define a threshold value (chosen to be 0.12) which is the minimum change in pixel intensity that is considered significant. For every successive frame, we calculate the number of non-background pixels that have undergone a significant change in both the RGB and the depth video. We call these pixels as important pixels.

If the count of important pixels is larger than a certain threshold (chosen to be 10% of the total number of pixels), we consider the data to be noisy and ignore the change. On the other hand, if it is lesser than 1% of the total number of pixels, the change is deemed insignificant. Otherwise, we calculate a mask using the connected component algorithm that extracts the performer's silhouette from the depth image to further reduce the noise in important pixels.

Creating the mask

In a sizable number of depth videos, there are other objects in the frame but aren't removed by the background removal algorithm described above. Further, due to noise, these pixels occasionally change values causing the noise to creep in the difference image. To overcome this, we try to extract the silhouette of the performer in the depth video (which is still recognizable) using the connected component algorithm.

The connected component algorithm (Rosenfeld and Pfaltz, 1966) is used to partition a binary image into separate labeled connected components. We create a fast generalized version of the algorithm that works on grayscale images. We define two pixels to be part of the same connected component if they are 4-neighbors and the absolute difference in their intensities is less than the connectivity threshold (chosen to be 0.08).

Because we are only trying to find the performer's silhouette, instead of calculating all the connected components, we use the following approach. We hypothesize that of all the non-background pixels that change significantly in successive frames, a majority belong to the connected component describing the performer's silhouette. Thus, a random sample of size 30% is taken from among the important pixels. The generalized connected component algorithm is run taking these points as starting points. Finally, the connected component that a majority of these pixels belong to is chosen as the required silhouette and the remaining pixels are set to zero to create the mask. Figure 3.5 shows some example silhouettes extracted using the mask and the depth frame.

The silhouette mask is used to further filter the important pixels. All other pixels are black in the difference images.

Creating the images

We create two 4-channel difference image feature representations.

FIGURE 3.5: Performer's silhouettes derived from the mask.

- Depth Difference Image - The first image primarily uses data from the depth video and the temporal information regarding important pixels. The intensity values of the channels are proportional to:
  - Red Channel - the mean depth value at the moments when the pixel had significant change
  - Blue Channel - the number of times that pixel had a significant change during the course of the video
  - Green Channel - the first time the pixel changed significantly
  - Alpha Channel - the last time the pixel changed significantly

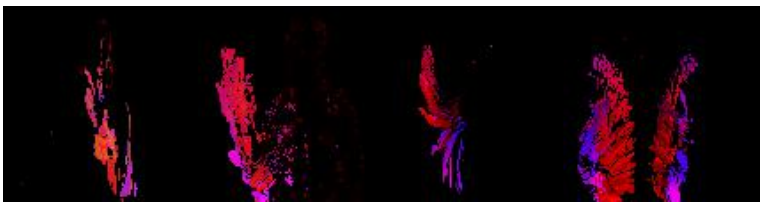

FIGURE 3.6: Some sample depth difference images.

- Gradient Difference Image - The second image primarily uses data from RGB video and the maximum gradient values at important pixels during the course of the video. At each important pixel, the gradient in X, Y and temporal dimension is calculated. The gradient with maximum magnitude is used at important pixels to generate the gradient difference image. The intensity values of the channels are proportional to:
  - Red Channel - the magnitude of the maximum gradient
  - Blue Channel - the gradient component in X direction
  - Green Channel - the gradient component in Y direction
  - Alpha Channel - the gradient component in temporal dimension

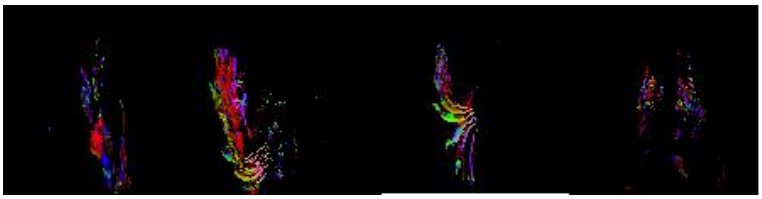

FIGURE 3.7: Some sample gradient difference images.

Figures 3.6 and 3.7 shows some examples of difference images. Notice that only the discriminative parts of the gestures remain in the images.

Layer	# of units	Kernel/Pooling Size	Stride	# of parameters
Convolution2D	64	(3,3)	(1,1)	2368
Convolution2D	64	(3,3)	(1,1)	36928
MaxPooling2D	-	(2,2)	(2,2)	0
Convolution2D	128	(3,3)	(1,1)	73856
Convolution2D	128	(3,3)	(1,1)	147584
MaxPooling2D	-	(2,2)	(2,2)	0
Convolution2D	256	(3,3)	(1,1)	295168
Convolution2D	256	(3,3)	(1,1)	590080
Convolution2D	256	(3,3)	(1,1)	590080
MaxPooling2D	-	(2,2)	(2,2)	0
Convolution2D	512	(3,3)	(1,1)	1180160
Convolution2D	512	(3,3)	(1,1)	2359808
Convolution2D	512	(3,3)	(1,1)	2359808
MaxPooling2D	-	(2,2)	(2,2)	0
Flatten	-	-	-	0
Fully Connected	1024	-	-	25691136
Dropout	-	-	-	0
Fully Connected	1024	-	-	1049600
Dropout	-	-	-	0
Fully Connected	249	-	-	255225

TABLE 3.2: The convolutional neural network architecture.

3.4.3 Network Architecture

Once the difference images are extracted for each gesture video, we train separate 2D ConvNets with image augmentation. We use a slight modification of VGG16 model for our system. The last convolutional group is removed for more simplicity, efficient calculation, and resource constraints. Also, the size of fully-connected layers is reduced to 1024 from 4096 neurons. We also used a dropout regularization of 0.5 to prevent overfitting. Table 3.2 shows the network model and its parameters in detail. The network accepts 4-channel input images of resolution $\mathbf{112} \times \mathbf{112}$ pixels.

Image augmentation was used to compensate for less training data. Random rotations, horizontal flips, height and width shifts, shear, and zoom augmentations were used and the model was trained on adam optimizer for 200 epochs. Both difference images were fed into separate ConvNets and the predicted probabilities were multiplied together. The index with maximum score was chosen to be the predicted class label.

Chapter 4

Results and Future Work

In this chapter, we present the results of our proposed methods and compare them with others' results. We also discuss the probable reasons for a method's good or bad performance. Finally, we conclude with future directions that the work presented in this thesis may lead to.

4.1 Evaluation Criterion

The evaluation criterion for IsoGD dataset is straightforward. We simply measure the accuracy of predictions on the validation dataset. The evaluation criterion is the proportion of data points correctly identified by the system belonging to one of the 249 class labels. There are 5784 data points in the validation dataset.

4.2 Results

Table 4.1 shows the performance of various methods on the validation set of IsoGD dataset.

The second proposed method could not produce a good clustering of depth frames. To produce a good clustering, a better feature representation of key frames is required. Therefore it was extremely slow to learn and even after 150 epochs had a low training as well as validation accuracy.

We also notice that using C3D bottleneck features did not perform nearly as well as expected. One of the possible reasons could be that the discriminating features for a sports action are much more dependent on spatial data including the background. Thus, a CNN only has to recognize a basketball court or a swimming pool in one of the frames to correctly predict the action label. On the other hand, in the case of gestures, the only important signal is spatio-temporal movement of the performer.

We see that using the background removal and silhouette extraction methods in the proposed method 3, there is a substantial jump in performance. It outperformed the baseline method that used a multitude of gradient based hand-crafted features. This demonstrates the effectiveness of 2D ConvNets as well as the importance of filtering out non-useful information such as the background and performer's torso. It can, therefore, be argued that developing even better background removal and silhouette extraction methods would lead to higher accuracy.

TABLE 4.1: Performance on the IsoGD validation dataset

Method	Accuracy
lostoy (Best submission on ChaLearn challenge)	61.39%
CNN on Dynamic Images (Wang et al., 2016)	39.23%
Composite Difference Images (Proposed Method 3)	23.69%
MFSK (Wan, Guo, and Li, 2016)	18.65%
C3D bottleneck features + LSTM (Proposed Method 1)	3.61%
Key frames + clustering (Proposed Method 2)	2.39%

4.3 Conclusion

There is still a lot of scope for research in gesture recognition tasks. Compared to some other tasks where computers have almost caught up to human level accuracies, video gesture recognition systems still have a long way to go. Other ways to generate images like difference images, used in tandem with 2D ConvNets are a way to go.

A variation of the second proposed method that uses a better feature representation that accurately describes the posture of a human silhouette and is invariant to other factors may be useful and perhaps improve the result. Another useful approach may be to use autoencoders to learn deep feature representations of gesture videos utilizing publicly available sign language videos.

Bibliography

Bay, Herbert et al. (2008). "Speeded-up robust features (SURF)". In: *Computer vision and image understanding* 110.3, pp. 346–359.

Bilen, Hakan et al. (2016). "Dynamic image networks for action recognition". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3034–3042.

Dalal, Navneet and Bill Triggs (2005). "Histograms of oriented gradients for human detection". In: *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. Vol. 1. IEEE, pp. 886–893.

Dalal, Navneet, Bill Triggs, and Cordelia Schmid (2006). "Human detection using oriented histograms of flow and appearance". In: *Computer vision–ECCV 2006*, pp. 428–441.

Dollár, Piotr et al. (2005). "Behavior recognition via sparse spatio-temporal features". In: *Visual Surveillance and Performance Evaluation of Tracking and Surveillance, 2005. 2nd Joint IEEE International Workshop on*. IEEE, pp. 65–72.

Karpathy, Andrej et al. (2014a). "Large-scale video classification with convolutional neural networks". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 1725–1732.

— (2014b). "Large-scale Video Classification with Convolutional Neural Networks". In: *CVPR*.

Krauss, Robert M, Yihsiu Chen, and Purnima Chawla (1996). "Nonverbal behavior and nonverbal communication: What do conversational hand gestures tell us?" In: *Advances in experimental social psychology* 28, pp. 389–450.

Li, Wanqing, Zhengyou Zhang, and Zicheng Liu (2008). "Expandable data-driven graphical modeling of human actions based on salient postures". In: *IEEE transactions on Circuits and Systems for Video Technology* 18.11, pp. 1499–1510.

Lucas, Bruce D, Takeo Kanade, et al. (1981). "An iterative image registration technique with an application to stereo vision". In:

Mikolov, Tomas et al. (2013). "Efficient estimation of word representations in vector space". In: *arXiv preprint arXiv:1301.3781*.

Rosenfeld, Azriel and John L Pfaltz (1966). "Sequential operations in digital picture processing". In: *Journal of the ACM (JACM)* 13.4, pp. 471–494.

Tran, Du et al. (2015). "Learning spatiotemporal features with 3d convolutional networks". In: *Proceedings of the IEEE international conference on computer vision*, pp. 4489–4497.

Wan, Jun, Guodong Guo, and Stan Z Li (2016). "Explore efficient local features from RGB-D data for one-shot learning gesture recognition". In: *IEEE transactions on pattern analysis and machine intelligence* 38.8, pp. 1626–1639.

Wan, Jun et al. (2014). "3D SMoSIFT: three-dimensional sparse motion scale invariant feature transform for activity recognition from RGB-D videos". In: *Journal of Electronic Imaging* 23.2, pp. 023017–023017.

Wan, Jun et al. (2016). "Chalearn looking at people rgb-d isolated and continuous datasets for gesture recognition". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 56–64.

Wang, Chuanxu (2011). "An Algorithm of Unsupervised Posture Clustering and Modeling Based on GMM and EM Estimation." In: *JSW* 6.7, pp. 1201–1208.

Wang, Pichao et al. (2016). "Large-scale isolated gesture recognition using convolutional neural networks". In: *Pattern Recognition (ICPR), 2016 23rd International Conference on*. IEEE, pp. 7–12.