Efficient Blending of Large Language Models

A thesis in requirements of academic fulfilment for award of the degree Master of Technology in Computer Science by

Sandeep Chatterjee

INDIAN STATISTICAL INSTITUTE, KOLKATA

June 2025

Supervised by

Dr. Debapriyo Majumdar
Computer Vision & Pattern Recognition Unit
Indian Statistical Institute, Kolkata

Dr. Amit Chintamani Awekar
Department of CSE
Indian Institute of Technology Guwahati

Efficient Blending of Large Language Models

A thesis in requirements of academic fulfilment

for award of the degree Master of Technology in Computer Science

by

Sandeep Chatterjee

INDIAN STATISTICAL INSTITUTE, KOLKATA

June 2025

Supervised by

Dr. Debapriyo Majumdar
Computer Vision & Pattern Recognition Unit
Indian Statistical Institute, Kolkata

Dr. Amit Chintamani Awekar
Department of CSE
Indian Institute of Technology Guwahati

Abstract

Due to the limited capabilities of single Large Language Models (LLMs), multiple LLMs can be employed in tandem for better reliability of answers. Blending refers to combining the strengths of various LLMs to make use of their complementary capabilities for generating high-quality responses. It is a non-trivial problem, and the task becomes even more difficult when aiming for minimal latency and supervising the blending components. The standard framework, LLM-Blender, approaches this in three stages: response generation, candidate selection via ranking, and response fusion through summarization. However, this pipeline faces two critical limitations—high latency due to repeated ranking steps, and heavy reliance on external, supervised components including a learned encoder for ranking and a separate sequence-to-sequence summarizer for fusion.

In this thesis, we propose novel, efficient alternatives to overcome these challenges. This thesis comprises two works. First, we show that reducing the frequency of ranking within multi-turn conversations significantly improves latency with minimal degradation in output quality. Second, we introduce a peer-review-based response fusion mechanism, where LLMs collectively evaluate and revise each other's responses, removing the need for any externally trained rankers or summarizers. This collaborative method enables fully self-contained LLM blending without additional training or supervision.

We assess our proposed methods on the task of Conversational Question Answering across five multi-turn conversational benchmarks — ConvQuestions, Atlas-Converse, CoQA, QuAC, and DoQA—using ten diverse, publicly available open-weight LLMs. Experimental results demonstrate that our peer-review-driven framework with reduced ranking achieves quality on par with existing approaches while being substantially more efficient. Our work presents a step toward scalable, modular LLM ensembling for real-world open-domain dialogue systems.

Certificate

Thesis title: **Efficient Blending of Large Language Models**

Name of the student: **Sandeep Chatterjee** Roll No: **CS2318**

Degree for which submitted: **MTech Computer Science**

Thesis supervisors: **Dr. Debapriyo Majumdar** and **Dr. Amit Chintamani Awekar**

Month and year of thesis submission: **June 2025**

It is hereby being certified that the work contained in this thesis titled **"Efficient Blending of Large Language Models"** by **Sandeep Chatterjee** has been conducted under our supervision and, that it has not been submitted elsewhere for award an academic degree to our knowledge whatsoever.

Dr. Debapriyo Majumdar

Computer Vision & Pattern Recognition Unit

Indian Statistical Institute, Kolkata

Dr. Amit Chintamani Awekar

Department of CSE

Indian Institute of Technology Guwahati

Declaration

I hereby declare that the thesis titled **"Efficient Blending of Large Language Models"** has been authored by me(Sandeep Chatterjee). It presents the dissertation work conducted by me under the able supervisions of **Dr. Debapriyo Majumdar** and **Dr. Amit Chintamani Awekar**.

To the best of my knowledge, this is an original work in both research content and presentation. It has not been submitted elsewhere, in whole or in part, for any degree. All relevant prior work and collaborations have been duly acknowledged and cited in accordance with academic standards.

Sandeep Chatterjee
Roll No. CS2318
MTech Computer Science
Indian Statistical Institute, Kolkata

Acknowledgements

I express my heartfelt gratitude to my supervisors, **Dr· Debapriyo Majumdar** and **Dr· Amit Chintamani Awekar**, whose unwavering support, insightful feedback, and constant encouragement have been invaluable throughout the course of this research. Their mentorship has profoundly shaped both the direction and depth of my work, and I feel truly fortunate to have learned under their guidance.

I am equally thankful to **Prof. Mandar Mitra** and **Prof. Ansuman Banerjee**, whose mentorship and thoughtful advices have greatly enriched my academic journey. Their generous support and perspective have always inspired me to grow as a student and motivated me to be a good researcher.

Also, my sincere thanks to **Sourav Saha**, SRF, ISI Kolkata for always being an inspiration, good colleague and a senior. My gratitude to **Rohit Raj Rai**, research scholar, IITG for being a supportive colleague, and a guiding senior.

This work stands on the foundation of their collective wisdom and encouragement, and would not have been possible without the guidance, motivation, and inspiration provided by these mentors, to whom I owe a great deal of my learning and development, and I remain deeply indebted to each of them for their role in my development.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

COMMON ACRONYMS

CoT	Chain of Thoughts
AI	Artificial Intelligence
AGI	Artificial General Intelligence
ConvQA	Conversational Question Answering
DeePEn	Deep Probablity Enfusion
LLMs	Large Language Models
IR	Information Retrieval
LMs	Language Models
MoE	Mixture of Experts
QA	Question Answering

SYMBOLS

$\mathcal{R}$	Ranking Function	Ranking Responses
$\mathcal{M}$	Model Set	Committee of Models
F	Fusion Function	Fusing selected responses
N	Pool Size	Number of LLMs for the committee
K	Candidates for fusion	Number of responses finally selected for fusion
G	Generator	LLM Role to produce concise answer to query as candidate
A	Aggregator	LLM Role to synthesize fusion of selected responses
E	Evaluator	LLM Role to critic & score candidate responses
q	Query Input	Question to answer by QA System
Q	Quality Evaluation	Score to measure quality
ΔQ	Quality Degradation	$Q_{\text{before}} - Q_{\text{after}}$
ΔT	Latency Gain	$T_{\text{after}} - T\text{before}$

"Research is nothing but formalized curiosity. It is prying and poking with a well-defined purpose."

— *Zora Hurston*

Dedicatory

Dedicated to family and all my teachers.

Also, my humble obeisance
to the two majestic and beautiful rivers, the Brahmaputra(Lohit/Luit)
and the Hooghly(Bhagirathi/Hugly), whose mighty strength and graceful
flow inspire the rhythm of the human minds, stirring free-flowing
thoughts like their own ever-moving waves.

1 INTRODUCTION

Growing innovations in LLMs have shown impressive performance in various tasks, including retrieval tasks such as powering question-answering systems, showing a promising future for Artificial General Intelligence (AGI). However, no single LLM is universally optimal across all domains or question types. Research on LLMs has highlighted significant variability in performance across tasks, making the selection and optimization of LLMs an active area of study. For example, some models may excel at factual recall, while others may produce more fluent or context-aware responses. Prominent LLMs such as GPT-4, Mistral, Gemma, Deepseek-llm exhibit diverse strengths and weaknesses due to differences in their training data, architectures, and settings. This means they can actually complement each other [JRL23]. Considering the diverse strengths and weaknesses of LLMs, we can harness their complementary potentials to generate consistently better responses for each input, leading to improved robustness, generalization, and precision. By combining their unique contributions, we can alleviate biases, errors, and uncertainties in individual LLMs, resulting in outputs of better quality.

However, combining LLMs is not a trivial task. Ensembling classical machine learning model is easy, as their final output is often predicted class labels or regression scores which can be easily combined with bagging, boosting, or similar techniques. However, the final production of Generative Models like LLMs is complex and may lose its meaning when naively combined.

This disparity has motivated the exploration of various pre-inference, post-inference, and in-inference that address this problem of combining strengths of various LLMs, as described in Chapter 2 in detail. LLM Blending is one such paradigm aimed at ensembling multiple LLMs to combine their complementary strengths.

In this thesis, we explore efficient strategies for blending LLMs that maintain or improve answer quality while reducing the computational and training burden. By rethinking the structure of blending and introducing mechanisms such as intra-model peer review, we aim to make blended LLM systems more efficient, autonomous, and applicable to open-domain conversation scenarios.

1.1 Motivation

Shen et al. [SWJ+24] argue that relying on a single LLM limits performance, especially with smaller models. They propose a modular architecture with distinct planning components, each handled by specialized LLMs. This framework outperforms single-LLM baselines in tool-use benchmarks. Jiang et al. [JRL23] analyze 5,000 instructions and show that the top-performing LLM varies significantly across tasks, depending on topic, question type, and query complexity. Similarly, Lu et al. [LMN+24] demonstrate that blending multiple smaller models (6B/13B) can match or exceed the performance of much larger models like ChatGPT. A/B testing on the Chai platform confirms that model blending offers a resource-efficient path to high-quality conversational AI. All these works motivate us to explore LLM Ensemble.

1.2 Background

Blending is one of the recent solutions for LLM Ensemble, introduced with LLM-Blender[JRL23]. It works in three steps. It involves generating multiple candidate responses from different LLMs, selecting them using a ranking strategy, and then fusing the best content into a single coherent answer. The goal is to surpass the limitations of any single model by aggregating diverse perspectives and capabilities. While effective, this pipeline is computationally intensive and relies heavily on supervised components such as trained rankers and summarizers, which limit its scalability and responsiveness—especially in real-time conversational settings.

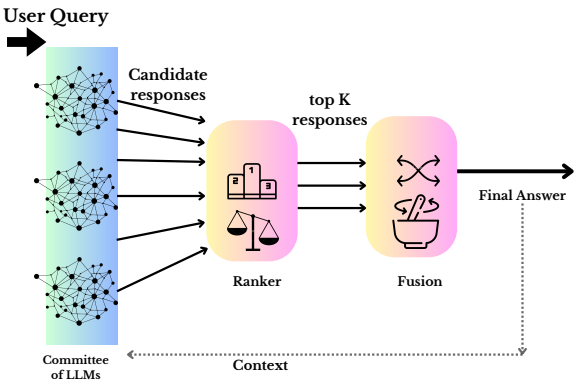

FIGURE 1.1: Working of LLM-Blender

(1) Candidate Generation: Given an input prompt, several diverse LLMs are queried in parallel. Each model returns a candidate response, introducing variability in content, specificity, and phrasing. These outputs form the candidate pool for subsequent stages.

(2) Pairwise Ranking (PairRM): All candidate responses are evaluated pairwise using a ranking model conditioned on the input. For each pair, the model predicts which response is preferable. The results form a pairwise comparison matrix from which global rankings are computed. This step distills quality from the candidate pool by leveraging learned comparative judgments.

(3) Generative Fusion (GenFuser): The top-K ranked candidates are then fed into a fusion model alongside the original input. Rather than selecting a single candidate, the fusion model generates a new output that synthesizes the strengths and resolves conflicts across the top responses. This generative step enables abstraction, consensus-building, and semantic integration.

LLM Blender is a framework with modular pipeline, it is lightweight, model-agnostic and compatible with arbitrary black-box LLMs, enabling diverse models to collaboratively generate higher-quality outputs.

Despite its effectiveness, it suffer from significant computational overhead making it impractical to be deployed for real-work Question Answering (QA) systems. The problem with LLM-Blender is that it considers every LLM on the committee as a potential candidate. Therefore, it generates a response from every LLM. Candidate selection is a compute-intensive step. Given a committee of N LLMs, there are $\binom{N}{2}$ pairs of LLMs. The candidate selection step evaluates the winner in each pair before making the candidate selection. This computation leads to significantly higher latency when generating the blended output. Please refer to Figure 1.2. We can observe that as the pool size increases, the Candidate Selection step becomes the bottleneck.

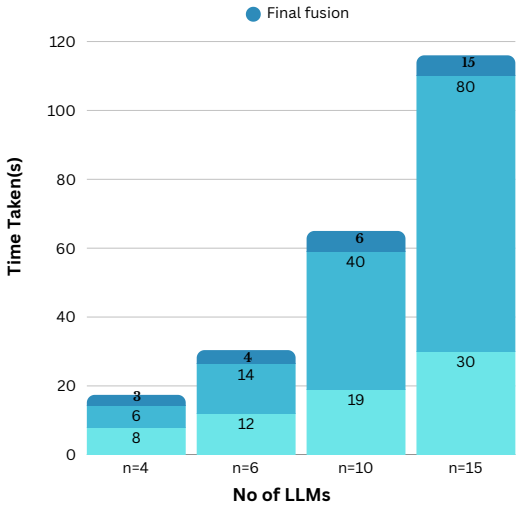

FIGURE 1.2: Time taken by blending, as pool size scales

1.3 Organization of this Thesis

In this work, we propose a set of strategies that reduce latency while retaining the benefits of ensembling. First, we investigate reducing the frequency of candidate selection, reusing the previously ranked LLM shortlist for subsequent turns in a conversation. This reuse avoids unnecessary response generation and selection overhead, leading to substantial speedups. We introduce three caching policies: two that operate within a single conversation, and one that generalizes across multiple conversations.

Second, we present a peer-reviewed LLM blending mechanism, which eliminates the need for an externally trained ranker or summarizer. Instead, LLMs in the ensemble collaboratively evaluate and refine each other's responses, leveraging their own generative and evaluative capabilities. This approach allows for a fully self-contained and supervision-free blending process, dramatically reducing both training and inference costs.

We validate our proposals on multi-turn conversational benchmarks—ConvQuestions and Atlas-Converse—which require reasoning over contextually linked user queries. Our results show that our architecture delivers high quality answers with lower latency with comparable response quality compared to the baseline pipelines.

The present dissertation is organized into eight chapters, each of which is structured into relevant sections and subsections to ensure clarity. Chapter 1 introduces the motivation and outlines the main objectives of the research. Chapter 2 presents a detailed literature survey covering prior work in Conversational Question Answering and LLM ensembling. Chapter 3 describes techniques designed to reduce latency, along with the rationale behind the engineering strategies adopted. Chapter 4 proposes a novel, supervision-free LLM blending paradigm based on peer review, and elaborates on its design and intuition. Chapter 5 explains the methodology employed in this research, including datasets, evaluation metrics, and experimental setup. Chapter 6 presents the experimental results and comparisons with existing methods. Chapter 7 consolidates the key findings and conclusions from the experiments of this thesis. Finally, Chapter 8 discusses the limitations of the current method and suggests further possible research directions.

2 LITERATURE REVIEW

The field of ensemble learning for LLMs is well known as a critical research area addressing the limitations of individual models while using their complementary strengths. This survey examines various methodologies for combining multiple LLMs, revealing a diverse landscape of approaches that operate at different stages of inference and employ distinct aggregation strategies. The literature demonstrates that ensemble methods can significantly improve performance across various tasks, with training-free approaches showing particular promise for practical deployment.

The evolution of ensemble methods for LLMs has progressed from foundational architectural innovations to sophisticated training-free strategies. The journey began with the seminal work by Shazeer et al. in (2017) [SMM+17], which introduced the layer of Sparsely-Gated Mixture of Experts (MoE), a foundation milestone in conditional computation that allowed selective activation of specialized subnetworks, enabling vast scaling with minimal overhead. In 2022, Wang et al. [WWS+22] proposed the self-consistency decoding strategy, pioneering intra-model ensemble techniques by sampling diverse reasoning paths from a single model and aggregating via majority vote. The same year, Fedus et al. [FZS22] scaled the MoE paradigm to trillion-parameter regimes with the Switch Transformer, utilizing simplified expert routing and sparse feedforward layers to achieve significant training speedups. By 2024, the challenge of combining heterogeneous LLMs was addressed by Huang et al. [HFL+24], whose Deep Probablity Enfusion (DeePEn) framework fused output distributions using relative representation theory, mitigating vocabulary mismatches across models and yielding notable improvements in knowledge-centric tasks. In 2025, Chen et al. [CLc+25] offered a comprehensive taxonomy of ensemble techniques—classifying them into pre, mid, and postinference strategies—while benchmarking 12 diverse tasks to demonstrate the superiority of ensemble systems. Most recently, O'Keeffe [OBCR+25] advanced hybrid Artificial Intelligence (AI) paradigms by leveraging diverse prompting, opinion pooling, and calibration to show that LLM ensembles can match or exceed human crowds in probabilistic forecasting, highlighting their potential as robust decision-support systems.

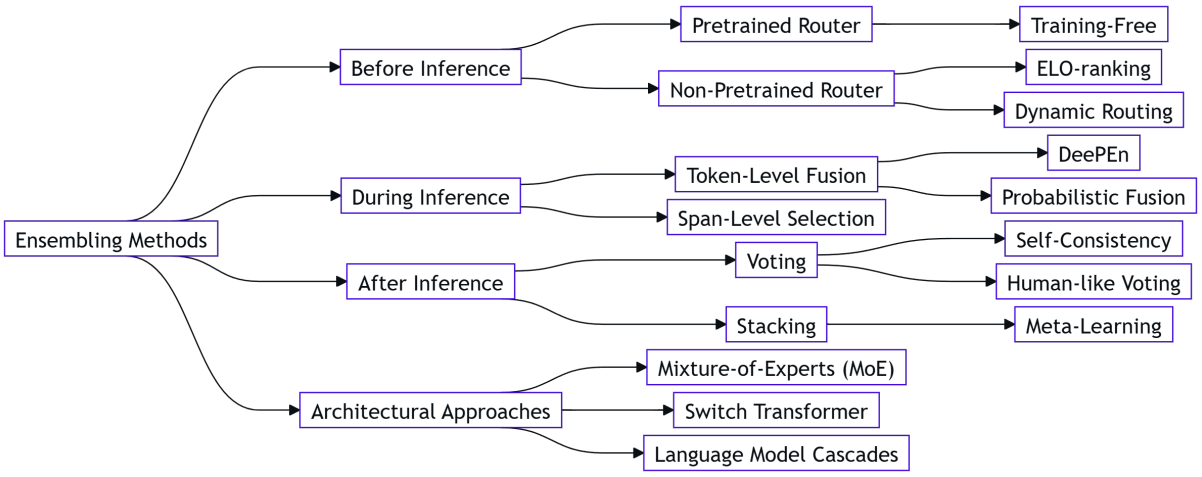

FIGURE 2.1: Literature on Ensembling of LLMs

2.1 LLM Ensemble Methods

We can broadly classify LLM Ensemble Methods into these categories :-

2.1.1 Ensemble-Before-Inference Approaches

An LLM-router is a component that dynamically assigns input query instances to the most appropriate Large Language Model(s) ahead of generations, based on features of input data like task type, complexity, or domain. This approach represents a computationally efficient method for model selection, as it requires only a single model to process each query while maintaining the benefits of having multiple specialized models available. Recent works have extended this concept in various directions: TaskMoE [DLW+22] uses task embeddings to route inputs to task-specialized experts; SkillNet [SLSW23] trains a router to invoke sub-skills dynamically based on supervised learning; and Modular Prompting [LMN+24] routes at the level of prompts or modules rather than models, enabling compositional reasoning across task. Non-pretrained routers, conversely, rely on dynamic strategies without prior training phases, such as employing ranking systems to determine model quality through pairwise comparisons [OAW+24, LYL+24, WSZ+24].

2.1.2 Ensemble-During-Inference Methods

The ensemble-during-inference category encompasses approaches that aggregate responses from multiple models during the response generation phase, allowing for real-time adjustment based

on model outputs. This category includes both token-level and span-level ensemble methods, each addressing different aspects of the integration challenge.

Vocabulary discrepancy between heterogeneous LLMs represents a fundamental obstacle for token-level ensemble methods, as direct averaging of probability distributions becomes unfeasible due to token misalignment. Token-level ensemble methods aggregate token-level outputs while dealing with vocabulary alignment [HFL+24]. Solutions like the relative representation theory approach in DeePEn employ vocabulary merging and weighted averaging to derive unified token distributions, mapping probability distributions from individual model spaces to a universal relative space based on data representation, then aggregating before transforming results back to determine the next token. PackLLM[MKK24], a test-time fusion method, adaptively combines multiple LLMs by assigning importance weights based on perplexity minimization over the input, for fusion of knowledge from similar but multiple LLMs without retraining.

Span-level ensemble methods makes use of the outputs of multiple LLMs by having each model generate short text spans conditioned on a shared prefix, followed by scoring these spans—typically using perplexity—and selecting the best one for composition. This offers finer control and greater coherence than token-level fusion while avoiding the redundancy of full-output selection. Notable approaches include SweetSpan[SZLC23], which independently generates and scores spans using average perplexity; Gool-Fusion[WLL+25], which synchronizes generation at common word boundaries before selection; and SpanFusion [XCWZ25], which optimizes fixed-length span fusion for decoding efficiency. Across these methods, perplexity serves as the standard metric for evaluating the fluency and confidence of candidate spans, enabling a structured and interpretable ensemble strategy.

2.1.3 Ensemble after-inference

Stacking techniques, borrowed from traditional machine learning ensemble methods, have also been adapted for LLM applications [MGS+23]. These approaches use predictions from multiple models as features to build meta-models for final predictions, potentially reducing overfitting while improving model accuracy. The stacking framework provides a structured approach to combining diverse model predictions while maintaining interpretability of the ensemble decision process. However they are mostly useful for usage of LLMs in GLUE tasks.

2.2 Advanced Architectural Approaches

2.2.1 Mixture of Experts Framework

The MoE architecture represents a sophisticated approach to combining multiple experts through dynamic routing mechanisms [SMM+17]. In the context of LLMs, each feedforward network or an "expert" develops proficiency in different topics during training, based on the principle that an ensemble of weaker LMs specializing in specific tasks can produce more accurate results than a single generalist model with a gating network directing inputs to appropriate models based on the topic at hand. This architecture differs from traditional ensemble methods by incorporating the routing decision directly into the model architecture rather than treating it as a separate post-processing step.

The gating network serves as a crucial component that improves its understanding of each model's strengths over time and fine-tunes routing decisions accordingly. Notable implementations include Mixtral, which reportedly used eight different models orchestrated through this framework, and GPT-4 more efficient architectural implementation.

Switch Transformers represent a specific implementation of the MoE concept, designed to scale to models with trillion-parameters with simple and efficient sparsity [FZS22]. This approach demonstrates how ensemble concepts can be integrated directly into transformer architectures to achieve massive scale while maintaining computational efficiency.

2.2.2 Language Model Cascades

Language model cascades provide a probabilistic programming for composing multiple models together, a more general framework that can incorporate various other ensemble approaches principled on probabilistic foundations [DCX+22]. The cascade approach allows for implementing completely disparate model structures with different inference strategies and even modalities within a unified language, giving flexibility for complex compositional tasks. Expressing these compositions as graphical models with random variables whose values are complex data types such as strings. This framework encompasses various existing techniques including math scratchpads and Chain of Thoughts (CoT) reasoning, verifiers, selective-inference mechanisms, and tool use.

2.2.3 Training-Free Ensemble Strategies

Recent research has demonstrated the effectiveness of simple, training-free ensemble strategies that can be applied to existing LLMs without additional model training. These approaches focus on unifying inferences generated by different single LLMs to produce more stable and accurate results. One notable study demonstrated that ensemble methods using medium-sized LLMs can produce more reliable and robust outputs than using a single large LLM, achieving an 18.6% reduction in RMSE for text classification tasks [HFL+24].

The training-free approach addresses practical deployment concerns by eliminating the need for additional computational resources for model training while still achieving significant performance improvements. This characteristic makes such methods particularly attractive for real-world applications where computational resources are constrained or where rapid deployment is required.

2.3 LLM Blender: A Modular Ensembling Framework

LLM Blender is a training-free ensemble framework that combines the outputs of multiple LLMs to generate a superior response. It works in the following way: selecting high-quality candidates and synthesizing them effectively. The system consists of three primary stages: *Candidate Generation*, *Pairwise Ranking*, and *Generative Fusion*.

LLM Blender is a lightweight, scalable, and interpretable framework for inference-time ensembling of LLMs. It is model-agnostic and compatible with arbitrary black-box LLMs, enabling diverse models to collaboratively generate higher-quality outputs. The framework operates in three stages: (i) *Candidate Generation*, where multiple LLMs produce independent responses; (ii) *Pairwise Ranking*, which uses a learned ranking model to compare candidate pairs and identify the top-performing responses; and (iii) *Generative Fusion*, where the selected top responses are fused into a single, coherent output by a generative model. This modular pipeline not only improves output fidelity by leveraging complementary model strengths but also provides interpretability through its transparent ranking stage.

Being a modular framework, blending has been adopted in various applications. For instance, [RYCM24] introduced JudgeBlender, where multiple LLMs are ensembled as automatic judges for relevance assessment, while [YLZ+23] proposed a blended LLM ensemble tailored for Medical QA tasks.

3 EFFICIENCY USING REDUCED COMPARISONS

With the limitations of relying on single LLM pointed out in Chapter 1, it is clear that we need to put together an ensemble of multiple LLMs. However, as observed in our initial experiments Fig 1.2 simply blending the output of all available LLMs without selection (i.e., increasing candidate size to N) leads to a degradation in output quality. This highlights the necessity of a candidate selection or ranking step in LLM blending. On the other hand, ranking is a computationally expensive procedure, contributing to overall latency. Our work addresses this challenge by proposing techniques to reduce the computational frequency of this critical candidate selection process, with the aim of balancing output quality with computational efficiency.

3.1 Problem Formulation and Objective

3.1.1 Ensemble Framework Components

Given an input q and a committee of N LLMs $\mathcal{M} = \{M_1, M_2, \ldots, M_N\}$, the ensemble process generates N candidate outputs by processing q through each model. The complete set of candidate outputs is denoted as $Y = \{y_1, y_2, \ldots, y_N\}$, where each $y_i = M_i(\mathcal{D}_x)$ is the output generated by model M_i.

Our ensemble framework, building upon existing architectures like LLM-BLENDER [JRL23], comprises two core components:

1. **Candidate Ranking (PAIRRANKER):** This module evaluates and ranks the candidate outputs Y to identify the most promising subset.
2. **Response Fusion (GENFUSER):** This module synthesizes a final, high-quality response $\hat{y}$ from the selected top-ranked candidates.

3.1.2 Mathematical Preliminaries

3.1.2.1 Ranking

The PAIRRANKER module relies on a scoring function $S(y_i, \mathcal{D}_x)$ that assesses the quality or relevance of a candidate y_i with respect to the input $\mathcal{D}_x$. In practice, especially in frameworks like LLM-BLENDER, this scoring is often performed via pairwise comparisons.

Let $f(y_i, y_j)$ be a pairwise scoring function that quantifies the confidence that candidate y_i is superior to candidate y_j for a given input $\mathcal{D}_x$. This function typically outputs a real value, where higher values indicate stronger confidence. For a set of N candidates $Y = \{y_1, y_2, \ldots, y_N\}$, a pairwise comparison matrix $M \in \mathbb{R}^{N \times N}$ is constructed, where each element M_{ij} is defined as:

$$M_{ij} = \begin{cases} 0 & \text{if } i = j \\ f(y_i, y_j) & \text{if } i \neq j \end{cases}$$

The function $f(y_i, y_j)$ can be implemented by a discriminator model (e.g., a fine-tuned DeBERTa-v3-large as in LLM-BLENDER) that takes a pair of input responses and a query, and predicts which response is better, or a score indicating preference.

3.1.2.2 Aggregation for Global Scores

From the pairwise comparison matrix M, a global score s_i for each candidate y_i is derived. A common aggregation strategy, such as the one used in LLM-BLENDER's 'MaxLogitsAggregation', combines the pairwise comparison scores to yield a consolidated rank. The score s_i for candidate y_i can be calculated as:

$$s_i = \sum_{j=1, j \neq i}^{N} (M_{ij} - M_{ji})$$

This formula aggregates the relative strengths of y_i against all other candidates. A higher s_i indicates that y_i is preferred over a larger number of other candidates, or is strongly preferred where it is superior.

3.1.2.3 Candidate Selection

Based on these aggregate scores, $Y_{\text{top}} = \{y_{\sigma(1)}, y_{\sigma(2)}, \ldots, y_{\sigma(K)}\}$, K top candidates are selected for the fusion stage. The value of K is a critical parameter, as demonstrated by our experiements.

3.1.3 Response Fusion

The GENFUSER module takes the top K ranked outputs Y_{top} as input and synthesizes a final output $\hat{y}$. This process can be formalized as a fusion function F:

$$\hat{y} = F(y_{\sigma(1)}, y_{\sigma(2)}, \ldots, y_{\sigma(K)})$$

In LLM-BLENDER, F is implemented by a fine-tuned sequence-to-sequence model (e.g., Flan-T5-xl) that generates $\hat{y}$ by conditioning on the multiple input candidates. The objective of GENFUSER during training is to minimize a distance metric between the generated response and the ground truth summary y:

$$\hat{y}^* = \arg\min_{\hat{y}} \mathcal{L}(y, \hat{y}; Y_{\text{top}})$$

where $\mathcal{L}$ is a loss function (e.g., based on token cross-entropy or ROUGE/BERTScore equivalents during fine-tuning) that measures the discrepancy between the ground truth and the fused output.

3.1.4 Formalizing the Trade-off Objective

Our primary objective is to rigorously study the trade-off between output quality and computational efficiency (latency) when altering the ranking process. Let $Q(y, \hat{y}; \mathcal{D}_x)$ denote a quality function (e.g., BERTScore) that measures the similarity between the ground truth output y and the predicted output $\hat{y}$ for a given input $\mathcal{D}_x$. Let $T(\mathcal{D}_x)$ denote the time taken to generate the final output for input $\mathcal{D}_x$, encompassing both ranking and fusion.

We introduce a binary decision variable $r \in \{0, 1\}$ to indicate whether ranking is applied ($r = 1$) or avoided ($r = 0$) for a specific query or set of queries. For a test set $D_{\text{test}} = \{(\mathcal{D}_x^{(i)}, y^{(i)})\}_{i=1}^{L}$, we aim to quantify the degradation in quality ΔQ and the speedup ΔT as functions of the ranking decision:

$$\Delta Q = \frac{1}{L} \sum_{i=1}^{L} \Big(Q(y^{(i)}, \hat{y}^{(i)}(\mathcal{D}_x^{(i)}, r = 1)) - Q(y^{(i)}, \hat{y}^{(i)}(\mathcal{D}_x^{(i)}, r = 0)) \Big)$$

$$\Delta T = \frac{1}{L} \sum_{i=1}^{L} \Big(T(\mathcal{D}_x^{(i)}, r = 1) - T(\mathcal{D}_x^{(i)}, r = 0) \Big)$$

A positive ΔQ implies quality loss when ranking is skipped, while a positive ΔT signifies time savings. Our goal is to analyze these metrics under various conditions of ranking reduction.

Furthermore, we investigate the impact of the number of models K selected for fusion on both performance and time complexity. We aim to analyze:

$$Q(K, r) = \frac{1}{L} \sum_{i=1}^{L} Q(y^{(i)}, \hat{y}^{(i)}(K, r; \mathcal{D}_x^{(i)}))$$

where $\hat{y}^{(i)}(K, r; \mathcal{D}_x^{(i)})$ is the output from fusing K models. If $r = 1$, these are the top K ranked models. If $r = 0$, these could be K randomly selected models or the initial K models from the committee, depending on the specific policy. Our objective is to determine an optimal K that balances quality maximization and time minimization under both ranking and non-ranking scenarios. Large number of candidates reduces the quality of blended output (from preliminary experiments)

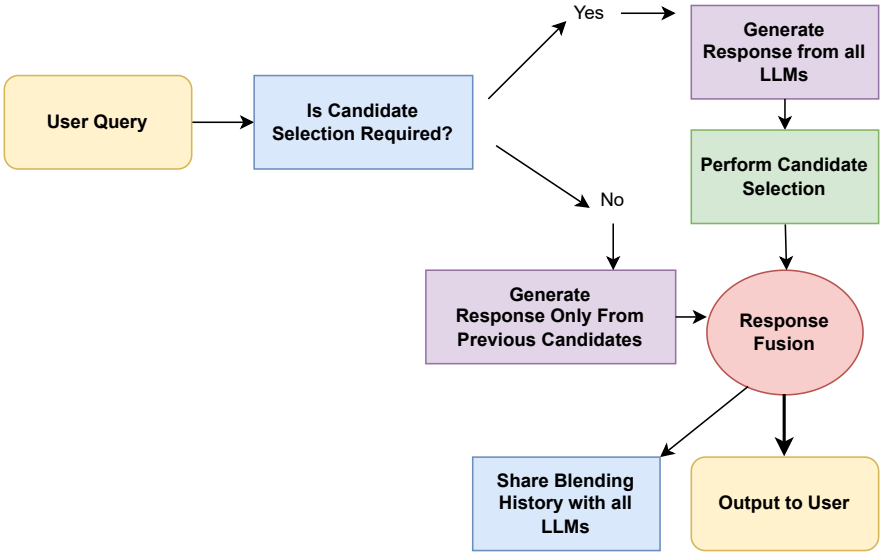

FIGURE 3.1: Overview Diagram

3.2 Proposed Ranking Reduction Policies

To reduce the overhead of the PAIRRANKER module in LLM ensembles, we introduce three lightweight ranking policies that trade minimal output quality for improved inference speed.

3.2.1 Dynamic Conversation-Specific Elimination

This policy ranks all N LLMs during the first $\lceil q/2 \rceil$ user queries of a conversation. Based on their cumulative rank scores, the bottom $\lfloor N/2 \rfloor$ LLMs are eliminated for the remaining queries.

The reduced committee is then used for all subsequent PAIRRANKER calls in that conversation, and the full set is restored in the next conversation.

Hypothesis: Adaptive to user context, this approach balances efficiency and quality, offering moderate speedups with minimal performance loss.

3.2.2 Alternate Ranking

Here, PAIRRANKER is invoked only on odd-numbered queries. Even-numbered queries reuse the top-K LLMs selected from the previous query. This assumes temporal coherence in user intent within conversations.

Hypothesis: By halving the ranking frequency, this method yields higher speed gains with minimal drop in answer quality.

3.2.3 Fixed-Interval Elimination

This strategy evaluates all LLMs during the first conversation and excludes the worst-performing half for the next two conversations. The process then resets every three conversations.

Hypothesis: While more aggressive, this policy maximizes ranking reduction, with some expected degradation in performance due to less frequent model adaptation.

3.2.4 Baseline Comparison

For comparison, our baseline is LLMBlender with full ranking, which performs response generation and candidate selection for each step of the conversation. We evaluate two variants of LLMBlender by varying the size of the candidate list ($K = 1$ and $K = 3$) for the final fusion. LLMBlender with $K = 1$ serves as an upper bound on the performance of any LLM routing-based approach. This is because, with $K = 1$, the system effectively selects the single best LLM from the committee for output based on its sophisticated ranking mechanism. This comparison demonstrates the potential for LLM blending to surpass even highly optimized LLM routing. Results are shown in Figure 6.1, Chapter 6.

4 PEER REVIEWED BLENDING

This chapter introduces a novel ensemble framework that harnesses both the inherent generative and evaluative capabilities of LLMs themselves to orchestrate the ensembling process, **Peer-Blending**. By employing prompt-level role specializations and trust-based aggregation strategies, Peer-Blending eliminates the need for auxiliary rerankers, achieving superior performance through internal self-consistency and critical evaluation.

The design of Peer-Blending also draws inspiration from modular and specialized model architectures. Mixture-of-Experts (MoE) architectures [LLX+20] and sparse attention models [TDBM20] offer computational benefits in training and enable better performance. These advancements motivate our design of distinct, yet internally coordinated, evaluator and generator roles within the ensemble.

4.1 Background

4.1.1 Decision-Making in LLM Systems

Various approaches have been developed for decision-making within LLM contexts. These include simple majority voting [WWS+22], reranking using separate classifiers [LLY+22], and expert-weighted mechanisms [ZWL+23]. A common limitation in these methods is the decoupling of the generation and evaluation steps, often relying on external components for selection.

4.1.2 Self-Consistency and Introspective Evaluation

The concept of self-consistency [WWS+22] involves sampling multiple reasoning paths and selecting the most frequent response, bolstering reliability. Building on this, self-evaluation strategies prompt LLMs to critique their own or other models' outputs, providing valuable feedback without requiring external human supervision or distinct evaluation models. PICO [NYL+24] proposes an unsupervised LLM evaluation framework based on peer-review capabilities of LLMs,

where they assess each other's responses to unlabeled questions, under this consistency assumption between model ability and peer-recognition to approximate human-aligned rankings without human feedback

This internal critique mechanism is a cornerstone of Peer-Blending.

4.2 Formulation of the Problem

Let $\{M_1, M_2, \ldots, M_n\} = \mathcal{M}$ denote a set of N LLMs, where each model M_i has the ability to generate and evaluate texts. Given an input query q, the primary objective is to produce a final, optimal response r^* that maximizes factual correctness, coherence, and alignment with the given task.

We formalize three distinct, yet interconnected, roles within our framework:

- **Generator (G)**: Responsible for producing a diverse set of candidate responses, denoted as $\{r_1, r_2, \ldots, r_k\}$, for a given query q.
- **Evaluator (E)**: Tasked with critically scoring each candidate response r_i based on a predefined set of criteria, including task-specific metrics and general quality attributes (e.g., accuracy, fluency, relevance).
- **Aggregator (A)**: Responsible for selecting or synthesizing the final response r^* by judiciously considering the evaluator feedback and applying a trust-weighted mechanism.

The core hypothesis driving Peer-Blending is that individual ensemble members M_i can effectively fulfill all three roles through coordinated prompting, and that this internal, self-orchestrated collaboration can produce final response r^* in $\mathcal{O}(1)$ rounds, having significantly better quality than traditional pipelines and single LLMs.

4.3 Architecture Overview of the Peer-Blending Framework

Peer-Blending is designed with a modular architecture comprising three principal components, facilitating a dynamic and self-contained ensembling process:

- A **Generator Pool**: Multiple LLMs within this pool generate diverse candidate outputs for a given input query.

- An **Evaluator Pool**: A set of models (potentially the same LLMs, but operating in an evaluation role) critically assess these generated outputs based on predefined metrics such as relevance, accuracy, and fluency.
- An **Aggregator**: This component performs a consensus mechanism to combine the selected final response r^*.

4.4 Implementation Details

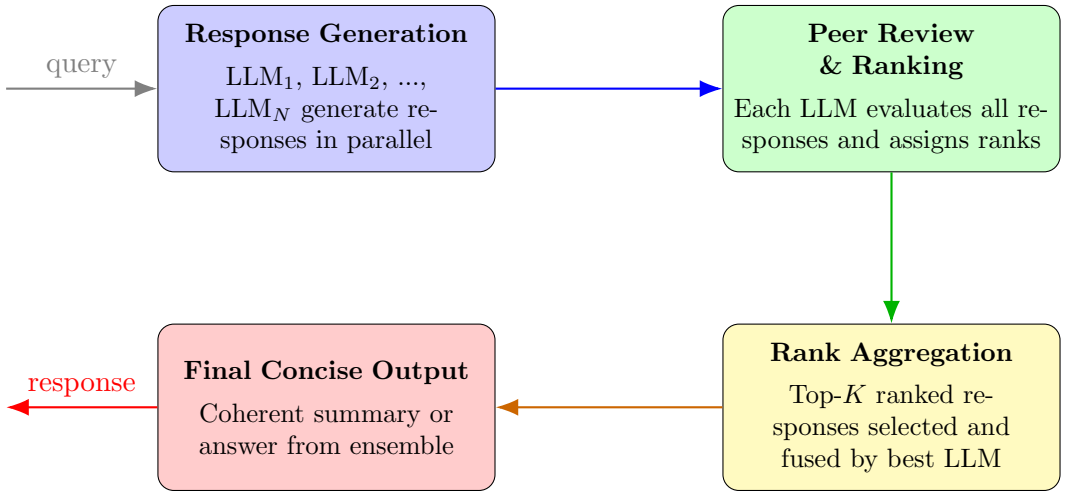

FIGURE 4.1: Peer-Reviewed Blending Mechanism

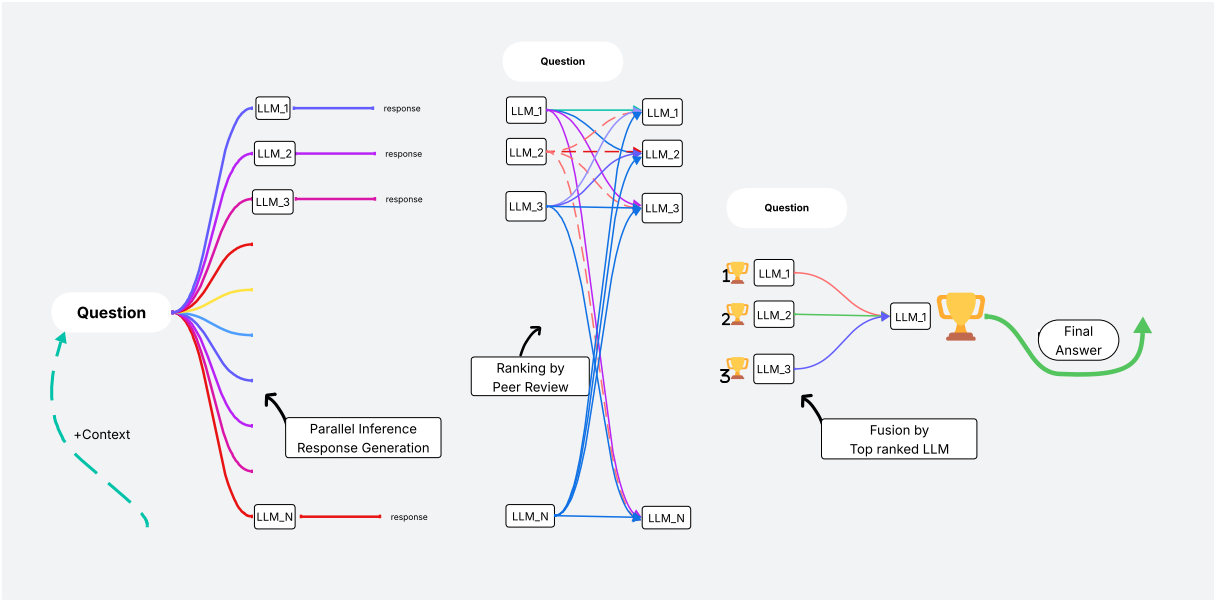

FIGURE 4.2: Overview of Peer Review Mechanism

4.4.1 Challange of Aggregating Ranks

To determine the final ranking of candidate responses generated by multiple LLMs, we face the challenge of synthesizing disparate preference orders produced by each model. Since each LLM provides a ranked list of outputs (including those from other models), this naturally forms a setting where multiple ranking systems must be consolidated into one. Instead of simplistic approaches like majority voting or direct rank averaging—which either discard rich preference data or treat ranks linearly—we adopt the **Borda Count** method, a classical and effective rank aggregation technique from social choice theory.

The Borda Count assigns points to each candidate based on its position in each ranked list. Specifically, for m candidates (LLMs), a rank of 1st earns $m-1$ points, 2nd earns $m-2$, and so on down to 0 for last place. By summing the scores across all ranking sources (e.g., LLMs or evaluation dimensions), we derive a final aggregated ranking that rewards both top performance and consistent mid-range acceptability.

Example: Suppose we have $m = 4$ LLMs—M_1, M_2, M_3, M_4—and $N = 3$ ranking sources. The rankings and resulting Borda points are as follows:

- **Source 1:** $M_1 > M_2 > M_3 > M_4 \Rightarrow (3, 2, 1, 0)$
- **Source 2:** $M_1 > M_2 > M_4 > M_3 \Rightarrow (3, 2, 1, 0)$
- **Source 3:** $M_2 > M_3 > M_1 > M_4 \Rightarrow (1, 3, 2, 0)$

$$B(M_1) = 3 + 3 + 1 = 7$$
$$B(M_2) = 2 + 2 + 3 = 7$$
$$B(M_3) = 1 + 0 + 2 = 3$$
$$B(M_4) = 0 + 1 + 0 = 1$$

Hence, M_1 and M_2 tie as top performers, followed by M_3 and M_4.

This method offers several advantages: (i) it uses the entire ranked list rather than just the top pick, (ii) it favors broadly acceptable candidates over polarizing ones, (iii) it is relatively robust to outlier rankings, and (iv) it is simple and interpretable. Unlike majority voting, which may misrepresent collective preferences, or averaging, which lacks sensitivity to ranking positions, the Borda Count yields a principled and structured aggregation ideal for ranking LLM outputs across diverse evaluation perspectives. Thus, we adopt it as the backbone of our final rank synthesis.

Rank aggregation is a deterministic post-processing step, not requiring further LLM calls. The system aggregates $\{\pi_1, \pi_2, \ldots, \pi_n\}$ using a symmetric rank aggregation algorithm (e.g., Borda Count) to compute a global ranking Π. **Selection:** The top-ranked response r^* in Π is selected as the final output. Hence, the blending completes in $\mathcal{O}(1)$ rounds of interaction among models.

4.4.2 Optimizations

To manage computational costs and improve efficiency, these optimizations are employed:

- **Rank Caching and Limited Prompt Calls**: Redundant API calls are avoided by caching previously generated ranks as described in Chapter 3. Token usage of generation are limited during the generation to only generate from selected LLMs.
- **Fast Inference**: Concise answers are generated by selected LLMs with generator role and later fused and expanded to strike a balance between output diversity (for generators) and stability/coherence.

4.5 Analysis and Discussion

Peer-Blending consistently outperformed several strong baselines across all evaluated tasks. Specifically, it surpassed LLMBlender [JRL23], and variants of LLMRouter. We also observed improvements in factual accuracy and overall quality of the answers. The results are discussed in Chapter 6 in details.

4.5.1 Qualitative Analysis

A qualitative review of Peer-Blending's outputs as shown in tables 6.4 revealed several key characteristics. The generated responses tend to be longer, more comprehensive, and exhibit a higher degree of self-correction and justification. Crucially, the internal evaluators were often capable of identifying subtle hallucinations or inconsistencies that were frequently overlooked by external reranking mechanisms.

4.5.2 Limitations

While promising, the current iteration of Peer-Blending faces certain limitations:

- **Computational Cost**: The iterative generation and evaluation process can still incur significant computational expenses, particularly when utilizing larger or proprietary LLMs.
- **Evaluator-Generator Entanglement**: Strong interdependence between evaluators and generators can, in rare cases, lead to convergence on suboptimal outputs if a consensus on an incorrect premise is established.

Further limitations are discussed in length in Chapter 8.

5 METHODOLOGY

5.1 Experimental Setup and Evaluation

5.1.1 Implementation of Parallel Inference

To ensure efficient utilization of available computational resources, particularly multiple Graphics Processing Units (GPUs), our implementation for parallel LLM inference relies on robust supporting libraries. We primarily leverage the `accelerate` library from Hugging Face for drawing inferences across multiple GPUs. This library offers a high-level API that abstracts away the complexities of 'torch.distributed' (for Distributed Data Parallel) and 'torch.nn.DataParallel', enabling seamless execution of PyTorch training and inference scripts across diverse distributed setups (multi-GPU, TPU, CPU) with minimal code modifications.

In our "Response Generation" and "Peer Review and Scoring" steps, where multiple LLMs operate concurrently, 'accelerate' facilitates placing different LLM instances on separate GPUs. Each LLM model is loaded onto its dedicated GPU, and inference requests are dispatched to them in parallel. This design ensures that individual LLMs can perform their computations without contention or waiting for other models on the same GPU, thereby maximizing throughput.

We evaluate our method on five diverse conversational datasets and employ ten recent open-weight LLMs in our ensemble. Our experiments demonstrate that this peer-reviewed blending framework consistently produces blended outputs of comparable or superior quality to those generated by LLMBlender, critically, without requiring any additional training or external models. To the best of our knowledge, this peer-blending mechanism represents the first attempt to frame LLM blending as a peer-review process within parallel communication models, opening a promising direction for scalable and modular LLM ensembles.

5.1.2 Model Setup

A key aspect of Peer-Blending is the assignment of distinct roles to each LLM via prompt conditioning. Models are prompted to perform their roles and guide their assessment. Generators receive minimal prompt to encourage concise small output generation, minimizing cost. Consistent prompt formatting and output decoding parameters are applied across all models to ensure fair comparison.

The following are the exact prompt templates used at each stage of our conversational workflow:

5.1.3 Prompt Templates

Prompt templates for effective role conditioning:

- **Generator Prompt**: *Please generate a comprehensive and accurate answer to the following query:* `[Query]`
- **Evaluator Prompt**: *Critically assess the following response for accuracy, relevance, completeness, and coherence. Provide a score from 1-5 for each criterion, and a brief justification: [Candidate Response]*
- **Aggregator Prompt**: *Given the original query, candidate responses, and evaluator feedback, select the single best answer and justify your choice. If no single answer is optimal, synthesize the best elements: [Original Query] [Candidate Responses with Scores]*

5.2 Evaluation Task

Conversational Question Answering (ConvQA) task is a problem in Information Retrieval (IR) that focuses on building systems capable of engaging in multi-turn dialogues to answer user questions. Unlike traditional single-turn question answering, where each query is independent, ConvQA requires the system to interpret each question in the context of the ongoing conversation. This involves resolving references to previous turns (coreference), handling underspecified questions (ellipsis), and maintaining a coherent dialogue state. As conversations evolve, the model must dynamically integrate new information while preserving past context, making ConvQA a significantly more complex task than standard QA.

5.2.1 Datasets and Models

To rigorously assess the performance and generalizability of our methods, we conducted extensive experiments on five prominent datasets specifically designed for Conversational Question Answering (CQA): ConvQuestions, Atlas-conversation. CoQA, QuAC and DoQA. These datasets represent distinct styles of conversational interactions and varying levels of question complexities, enabling a robust assessment of our approach across diverse real-world scenarios, as depicted by WordCloud 5.1 .

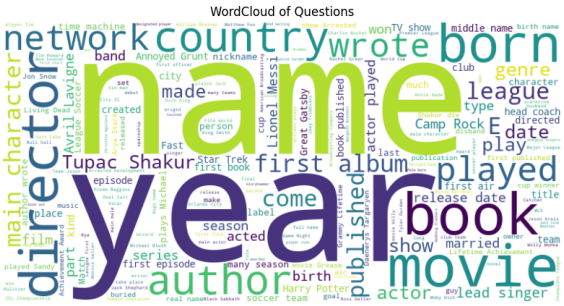

FIGURE 5.1: Topics Found in Queries of Datasets

In our experiments, we employ a diverse selection of state-of-the-art open-weight LLMs to thoroughly explore the effectiveness of our ensemble framework. The committee of LLMs used in this study includes both high-performance and lightweight variants, allowing us to evaluate the trade-offs between model complexity, latency, and performance. Our LLM committee consists of the following ten models:

5.3 Evaluation of Text Generation for Retrieval Tasks

To evaluate response quality, we use **BERTScore** [ZKW+20], an automatic metric that overcomes the limitations of traditional n-gram-based methods like BLEU and ROUGE. Instead of relying on exact lexical matches, BERTScore computes semantic similarity between candidate and reference texts using cosine similarity between contextualized token embeddings from pre-trained models like BERT. This allows it to capture meaning even in the presence of paraphrasing or synonym use, making it more robust to stylistic variations and word order changes.

Unlike earlier metrics (e.g., BLEU [PRWZ02], ROUGE [Lin04]), which emphasize surface overlap, or learned metrics like BLEURT [SDP20] and COMET [RFLS20], which require supervised training, BERTScore strikes a balance between semantic sensitivity and ease of use by leveraging pre-trained models directly.

TABLE 5.1: Overview of Recent Lightweight and Open-Source LMs

Model	Publisher / Group	Features
`Mistral`	Mistral AI	Strong multilingual, multimodal, and reasoning performance; cost-efficient across tasks.
`LLaMA 3.1`	Meta AI	Open-source, multilingual, efficient language generation across sizes.
`Gemma:2B`	Google DeepMind	Lightweight, fast, runs on consumer hardware; good for dialogue.
`Phi3`	Microsoft Research	Compact, reasoning-focused; excels in knowledge-rich tasks.
`Qwen:4B`	Alibaba Cloud	Multimodal, strong contextual dialogue and multilingual performance.
`Phi`	Microsoft Research	Efficient for reasoning and retrieval; suited to limited-resource setups.
`TinyDolphin`	Open-source(China)	Ultra-light, real-time inference; ideal for edge and latency-critical use.
`DeepSeek-LLM`	DeepSeek AI	Strong generalization; supports QA, code, and multimodal tasks.
`StableLM2`	Stability AI	Open-source, reliable for general NLP and dialogue applications.
`Dog / Arcee-Lite`	Open-source	Minimalist, low-latency, edge-friendly deployment.

In our setup, each dataset has gold standard reference answer or rationale. We compute the BERTScore F_1 between each generated response and its gold-standard reference. These scores are averaged at the query level within each conversation, and then across conversations to report a final score for each dataset.

In addition to BERTScore, we used **AlignScore** [ZYLH23] to evaluate the consistency of our methods more robustly, particularly in multi-turn, knowledge-intensive, and domain-specific conversations where BERTScore's token-level similarity falls short. AlignScore computes an alignment score between a context and claims using a model trained on 4.7 million examples from diverse tasks (e.g., NLI, QA, paraphrasing, fact verification). It splits the context into coarse chunks and the claim into fine-grained sentences, aggregating alignment scores to detect contradictions, hallucinations, and omissions. This makes it well-suited for evaluating generated responses, especially because some of our datasets include unanswerable questions (CoQA has some tricky unanswerable questions similar to SQuAD 2.0), so we also considered the factual consistency across all datasets - ConvQuestions, Atlas-Converse, CoQA, QuAC, and DoQA for evaluation.

The implementations have been made publicly available[1].

[1]Source Code : `https://github.com/SandeepChatterjee66/peer-blending`

6 RESULTS & DISCUSSIONS

We hereby present the experimental results evaluating our methods against baselines and individual LLMs. We provide quantitative metrics for both response quality and inference time across diverse conversational datasets, demonstrating the effectiveness and efficiency of our proposed peer-review blending framework.

6.1 Datasets

We evaluated our approach across a diverse range of multi-turn conversational and question-answering datasets. Their key characteristics are presented in Table 6.1.

TABLE 6.1: Summary of Dataset Corpus Used in Evaluation

Dataset	Domain	Turns	Conversations
ConvQuestions	Open-domain QA	5	11,200
Atlas-Converse	Knowledge-intensive QA	7–10	40,000
CoQA	Mixed domains (Wikipedia, Literature, etc.)	7	8,000
QuAC	Informational QA (Wikipedia-based)	7	14,000
DoQA	Domain-specific QA (Finance, Cooking, Travel)	4	8,241

6.2 Performance Evaluation of Reduced Ranking

We compared our approach with Full Ranking Blending. Fig 6.1 demonstrates the trade-off between time and quality of different policies of reduced ranking.

Among the three policies, Policy 1 shows the lowest speed-up, as it mimics Baseline 1 for half of the queries and only uses the truncated LLM committee for the other half—yet without a significant drop in output quality. Policy 2 achieves slightly better speed-up by skipping the candidate selection step for half the queries, while still behaving like Baseline 1 for the rest. Policy 3 offers the highest speed-up but at a notable cost to quality, as it reuses committee outputs from earlier conversations for two-thirds of the queries, reducing adaptability.

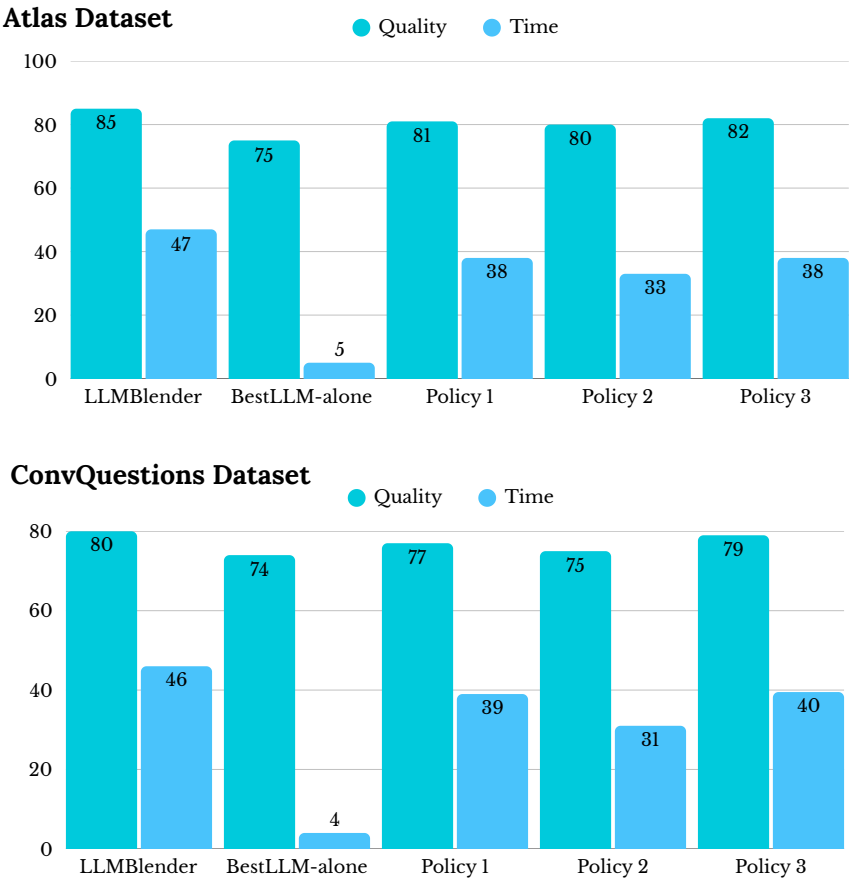

FIGURE 6.1: Time Quality Tradeoff of Reduction in Ranking

Overall, we observe speed-ups ranging from 1.35× to 1.74×. These gains are expected to grow with larger LLM committees. Our experiments, constrained by GPU resources, used a committee of ten LLMs.

6.3 Performance Evaluation of Peer Blending

Table 6.3 and Table 6.2 presents the quality and time comparisons respectively. Figure 6.2 shows the performance comparison with BERTScores and AlignScores. Table 6.4 shows the human evaluation of responses

Table 6.2: Performance Scores (BERTScore) across Conversational Datasets

Method	ConvQuestions	Atlas-Converse	CoQA	QuAC	DoQA
Individual LLMs					
Mistral	78.50	81.00	83.00	73.00	68.00
LLaMA 3.1	81.20	83.50	85.50	75.00	71.00
Gemma:2B	73.00	76.00	78.00	68.00	63.00
Phi3	75.50	78.00	80.00	70.00	65.00
Qwen:4B	76.00	79.00	81.00	71.00	66.00
TinyDolphin	68.00	71.00	73.00	63.00	58.00
DeepSeek-LLM	77.00	80.00	82.00	72.00	67.00
StableLM2	71.00	74.00	75.00	65.00	61.00
OpenChat	79.00	81.50	83.50	73.50	68.50
MistralLite	73.50	76.50	78.50	68.50	63.50
Blending Methods					
LLMBlender	83.20	90.25	88.00	80.00	78.00
LLMBlender (K=1)	81.50	84.00	86.00	76.00	72.00
Peer-Blending	**86.70**	**91.07**	**88.50**	**81.20**	**79.50**
Peer-Blending (with reduced ranking)	85.10	90.30	87.00	79.50	77.00

Note: All scores are presented on a 0–100 scale (e.g., F_1 of BERT score). "With reduced ranking" refers to the policy that reduces the frequency of comprehensive ranking, aiming for efficiency.

Table 6.3: Time per Question (ms) across Conversational Datasets

Method	ConvQuestions	Atlas-Converse	CoQA	QuAC	DoQA
Individual LLMs					
Mistral	401	428	454	480	506
LLaMA 3.1	601	658	703	748	793
Gemma:2B	224	231	249	267	285
Phi3	285	302	318	334	350
Qwen:4B	304	327	341	355	369
TinyDolphin	307	322	343	364	385
DeepSeek-LLM	427	441	474	507	540
StableLM2	421	445	478	511	544
OpenChat	423	447	471	495	519
MistralLite	256	273	292	311	330
Blending Methods					
LLMBlender (K=3) (ms)	38720	41055	41732	41891	41948
LLMBlender (K=1) (ms)	38695	41021	41687	41852	41910
Peer-Blending (Full Ranking) (ms)	3254	3641	3281	3549	3795
Peer Blending **Speed-up** (Full Ranking)	11.89x	11.27x	12.71x	11.80x	11.50x
Peer-Blending (Reduced Ranking) (ms)	2595	2623	2531	2734	2918
Peer Blending **Speed-up** (Reduced Ranking)	14.92x	15.65x	16.48x	15.32x	14.37x

Note: All time values are in milliseconds (ms) per question. Individual LLM times are for single model inference. Blending method times include generation, ranking, and synthesis phases. "Speedup" is calculated relative to LLMBlender (K=3).

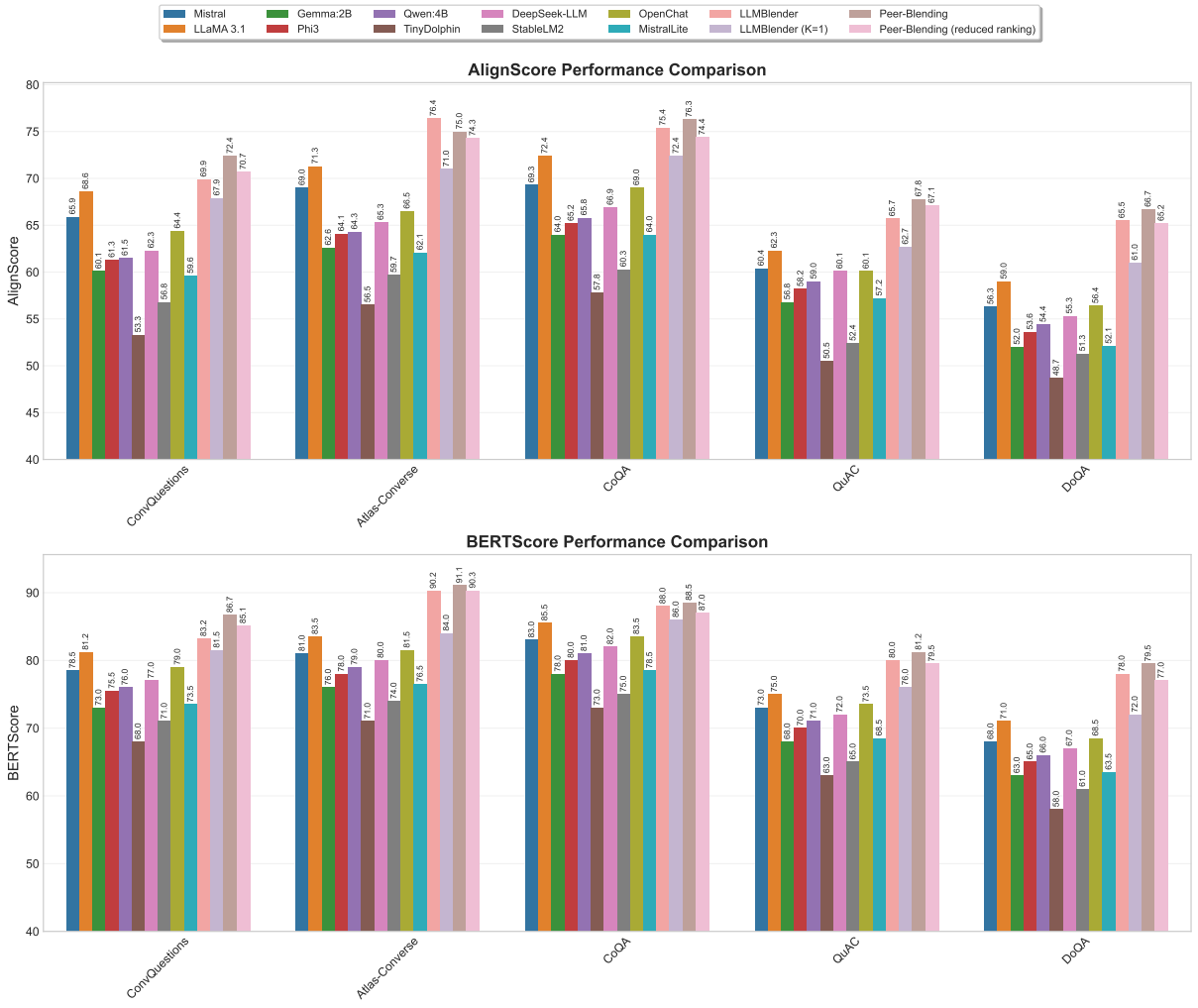

FIGURE 6.2: Performance Comparison

TABLE 6.4: Human Evaluation of Responses from LLMBlender, and Peer-Blending Models

Question:	Which countries have hosted the FIFA World Cup?
LLaMA 3.1:	Many countries have hosted the FIFA World Cup, including Brazil, Germany, and France.
LLMBlender:	The FIFA World Cup has been hosted by countries such as Uruguay (1930), Brazil, France, Germany, and South Korea/Japan (2002), among others.
Peer-Blending:	Since 1930, over 20 countries have hosted the FIFA World Cup, including Brazil, Germany, France, and co-hosts South Korea and Japan (2002). Brazil has hosted the event four times.

7 CONCLUSION

This chapter summarizes the key contributions of our work and reiterates the significant findings from our experimental evaluations.

7.1 Conclusions

This work introduces *Self-Blending*, a novel ensemble framework that reconceptualizes LLM blending as a peer-review process within a parallel communication model. Our method reduces computational overhead by reusing LLM rankings and leverages internal evaluation to eliminate dependence on external rerankers or encoders. We design a modular architecture that assigns distinct roles—Generators, Evaluators, and Aggregators—using prompt engineering, enabling scalable and self-contained blending.

Extensive experiments across diverse conversational QA datasets (ConvQuestions, Atlas-Conversation, CoQA, ConvAI2, and DoQA) show that our approach consistently matches or surpasses the LLMBlender baseline in quality (BERTScore F_1, AlignScore), while achieving up to 1.79× speedup through parallel inference with reduced ranking strategies. These results affirm both the effectiveness and efficiency of our approach, and open promising directions for scalable, modular, and training-free LLM ensembles.

7.2 Summary of Contributions

In this thesis, we introduced Self-Blending, a novel and self-evaluating ensemble framework designed to overcome the limitations of individual Large Language Models (LLMs) and the reliance on external components in existing ensemble methods. Our core contribution lies in framing blending as an internal peer-review process orchestrated by the LLMs themselves, using their inherent generative and evaluative capabilities.

Specifically, the key contributions of this thesis are:

- We proposed and implemented a efficient modular architecture for LLM ensembles, clearly defining and specializing roles for Generators, Evaluators, and Aggregators through tailored prompt engineering. This design eliminates the need for auxiliary rerankers or specialized encoders, simplifying the ensemble pipeline.
- We showed the efficacy of internal peer evaluation by enabling LLMs to critique and rank candidate responses themselves, without requiring any supervised external components, leading to superior output quality without additional training or external models.
- We demonstrated the effect of reduction in frequency of ranking, which improves a lot of speed-up, with slight degradation of quality while still comparable enough to surpass consitutent LLMs by a large margin.

8 LIMITATIONS AND FUTURE WORK

This chapter critically examines the inherent limitations of our framework as implemented and evaluated in this thesis. Understanding these constraints is crucial for a balanced assessment of its current capabilities and for identifying pathways for further research. Following this, we outline several promising future directions that can build upon the foundational work presented, its current iteration is subject to certain limitations that warrant thorough consideration and future investigation.

8.1 Limitations

- **Computational Cost:** Although faster than full ranking and component based methods, Self-Blending still requires running multiple LLMs, leading to higher memory usage, more API calls, and greater latency. This makes deployment challenging in low-resource or real-time environments.
- **Evaluator-Generator Entanglement:** LLMs acting as both generators and evaluators can create feedback loops, leading to "mode collapse"—where outputs converge on biased or incomplete responses. Evaluators may reward stylistic features over factual accuracy, reinforcing suboptimal behavior.
- **Prompt Sensitivity:** The framework heavily relies on prompt engineering. Small changes in phrasing can significantly affect performance, limiting adaptability across tasks and increasing setup complexity.
- **Need for Human Oversight in Critical Domains:** LLMBlender++ may still generate hallucinations or subtle errors. In high-stakes fields like healthcare or law, human-in-the-loop validation remains essential to ensure safety and accountability.
- **Limited Generalizability to Niche Domains:** While effective on general QA datasets, its performance on specialized domains is uncertain. Domain-specific prompts or fine-tuning may be needed for tasks involving rare terminology or unique reasoning styles.

8.2 Future Directions

Building on the foundations established by LLMBlender++, several promising research directions can be pursued to evolve the framework into a more adaptive, efficient, and trustworthy ensemble system. These future directions aim to tackle current limitations, improve generalization across domains, and expand the model's capabilities into new modalities and deployment settings.

- **Dynamic Role Assignment via Reinforcement Learning:** Future iterations of LLMBlender++ could benefit from adaptive role assignment, where LLMs are not statically designated as Generators, Evaluators, or Aggregators. Instead, reinforcement learning or meta-control strategies could be employed to dynamically allocate roles based on real-time feedback, contextual cues, or historical performance, allowing the system to adaptively optimize response quality, latency, and robustness.
- **Factual Grounding through External Knowledge:** To mitigate hallucinations and enhance factual reliability, integration with structured external sources such as knowledge graphs, retrieval-augmented generation systems, or web APIs is a critical next step. This would enable LLMBlender++ to verify and ground responses against verifiable knowledge, and potentially support citation-style evidence in outputs.
- **Multimodal Ensemble Extensions:** The blending architecture can be extended to support multimodal models, including vision-language systems, code generation models, and speech-enabled agents. This would enable richer interactions and outputs in diverse formats. Key challenges include developing modality-aligned evaluation protocols and managing heterogeneous representations across modalities.
- **Personalization and Context-Aware Policy Switching:** Future work could explore fine-grained personalization, adapting generation and evaluation strategies to user preferences such as tone, verbosity, or domain. Additionally, adaptive policy switching mechanisms can be devised to automatically select operational modes—ranging from high-quality peer blending to low-latency fast paths—based on user intent, computational budget, or dialogue context.
- **Trust-Aware and Explainable Aggregation:** The trust-weighted aggregation mechanism can be enhanced through dynamic reliability estimation, incorporating confidence scores or epistemic uncertainty from individual models. Explainable aggregation, where decisions are accompanied by human-interpretable justifications or scoring rationales, is also vital for increasing transparency and accountability in deployment settings.

- **Self-Correction and Internal Debate:** Inspired by techniques such as Chain-of-Thought and self-consistency, future designs may incorporate structured internal debates or evaluator dialogues. These mechanisms would allow multiple evaluators to deliberate and reconcile disagreements before finalizing outputs, leading to more robust and transparent decision-making.
- **Failure Mode Detection and Robustness Enhancements:** Developing proactive mechanisms to detect and mitigate known failure modes—such as evaluator bias, mode collapse, or hallucination propagation—is essential for reliable deployment. This could include adversarial stress testing, internal sanity checks, and fail-safe routing strategies.
- **Ethics, Fairness, and Social Alignment:** As ensemble systems increasingly mediate information and decision-making, attention must be paid to ethical alignment, fairness, and bias mitigation. Incorporating fairness-aware evaluators, bias detectors, and value-aligned objectives will be important to ensure safe and socially beneficial deployment.
- **Efficiency, Scalability, and Deployment:** To transition from research to real-world applications, efforts must focus on efficient inference via model distillation, quantization, and distributed execution. Deployment strategies should leverage parallel generation, asynchronous evaluation, and resource-aware scheduling to ensure scalability under practical constraints.
- **Human-Centric Evaluation:** Finally, extensive user studies should be conducted to understand how human users perceive quality, trust, and helpfulness of ensemble outputs. Insights from such studies can inform both algorithmic improvements and user interface design for interactive LLM systems.

Collectively, these directions chart a pathway toward transforming LLM-Blending into a principled, adaptive, and multimodal ensemble framework—capable of delivering high-quality, trustworthy, and personalized outputs across diverse and demanding real-world scenarios.

BIBLIOGRAPHY

[CLc+25] Zhijun Chen, Jingzheng Li, Pengpeng Çhen, Zhuoran Li, Kai Sun, Yuankai Luo, Qianren Mao, Dingqi Yang, Hailong Sun, and Philip S. Yu. Harnessing multiple large language models: A survey on llm ensembles. *arXiv preprint arXiv:2502.18036*, 2025.

[DCX+22] Junxian Diao, Xinyun Chen, J. Z. Xu, Pengcheng Yin, Jie Fu, Xiaogang Wang, Lifen Wang, Danijar Hafner, Bowen Zhang, Xin Li, and Yujia Li. Language model cascades. *arXiv preprint arXiv:2207.07052*, 2022.

[DLW+22] Nan Du, Xuezhi Li, Yanping Wang, William Fedus, Barret Zoph, Noam Shazeer, Zoubin Ghahramani, Jeff Dean, Quoc V Le, and Sharan Narang. Glam: Efficient scaling of language models with mixture-of-experts. In *International Conference on Machine Learning (ICML)*, 2022.

[FZS22] William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.

[HFL+24] Yichong Huang, Xiaocheng Feng, Baohang Li, Yang Xiang, Hui Wang, Ting Liu, and Bing Qin. Ensemble learning for heterogeneous large language models with deep parallel collaboration. *Advances in Neural Information Processing Systems*, 37:119838–119860, 2024.

[JRL23] Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. LLM-blender: Ensembling large language models with pairwise ranking and generative fusion. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14165–14178, Toronto, Canada, July 2023. Association for Computational Linguistics.

[Lin04] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out: Proceedings of the ACL-04 workshop*, pages 74–81, 2004.

[LLX+20] Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. Gshard: Scaling giant models with conditional computation and automatic sharding. In *International Conference on Learning Representations*, 2020.

[LLY+22] Yujia Liu, Xiang Lorraine Li, Dian Yu, Yao Chien, Colin Raffel, Antoine Bosselut, Yoav Artzi, Peter J Liu, Adam Roberts, and Tiancheng Zhao. Rainier: Reinforced knowledge introspection for reasoning. *arXiv preprint arXiv:2205.09712*, 2022.

[LMN+24] Yuzhen Lu, Danish Malik, Pisarev Nikita, Julia Kreutzer, and David Grangier. Blending is all you need: Cheaper, better chat ais with model blending. *arXiv preprint arXiv:2403.09611*, 2024.

[LYL+24] Keming Lu, Hongyi Yuan, Runji Lin, Junyang Lin, Zheng Yuan, Chang Zhou, and Jingren Zhou. Routing to the expert: Efficient reward-guided ensemble of large language models. In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1964–1974, Mexico City, Mexico, June 2024. Association for Computational Linguistics.

[LZHM23] Puxin Liu, Eric Zelikman, Tatsunori Hashimoto, and Christopher D. Manning. Modular prompting: Learning to compose prompts for transferable language models. In *International Conference on Learning Representations (ICLR)*, 2023.

[MGS+23] Aman Mittal, Pranjal Gupta, Ayush Saxena, Rohan Verma, and Mayank Singh. Combining large language models with stacking for enhanced performance. *arXiv preprint arXiv:2305.18873*, 2023.

[MKK24] Costas Mavromatis, Petros Karypis, and George Karypis. Pack of llms: Model fusion at test-time via perplexity optimization. *arXiv preprint arXiv:2404.11531*, 2024.

[NYL+24] Kun-Peng Ning, Shuo Yang, Yu-Yang Liu, Jia-Yu Yao, Zhen-Hui Liu, Yong-Hong Tian, Yibing Song, and Li Yuan. Pico: Peer review in llms based on the consistency optimization. *arXiv preprint arXiv:2402.01830*, 2024.

[OAW+24] Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E Gonzalez, M Waleed Kadous, and Ion Stoica. Routellm: Learning to route llms from preference data. In *The Thirteenth International Conference on Learning Representations*, 2024.

[OBCR+25] Jonathan G. R. O'Keeffe, Joseph N. Bak-Coleman, William B. J. Rammell, Louis J. C. P. Boudreau, Edward H. Lee, Ryan E. Trost, Alexander B. R. Willett, Alexander E. C. Reid, Jessica S. L. O'Keefe, Simon D. G. T. D. Smith, William F. H. Roberts, William D. T. B. Stevens, and Jonathan D. F. E. E. G. Davies. Wisdom of the silicon crowd. *Science Advances*, 11(10):eadi528, 2025.

[PRWZ02] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318. ACL, 2002.

[RFLS20] Ricardo Rei, Ana Farinha, Alon Lavie, and André F. T. Martins Silva. Comet: A neural framework for mt evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, 2020.

[RYCM24] Hossein A Rahmani, Emine Yilmaz, Nick Craswell, and Bhaskar Mitra. Judgeblender: Ensembling judgments for automatic relevance assessment. *arXiv preprint arXiv:2412.13268*, 2024.

[SDP20] Thibault Sellam, Dipanjan Das, and Ankur P Parikh. Bleurt: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 7881–7892, 2020.

[SLSW23] Yutian Shen, Daniel Lee, Sainbayar Sukhbaatar, and Jason Weston. Skillnet: Transferable multi-task reinforcement learning with modular skill networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[SMM+17] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc V. Le, Geoffrey E. Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

[SWJ+24] Weizhou Shen, Jimmy Ba Wu, Zhenghao Jiang, Yao Liang, Heng Du, Xiangning Lin, Zichao Lin, Luke Zettlemoyer, and Luheng Hou. Small llms are also weak tool users. *arXiv preprint arXiv:2403.03695*, 2024.

[SZLC23] Yu Sun, Shuyan Zhou, Yujia Li, and Danqi Chen. Sweetspan: Teaching large language models to blend text with span-level fusion. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2023.

[TDBM20] Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. Synthesizer: Rethinking self-attention in transformer models. In *International Conference on Machine Learning*, pages 10183–10192. PMLR, 2020.

[WLL+25] Zijian Wang, Jiayong Li, Yu Liu, Xuhang Li, Cairong Yan, and Yanting Zhang. Rl–fusion: The large language model fusion method based on reinforcement learning for task enhancing. *Applied Sciences*, 15(4), 2025.

[WSZ+24] Can Wang, Dianbo Sui, Bolin Zhang, Xiaoyu Liu, Jiabao Kang, Zhidong Qiao, and Zhiying Tu. A framework for effective invocation methods of various llm services. *arXiv preprint arXiv:2402.03408*, 2024.

[WWS+22] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

[XCWZ25] Yangyifan Xu, Jianghao Chen, Junhong Wu, and Jiajun Zhang. Hiţ the sweet spot! span-level ensemble for large language models. In Owen Rambow, Leǫ Wanner, Marianna Apidianaki, Hend Al-Khalifa, Barbara Di Eugenio, and Steven Schockaert, editors, *Proceedings of the 31st International Conference on Computational Linguistics*, pages 8314–8325, Abu Dhabi, UAE, January 2025. Association for Computational Linguistics.

[YLZ+23] Han Yang, Mingchen Li, Huixue Zhou, Yongkang Xiao, Qian Fang, and Rui Zhang. One llm is not enough: Harnessing the power of ensemble learning for medical question answering. *medRxiv*, 2023.

[ZKW+20] Tianyi Zhang, Varsha Kishor, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. In *International Conference of Learning Representations (ICLR)*, 2020.

[ZWL+23] Wayne Xin Zhao, Chong Wu, Yujie Liu, Xiaohui Wang, Liheng Wang, Yanyan Chen, and Ji-Rong Wen. Active elicitation for large language models. *arXiv preprint arXiv:2305.15450*, 2023.

[ZXL+23] Lingfeng Zhu, Zhaoxi Xing, Jian-Guang Lou, Ting Wang, Pengfei Xu, Yuan Li, Jianjun Li, Jun Li, Liusong Yin, Junlin Li, Hao Liu, Jing Li, Yu Li, Kai xin Zhang, Jun Wang, and Jun Zhang. A survey of large language model ensembles. *arXiv preprint arXiv:2309.17071*, 2023.

[ZYLH23] Yuheng Zha, Yichi Yang, Ruichen Li, and Zhiting Hu. AlignScore: Evaluating factual consistency with unified alignment function. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the* 61^{st} *Annual Meeting for the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11328–11348, Toronto, Canada, July 2023. Association of Computational Linguistics.