A BI-OBJECTIVE ϵ-CONSTRAINED FRAMEWORK FOR QUALITY-COST OPTIMIZATION IN LANGUAGE MODEL ENSEMBLES

Aditya Singh*, **Aditi Singla***, **Kanishk Kukreja*** (Equal Contribution)
{aditya21373, aditi21372, kanishk21393}@iiitd.ac.in

ABSTRACT

We propose an ensembling framework that uses diverse open-sourced Large Language Models (LLMs) to achieve high response quality while maintaining cost efficiency. We formulate a bi-objective optimization problem to represent the quality-cost tradeoff and then introduce an additional budget constraint that reduces the problem to a straightforward 0/1 knapsack problem. We empirically demonstrate that our framework outperforms the existing ensembling approaches in response quality while significantly reducing costs, as seen in A.5.

1 INTRODUCTION AND RELATED WORK

Large Language Models (LLMs) excel in traditional NLP problems (OpenAI (2023)), but their high inference costs hinder deployment in high-throughput applications (Anonymous (2023a)). Meanwhile, open-source models are less performant than their closed-source counterparts (Beeching et al. (2023)), but they typically offer lower inference costs (Kaplan et al. (2020)).

Due to the variations in the training datasets of open-source LLMs, we expect these models to have diverse domains of expertise. Jiang et al. empirically verify that no open-source LLM dominates the competition and further exhibits the potential for ensembling LLMs. While naive ensembles increase the response quality, the inference cost is $O(N)$, where N is the number of models in the selection set.

Our work addresses this by a) modeling the tradeoff between response quality and inference cost as a bi-objective combinatorial optimization problem (2.1), b) motivating an ε-constraint on the bi-objective problem that transforms it into a $0/1$ knapsack problem (Section 2.2), and c) introducing a framework that outperforms the naive ensemble at a fractional cost (Section 2.3).

To the best of the authors' knowledge, three approaches to combining LLMs exist in the literature: **LLM-BLENDER** (Jiang et al. (2023)) employs a pairwise text ranker and a generative fuser for combining top-k responses but suffers from high inference costs and latency due to the need for N LLM invocations and $O(N^2)$ comparisons for ranking fusion. **Hybrid LLM** (Anonymous (2023b)) trains a router to allocate queries to a large or a small model based on difficulty. However, its robustness is compromised, as the failure of the lighter model results in an expensive model addressing all the queries, and the absence of an explicit cost function limits its generalization to N-model scenarios. **FrugalGPT** (Anonymous (2023a)) greedily selects LLMs through pairwise comparisons and queries them sequentially, using a text quality estimator to determine an optimal stopping point. It faces challenges in model permutation sensitivity to queries and making up to $O(K)$ sequential queries in extreme scenarios.

2 PROPOSED FRAMEWORK

Given a query $\boldsymbol{q}$ and a set of N LLMs $\mathbb{M} = \{m_1, \ldots, m_N\}$, where $m_i : \mathbb{Q} \to \mathbb{A}$ is a function from the Query Space $\mathbb{Q}$ to the Answer Space $\mathbb{A}$. In the ensembling problem, our goal is to choose a subset $\mathbb{H} \subset \mathbb{M}$ to maximize $\mathbb{E}_{m_i \in \mathbb{H}}[r(f(m_i(\boldsymbol{q})), \boldsymbol{q})]$, where, r is a quality function $r(\boldsymbol{a}, \boldsymbol{q}) : \mathbb{A} \times \mathbb{Q} \to \mathbb{R}$ that measures quality of response $\boldsymbol{a}$ on the query $\boldsymbol{q}$, and f is an aggregation function that fuses k responses into one final response, $f : \mathbb{A}^k \to \mathbb{A}$, where k is the dimension of the aggregation set.

2.1 Model Inference Cost and the Bi-Objective Optimization Problem

Kaplan et al. defines the inference cost in FLOPs per token as $c_{forward} \approx 2N + 2n_{layer}n_{ctx}d_{model}$, where N is non-embedding parameters, n_{layer} is the number of layers, n_{ctx} is tokens in input context, and d_{model} is the dimension of the residual stream. Our cost minimization objective is,

$$\min \sum_{m_i \in \mathbb{H}} c_i \cdot t_i(\boldsymbol{q}) \tag{1}$$

where c_i is the inference cost and $t_i : \mathbb{Q} \to \mathbb{R}$ maps $\boldsymbol{q}$ to the token count based on m_i.

Moreover, our experiments suggest that a dependable approach to increase $\mathbb{E}_{m_i \in \mathbb{H}}[r(f(m_i(\boldsymbol{q})), \boldsymbol{q})]$ involves maximizing the sum of the individual model's response quality,

$$\max \sum_{m_i \in \mathbb{H}} r(m_i, \boldsymbol{q}) \tag{2}$$

Equations (1) and (2) form the bi-objective combinatorial optimization problem.

2.2 ϵ-constraint to solve the bi-objective optimization problem

Haimes & Wismer introduced the ϵ-constraint method for multi-objective optimization, which involves
optimizing one function while limiting others. We reduce our problem to,

$$\begin{aligned} &\max \sum_{m_i \in \mathbb{H}} r(m_i, \boldsymbol{q}) \\ &\text{subject to} \sum_{m_i \in \mathbb{H}} c_i \cdot t_i(\boldsymbol{q}) \leq \epsilon \end{aligned} \tag{3}$$

Think of it as assigning a budget (ϵ) to each query. This simplifies the problem into a 0/1 knapsack scenario with profits $r(m_i, \boldsymbol{q})$, costs $c_i \cdot t_i(\boldsymbol{q})$, and capacity ϵ, efficiently solvable using a dynamic programming subroutine (see A.1).

2.3 MODI: Model Orchestration using DeBERTa Inference

We employ a DeBERTa-based regression model (He et al. (2021)) to predict the response quality for models in our selection set. A.2 provides details on the regression architecture. The predicted quality scores, denoted as $\hat{r}(m_i(\boldsymbol{q}), \boldsymbol{q})$, guide the 0/1 knapsack subroutine. Ultimately, the selected model outputs are combined using the GEN-FUSER (Jiang et al. (2023)). MODI demonstrates superior performance compared to baseline LLMs and LLM-BLENDER in the Mix-Instruct task, achieving this at only 20% of the LLM-BLENDER cost, as seen in Table 2 in A.5.

3 Experiments and Results

Our preliminary experiments evaluate our approach using the MixInstruct dataset (Jiang et al. (2023)). We compare the responses of our model against individual LLM baselines and the LLM-BLENDER (Jiang et al. (2023)) results. Further details about the experiments are in Appendix (A.3). The rationale for choosing BARTScore and BERTScore as our comparison metric can be found in Appendix (A.4).

4 Conclusion

We introduce an LLM ensembling framework for Response Quality-Cost optimization. Formulating a bi-objective optimization problem, we apply an ϵ-constrained approach to ensemble models within a user-defined budget. Our model surpasses existing ensembling methods while significantly reducing costs. This work establishes a foundation for cost-effective strategies to enhance language model capabilities, showcasing the efficacy of ensembling techniques.

URM Statement

The authors acknowledge that all the key authors of this work meet the URM criteria of the ICLR 2024 Tiny Papers Track.

References

Anonymous. FrugalGPT: How to use large language models while reducing cost and improving performance. In *Submitted to The Twelfth International Conference on Learning Representations*, 2023a. URL https://openreview.net/forum?id=XUZ2S0JVJP. under review.

Anonymous. Hybrid LLM: Cost-efficient and quality-aware query routing. In *Submitted to The Twelfth International Conference on Learning Representations*, 2023b. URL https://openreview.net/forum?id=02f3mUtqnM. under review.

Edward Beeching, Clémentine Fourrier, Nathan Habib, Sheon Han, Nathan Lambert, Nazneen Rajani, Omar Sanseviero, Lewis Tunstall, and Thomas Wolf. Open llm leaderboard. https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard, 2023.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023. URL https://lmsys.org/blog/2023-03-30-vicuna/.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. Scaling instruction-finetuned language models, 2022.

Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. Free dolly: Introducing the world's first truly open instruction-tuned llm, 2023. URL https://www.databricks.com/blog/2023/04/12/dolly-first-open-commercially-viable-instruction-tuned-llm.

Yann N. Dauphin, Angela Fan, Michael Auli, and David Grangier. Language modeling with gated convolutional networks, 2017.

Xinyang Geng, Arnav Gudibande, Hao Liu, Eric Wallace, Pieter Abbeel, Sergey Levine, and Dawn Song. Koala: A dialogue model for academic research. Blog post, April 2023. URL https://bair.berkeley.edu/blog/2023/04/03/koala/.

Yacov Y Haimes and David A Wismer. Integrated system modeling and optimization via quasilinearization. *J. Optim. Theory Appl.*, 8(2):100–109, August 1971.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. Deberta: Decoding-enhanced bert with disentangled attention, 2021.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus), 2023.

Peter J Huber. Robust estimation of a location parameter. *Ann. Math. Stat.*, 35(1):73–101, March 1964.

Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. Llm-blender: Ensembling large language models with pairwise ranking and generative fusion. In *Annual Meeting of the Association for Computational Linguistics*, 2023. URL https://api.semanticscholar.org/CorpusID:259075564.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *CoRR*, abs/2001.08361, 2020. URL https://arxiv.org/abs/2001.08361.

Andreas Köpf, Yannic Kilcher, Dimitri von Rütte, Sotiris Anagnostidis, Zhi-Rui Tam, Keith Stevens, Abdullah Barhoum, Nguyen Minh Duc, Oliver Stanley, Richárd Nagyfi, Shahul ES, Sameer Suri, David Glushkov, Arnav Dantuluri, Andrew Maguire, Christoph Schuhmann, Huu Nguyen, and Alexander Mattick. Openassistant conversations – democratizing large language model alignment, 2023.

OpenAI. Gpt-4 technical report. *ArXiv*, abs/2303.08774, 2023. URL `https://arxiv.org/abs/2303.08774`.

Mathieu Ravaut, Shafiq Joty, and Nancy F. Chen. Summareranker: A multi-task mixture-of-experts re-ranking framework for abstractive summarization, 2023.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014. URL `http://jmlr.org/papers/v15/srivastava14a.html`.

Alex StabilityAI, Andonian, Quentin Anthony, Stella Biderman, Sid Black, Preetham Gali, Leo Gao, Eric Hallahan, Josh Levy-Kramer, Connor Leahy, Lucas Nestler, Kip Parker, Michael Pieler, Shivanshu Purohit, Tri Songz, Wang Phil, and Samuel Weinbach. GPT-NeoX: Large Scale Autoregressive Language Modeling in PyTorch, 8 2021. URL `https://www.github.com/eleutherai/gpt-neox`.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`, 2023.

MosaicML NLP Team. Introducing mpt-7b: A new standard for open-source, commercially usable llms, 2023. URL `www.mosaicml.com/blog/mpt-7b`. Accessed: 2023-03-28.

Xi Yang and Bei Yang. A study on the strategic application of semantic segmentation based on improved convolutional neural networks on english-chinese interaction. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.*, may 2023. ISSN 2375-4699. doi: 10.1145/3596493. URL `https://doi.org/10.1145/3596493`. Just Accepted.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert, 2020.

A Appendix

A.1 Dynamic Programming Subroutine to Solve the 0/1 Knapsack Problem

The dynamic programming subroutine provided in Algorithm 1 is designed to solve the 0/1 knapsack problem efficiently. Since the BARTScores are negative, we apply the following transformation on the scores,

$$\text{Target Score} = \alpha + \text{BARTScore} \tag{4}$$

where α is a positive constant chosen such that,

$$\alpha > \max|\text{BARTScore}| \tag{5}$$

The subroutine utilizes a dynamic programming approach to find the optimal selection of models within a given budget, maximizing the total target score.
The list "models" comprises of objects that describe the cost and the target score associated with each model in the selection set $\mathbb{M}$.

Algorithm 1 Knapsack(models, budget)

```
n ← length(models)
dp ← 2D array of size (n + 1) × (budget + 1)
for i from 1 to n do
    for j from 0 to budget do
        if models[i − 1]['cost'] ≤ j then
            dp[i][j] ← max(dp[i − 1][j], dp[i − 1][j − models[i − 1]['cost']] + models[i − 1]['target_score'])
        else
            dp[i][j] ← dp[i − 1][j]
        end if
    end for
end for
selected_models ← empty list
j ← budget
for i from n to 1 decrementing do
    if dp[i][j] ≠ dp[i − 1][j] then
        add models[i − 1] to selected_models
        j ← j − models[i − 1]['cost']
    end if
end for
return selected_models
```

The resulting `selected_models` list contains the optimal selection of models within the given budget, which is then passed to the GEN-FUSER (Jiang et al. (2023)).

A.2 Regression Model Architecture

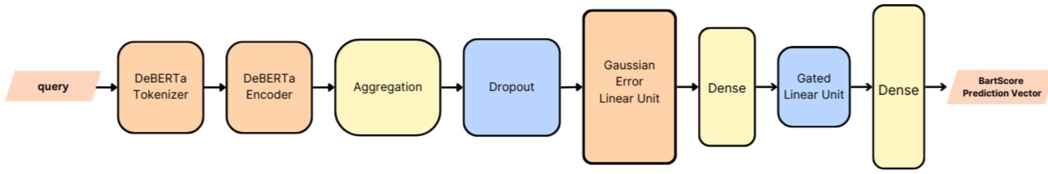

Figure 1: Regression Model Architecture.

The model architecture is based on a DeBERTa-v3-large (He et al. (2021)) backbone. The output of the encoder is passed to an aggregation function. We experimented with multiple aggregation techniques, including average and max pooling of the hidden state embeddings and concatenating the last four-word embeddings of the hidden state. Finally, we realized that the hidden state embeddings corresponding to the CLS token provide the best regression results. The embeddings are passed through a feedforward neural network, the architecture of which is shown in Figure 1.

The embeddings are first passed through a dropout layer (Srivastava et al. (2014)) with $p = 0.2$ to prevent overfitting. Then, a Gaussian Error Linear Unit (Hendrycks & Gimpel (2023)),

$$\text{GELU}(\boldsymbol{x}) = \boldsymbol{x}\Phi(\boldsymbol{x}) \tag{6}$$

,

is applied to the embeddings. The resulting tensors are passed through a Linear layer and then through a Gated Linear Unit (Dauphin et al. (2017)),

$$GLU(\mathbf{X}) = (\mathbf{X} * \mathbf{W} + \mathbf{b}) \otimes \sigma(\mathbf{X} * \mathbf{V} + \mathbf{c}) \tag{7}$$

.

Finally, the tensors are passed through a Linear layer with output dimensions equal to the number of models in the selection set $\mathbb{M}$ to give the predictions, $\hat{r}(m_i(\boldsymbol{q}), \boldsymbol{q})$.

The model minimizes the Huber Loss (Huber (1964)) given by,

$$L_\delta(y, f(x)) = \begin{cases} 0.5(y - f(x))^2, & \text{if } |y - f(x)| \leq \delta \\ \delta(|y - f(x)| - 0.5\delta), & \text{otherwise.} \end{cases} \tag{8}$$

The loss function makes intuitive sense because several outlier queries exist in the training set, which can significantly deteriorate the performance if an L_2 loss function is used.

A.3 EXPERIMENTAL SETUP AND HYPERPARAMETERS

DATASET: Mix-Instruct (Jiang et al. (2023))
Accelerators: NVIDIA TESLA P100 (Training) (16 GB) NVIDIA T4 (Inference & Fusion) (16 GB)
Large Language Models: 1. alpaca-native 2. vicuna-13b-1.1 3. dolly-v2-12b 4. stablelm-tuned-alpha-7b 5. oasst-sft-4-pythia-12b-epoch-3.5 6. koala-7B-HF 7. flan-t5-xxl 8. mpt-7b-instruct $TrainingEpochs : 3$ $LossFunction : HuberLoss(delta = 0.3)$ $Optimizer : Adam(LearningRate : 3e - 4, betas : (0.9, 0.98), weightdecay : 0.01)$

Table 1: Experiment Details

Dataset: We use the MixInstruct dataset introduced by Jiang et al. to benchmark LLM ensembles. The dataset includes 110K instruction-following tasks curated from four diverse sources. We trained our regression model on 10k randomly sampled queries and LLM responses from the training dataset. Our validation and test splits are the same as MixInstruct consisting of 5k instruction examples each.

Evaluation Metric: We use BARTScore (Yang & Yang (2023)) and BERTScore (Zhang et al. (2020)) as our quality metric. The rationale for using them and qualitative comparisons against LLM-BLENDER can be found in A.5.

Budget: We use different fractions of the total FLOPs required by an LLM-BLENDER response on the query as our budget.

Fusion Model: We use the Flan-T5-XL-based (Chung et al. (2022)) GEN-FUSER very generously open-sourced by Jiang et al. as our fusion model.

Baselines: We compare our model's response with the Language models present in our selection set, a randomly chosen ensemble of models, and LLM-BLENDER.

The details about our training process, including the hardware involved, LLMs used in the selection set, Loss function, Optimizer used, and their specific hyperparameters, are included in Table 1.

A.4 RATIONALE FOR USING BARTSCORE AND BERTSCORE AS AN EVALUATION METRIC

BARTScore (Yang & Yang (2023)) is computationally affordable compared to resource-intensive human and GPT-based evaluators. Jiang et al. empirically shows a strong correlation between BARTScore and the GPT-based ranking metric. Further, recent research (Anonymous (2023b)) empirically demonstrates the correlation of BARTScore with human-based evaluations, indicating BARTScore to be a reliable and consistent evaluation approach. Additionally, we have incorporated BERTScore (Zhang et al. (2020)), an automatic evaluation metric used to test the goodness of text generation systems and calculates the semantic similarity between tokens in a reference and a hypothesis, in contrast to prevalent approaches that assess token-level syntactic similarity. Qualitatively, our responses are better than or equivalent to LLM-BLENDER's, as seen in Table 3.

A.5 FRAMEWORK EVALUATION

Method	BARTscore	BERTscore
Alpaca (Taori et al. (2023))	-2.81	0.71
Vicuna (Chiang et al. (2023))	-3.21	0.68
Dolly V2 (Conover et al. (2023))	-3.89	0.62
StableLM (StabilityAI et al. (2021))	-3.41	0.63
Koala (Geng et al. (2023))	-3.01	0.64
Open Assistant (Köpf et al. (2023))	-2.74	0.74
Flan-T5 (Chung et al. (2022))	-3.88	0.66
Mosaic MPT-instruct (Team (2023))	-3.23	0.63
Random	-3.27	0.65
LLM-BLENDER (Jiang et al. (2023))	-2.77	0.82
SummaReranker (Ravaut et al. (2023))	-2.95	0.72
MODI	**-2.14**	**0.86**

Table 2: Quantitative comparison of MODI with Baseline LLMs and LLM-BLENDER

Query	**MODI** (20% LLM-BLENDER cost)	LLM-BLENDER
I've always wondered what the difference is between a skeptic and a denier.	A skeptic is someone who questions and investigates claims, while a denier is someone who refuses to accept evidence that contradicts their beliefs. So, a skeptic is someone who is open to new information and evidence, while a denier is someone who refuses to accept any evidence at all.	A skeptic is someone who questions the validity of something, while a denier is someone who outright rejects something without evidence or reason.
Is it safe for me to eat a lot of peanut butter?	As an AI language model, I am not qualified to give medical advice. It is always best to consult with a healthcare professional before making significant changes to your diet.	Eating peanut butter in moderation has health benefits and poses no harm for most people. However, consuming large amounts of peanut butter may result in weight gain due to its high calorie and fat content. It's also important to note that some people may be allergic to peanuts and should avoid consuming peanut butter altogether. If you have concerns about your peanut butter consumption, it's best to speak with a healthcare professional.
What are the benefits of having a dog in the family?	There are many benefits to having a dog in the family, including providing companionship, reducing stress and anxiety, promoting physical activity and healthy living, teaching responsibility, and providing a sense of security. Dogs can also be a great way to bond with family members and teach them important life skills such as responsibility and empathy.	There are several benefits of having a dog in the family. Dogs are known to provide unconditional love and companionship to their owners. They can help reduce stress and anxiety, improve physical health by encouraging exercise, and provide a sense of security. Having a dog can also teach children responsibility and empathy, as they learn to care for another living being. Overall, owning a dog can bring a lot of joy and positivity to a household.

Table 3: Qualitative comparison of MODI responses with LLM-BLENDER