MESS+: Energy-Optimal Inferencing in Language Model Zoos with Service Level Guarantees

Ryan Zhang
Horace Greeley High School
ryzhangofficial@gmail.com

Herbert Woisetschläger
Technical University of Munich
h.woisetschlaeger@tum.de

Shiqiang Wang
IBM Research
wangshiq@us.ibm.com

Hans-Arno Jacobsen
University of Toronto
jacobsen@eecg.toronto.edu

Abstract

Open-weight large language model (LLM) zoos allow users to quickly integrate state-of-the-art models into systems. Despite increasing availability, selecting the most appropriate model for a given task still largely relies on public benchmark leaderboards and educated guesses. This can be unsatisfactory for both inference service providers and end users, where the providers usually prioritize cost efficiency, while the end users usually prioritize model output quality for their inference requests. In commercial settings, these two priorities are often brought together in Service Level Agreements (SLA). We present MESS+, an online stochastic optimization algorithm for energy-optimal model selection from a model zoo, which works on a per-inference-request basis. For a given SLA that requires high accuracy, we are up to $2.5\times$ more energy efficient with MESS+ than with randomly selecting an LLM from the zoo while maintaining SLA quality constraints.

1 Introduction

As the number of open-weight large language models (LLMs), such as Llama [3], Mistral/Mixtral [8], and Granite [6], is increasing rapidly, deep learning infrastructure providers and end users are confronted with an abundance of models (model zoo) for their language modeling tasks. This leaves many users questioning what model is best to choose and whether highly regarded benchmark results apply to their specific problem [13]. Currently, the best way to approach model selection is educated guessing. Since working with LLMs can be expensive [17], minimizing costs is an equally high priority for end users and inference endpoint operators. This leaves us with a tri-fold problem:

End-users primarily care about a correct model output. When inquiring about text information, e.g., by asking questions or requesting language translation, end users are mostly interested in obtaining factually correct and sufficiently clear language output [20]. Additionally, many users are unfamiliar with the technical details of LLMs, making it challenging for them to select the right model for the job, i.e., their primary references are domain-specific benchmark rankings [4, 5]. However, there is no intuitive method to compare the complexity of individual requests with benchmark tasks. Thus, we require an automatic method to select the most appropriate LLM for any given request.

Inference endpoint providers prioritize low operating costs. Operating infrastructure that can run state-of-the-art LLMs can be costly. Microsoft has announced it will acquire a stake in the Three Mile Island nuclear power plant in the United States to satisfy the energy demand of its planned data center in Pennsylvania, which has two reasons: consistent energy delivery and low energy cost

Accepted at the 2024 Workshop on Adaptive Foundation Models in conjunction with the 38th Conference on Neural Information Processing Systems (NeurIPS 2024).

[16]. In times of globally rising energy costs, this underpins the necessity of energy-optimal service operations. Currently, inference service providers only allow their users to query specific models on serverless endpoints or to deploy individual models on dedicated hardware. To further optimize operating costs and improve user experience, we require a method that can choose the best model for any given request while minimizing energy consumption.

Enterprise use-cases require a consistently high-quality model inference output while keeping costs in check. Enterprise users unite the requirement for high-quality model outputs and price sensitivity. Thus, commercial players typically rely on service-level agreements (SLAs) when sourcing services for their own products. This creates a legal basis for holding the model operator responsible for delivering high quality and performance. Such SLAs typically come with various levels where, in the case of model outputs, the primary quality metric is accuracy. Thus, we require a method for formalizing and quantifying the SLA requirements.

In summary, we ask the question:

> *Can we select appropriate models from the model zoo to ensure energy efficiency while satisfying SLAs?*

In this paper, we address this question by using a stochastic optimization [14] framework to develop an optimal control algorithm, which enables end users to query an inference service and automatically select the most appropriate **M**odel with **E**nergy-optimal **S**ervice-level Guarantee**S** (MESS+).

Our work is related to two major research directions: dynamic inference and inference request scheduling. A broad overview is provided in [24].

Dynamic inference. Typically, dynamic inference approaches involve early exit strategies within a single model and require changes to the original model architecture. These changes are usually additional layers that decide whether a request continues to subsequent layers or is evicted [12, 22]. While these approaches can save inference costs, they require additional training of the decision layers. Further, increasing the capabilities of dynamic inference models is difficult as it requires changing the model architecture and re-training. MESS+ removes the need for altering the architecture of readily available pre-trained models and automatically selects the most appropriate model with regard to a given SLA for each request.

Inference request scheduling. To this end, scheduling work primarily focuses on reducing latency during an inference call. Here, the main idea is to group [10], arrange [7], or early evict [21] inference requests for faster processing. These approaches focus exclusively on scenarios with a single fixed model for all incoming requests and a priority for latency. In contrast, MESS+ is a decision-making method to route requests for energy-optimal processing while maintaining a minimum accuracy over time. As such, our approach contributes to establishing minimum quality guarantees but does not consider latency directly. However, there is a relationship between latency and energy efficiency since smaller models are typically faster to execute an inference call [17]. Since MESS+ routes inference requests to different models, it can build on top of existing scheduling techniques.

Taken together, we enable dynamic inference across readily available pre-trained LLMs with service level guarantees, while offering compatibility with existing inference load scheduling techniques. MESS+ can reduce the energy consumption of a given task by up to $4.6\times$ compared to a fixed selection of a single model from a model zoo.

2 MESS+: Model Selection With Energy-Optimal Service Level Guarantees

The overall goal of MESS+ is to find the most suitable LLM for each inference request $t \in T$ to minimize the energy consumption $E_m(t)$ under model performance constraints defined by an SLA over time to ensure contractual compliance.

2.1 Overall Problem Formulation

We first formulate the optimization problem and introduce typical constraints that come with SLAs.

Inference Cost (Objective). The energy costs for querying a model can vary significantly over time and depend on the chosen model, i.e., our computational costs are volatile in model zoos. For instance, a model zoo can include an LLM with 1B parameters and another with 13B parameters. The smaller LLM requires significantly fewer resources than the larger model.

Service-Level Agreement (Constraints). In SLAs, we typically find a minimum service quality requirement. We consider a minimum average accuracy over requests, denoted by α.[1] We denote each model's accuracy for processing the inference request t as $A_m(t)$, and we choose exactly one model for any t to identify the most appropriate model.

Overall control problem. Taken together, the objective and constraints can be formalized into the following problem of minimizing the average energy consumption per request under performance constraints defined in an SLA:

$$\min_{\{y_m(t):\forall t,m\}} \quad \frac{1}{T}\sum_{t=1}^{T}\sum_{m=1}^{M} y_m(t)E_m(t), \tag{1a}$$

$$\text{s.t.} \quad \frac{1}{T}\sum_{t=1}^{T}\sum_{m=1}^{M} y_m(t)A_m(t) \geq \alpha, \tag{1b}$$

$$\sum_{m=1}^{M} y_m(t) = 1, \forall t \in \{1,\dots,T\}, \tag{1c}$$

$$y_m(t) \in \{0,1\}, \forall t, m, \tag{1d}$$

where $y_m(t) = 1$ if model m is chosen and $y_m(t) = 0$ otherwise.

Challenges. Our optimization problem involves an inherent trade-off between model accuracy and energy consumption, since larger LLMs, and thus more capable ones, typically yield higher accuracy while consuming more energy at the same time. As such, we see a *correlation over time between* the objective and constraints. Further, optimizing the energy costs involves a time average that is hard to estimate a priori as the properties of future requests are generally *unknown* and *heterogeneous*. Similarly, $A_m(t)$ is only available *after* querying the model and only if there is a *feedback signal* on whether the response is satisfactory or not.

2.2 Translation into an Online Decision Making Problem

To address the aforementioned challenges, we introduce an online decision-making process. Here, the quantities of $E_m(t)$ and $A_m(t)$ are captured in every request, i.e., for every arriving t, without knowledge of future statistics.

Methodology. We base our approach on the Lyapunov drift-plus-penalty framework [14]. Since SLAs are often volume-based, service quality guarantees are typically given for a maximum number of requests or transactions. Thus, we deviate from [14] and assume a finite T instead of an infinite T.

Virtual Queues. We use a virtual queue with length Q to capture SLA violations, where

$$Q(t+1) \leftarrow \max\left\{0, Q(t) + \alpha - \sum_{m=1}^{M} y_m(t)A_m(t)\right\}, \tag{2}$$

where for $t = 1$, we initialize $Q(1)$ to either zero or a small positive value. Intuitively, this captures the cumulative violations. Hence, we aim to collectively minimize our objective (1a) and the queue length. We note that the average accuracy over requests needs to be greater than or equal to α in the constraint, while the direction of inequalities in the constraint is opposite in [14], thus our queue update equation in (2) is slightly different from that in [14].

Decision problem for each request. For each inference request t, we aim to minimize energy consumption while complying with SLA requirements. We formulate this trade-off by

$$\min_{\{y_m(t):\forall m\}} \quad V \cdot \sum_{m=1}^{M} y_m(t)E_m(t) + Q(t)\left(\alpha - \sum_{m=1}^{M} y_m(t)\hat{A}_m(t)\right), \tag{3a}$$

$$\text{s.t.} \quad \text{Constraints (1c), (1d)}, \tag{3b}$$

where $\hat{A}_m(t)$ is a predictor of the *estimated output* of $A_m(t)$ because it is impossible to know the exact accuracy before the LLM processes the request.[2] We will describe the computation of $\hat{A}_m(t)$ in Section 2.3. We consider the energy consumption per request to be known, i.e., we measured the cost for an inference call before adding a model to the model zoo, which can depend on some

[1] In practice, α should be chosen with a certain safety margin from the SLA requirement such that we do not violate the SLA even if the average accuracy is slightly below α.

[2] Note that, in this work, we assume that the accuracy can be estimated immediately after the LLM generates its output. Therefore, in (2), we use $A_m(t)$ to update the virtual queue length, but we use $\hat{A}_m(t), \forall m$, to solve the per-request optimization problem (3). The more realistic scenario where the accuracy is not known even after obtaining the LLM output is left for future work.

characteristics of the input request such as its length. As SLAs come in various configurations, we introduce the control parameter $V > 0$ that caters to different quality requirements. A high value of V increases the priority of energy efficiency and lowers the need for accuracy.

Solution to (3). Due to Constraints (1c), (1d), the problem in (3) can be easily solved using a linear search by setting $y_m(t) = 1$ and $y_{m'}(t) = 0$ (for all $m' \neq m$), for each $m \in \{1, 2, \ldots, M\}$, and comparing the values of the objective (3a). This procedure has a linear complexity of $\mathcal{O}(M)$. Using similar proof techniques as those in [14], we can show constraint satisfaction guarantee, i.e., SLA guarantee, and optimality as $T \to \infty$. The detailed proof is left for a future version of this work.

2.3 Predicting the Accuracy for Each Request

We now describe how to obtain the predicted accuracy $\hat{A}_m(t), \forall m, t$, which is needed to solve (3). To facilitate the description, we write out $\hat{A}_m(t) = \hat{A}(\mathbf{x}_{m,t}, \mathbf{a}_t)$, where $\mathbf{x}_{m,t}$ is the parameter vector of the (small) accuracy estimation model for the m-th LLM used for request t, and $\mathbf{a}_t$ is the t-th input request. We omit the subscript t in $\mathbf{x}_{m,t}$ in the following when it is unnecessary to specify the parameter used for a specific request t. We learn $\mathbf{x}_m$ through a probabilistic exploration procedure. To *explore* the model zoo, we query multiple models with the same input request to obtain their actual accuracies $A_m(t)$, allowing us to learn $\mathbf{x}_m$ from $A_m(t)$ over time. More specifically, we sample from a distribution $\mathcal{X}_t \sim \text{Bernoulli}(p_t)$, where the exploration probability $p_t \leftarrow \min(1, \frac{c}{\sqrt[3]{t}})$ and parameter $c > 0$ controls the exploration likelihood. The larger c, the more likely it is to do an exploration step. We decay the probability p_t over time as the estimation $\hat{A}_m(t)$ improves with each exploration step.

Especially for the first set of arriving requests, when we do not know how to choose the optimal LLM for request t, we must explore each m in the model zoo for the best-performing model for t. While doing so, we capture the actual model accuracy $A_m(t)$ of each m, which we use to learn $\mathbf{x}_m$. We define the mean square error (MSE) objective of accuracy prediction for an estimation over all the possible incoming requests:

$$L(\mathbf{x}_m) = \mathbb{E}_{\mathbf{a}_t}\left(\hat{A}(\mathbf{x}_m, \mathbf{a}_t) - A_m(t)\right)^2. \tag{4}$$

If we learn $\mathbf{x}_m$ using stochastic gradient descent (SGD), assuming that the distribution of the input request $\mathbf{a}_t$ is IID across t, it is easy to prove that the convergence upper bound is

$$\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|\nabla L(\mathbf{x}_{m,t_k})\|^2 \leq \mathcal{O}\big(\tfrac{1}{\sqrt{K}}\big), \tag{5}$$

where K is the number of exploration steps and t_k is the request index corresponding to the k-th exploration step.

Algorithm 1: Selecting the **M**odel with **E**nergy-optimal **S**ervice level Guarantee**S** (MESS+)

Input: T; V; α; c; $\{E_m(t) : \forall m, t\}$; learning rate $\eta > 0$
Output: Outputs of models chosen for all t

Initialize $Q(1) \leftarrow 0$; predictor parameters $\mathbf{x}_m$ to a common random vector for all m; $k \leftarrow 1$;
for $t \leftarrow 1$ **to** T **do**
  Compute $p_t \leftarrow \min\left(1, \frac{c}{\sqrt[3]{t}}\right)$;
  Sample $\mathcal{X}_t \sim \text{Bernoulli}(p_t)$;
  if $\mathcal{X}_t = 1$ **then**
    `// Exploration`
    foreach $m \in \{1, 2, ..., M\}$ **do**
      Obtain true accuracy $A_m(t)$;
      $\mathbf{x}_{m,t+1} \leftarrow \mathbf{x}_{m,t} - \eta\nabla_{\mathbf{x}}\left(\hat{A}(\mathbf{x}_{m,t}, \mathbf{a}_t) - A_m(t)\right)^2$;
    $m^* \leftarrow \arg\max_m A_m(t)$;
  else
    `// Solve (3)`
    $m^* \leftarrow \arg\min_m V \cdot E_m(t) + Q(t) \cdot (\alpha - \hat{A}_m(t))$;
    $\mathbf{x}_{m,t+1} \leftarrow \mathbf{x}_{m,t}$;
  Get output from model m^* and its accuracy $A_{m^*}(t)$;
  `// Virtual queue update`
  $Q(t+1) \leftarrow \max\{0, Q(t) + \alpha - A_{m^*}(t)\}$;

When exploring, we always use the output from the largest model as the final model output as we have already spent the energy to query the largest model, i.e., we *do not* use the solution from (3) in this case. Since we query all the models in an exploration step, it comes at additional energy costs. We can upper bound the *average additional* energy consumption per exploration step over time by

$$\frac{1}{T}\sum_{k=1}^{K} E = \frac{KE}{T}, \tag{6}$$

where E is the maximum energy consumption for querying the entire model zoo.

By choosing p_t as described above, we have $\mathbb{E}[K] = \Theta\big(T^{\frac{2}{3}}\big)$. Then, the SGD convergence bound in (5) becomes $\mathcal{O}\left(1/\sqrt[3]{T}\right)$ and the upper bound of average additional energy consumption due to

exploration in (6) becomes $\mathcal{O}\left(E/\sqrt[3]{T}\right)$. Both of them approach zero as $T \to \infty$. We leave the formal statement and full proof to a future version of this work.

Full algorithm. The full procedure is described in Algorithm 1.

3 Experiments

We demonstrate the effectiveness of MESS+ with state-of-the-art language modeling tasks, including WMT14 [2] for language translation and CNNDailyMail [18] for text summarization. We construct our model zoo from the TinyLlama 1.1B [23] and Llama-2 13B [19] models, each representing a different model class based on its number of model parameters. To measure the performance of MESS+ in choosing the most appropriate model in the model zoo, we use the BLEU-1 [15] metric to evaluate the language translation task and the ROUGE-1 score [11] for the summarization task. We establish one SLA for each task, where the requirement is to reach an average BLEU score of $\alpha = 0.52$ and ROUGE score of $\alpha = 0.315$ for the translation and summarization tasks, respectively. We provide additional experimental details in Appendix A.

Since V and c have to be chosen manually, we run a set of ablation studies to explore the sensitivity of both parameters. Varying the importance of energy efficiency V between $[0.01, 100]$ for both tasks while complying with their respective SLA shows that the maximum V we can choose to reliably achieve α is $V = 0.1$. For $V > 0.1$, we violate the SLA. Interestingly, when looking at values of $[1, 10]$ for c, which controls the exploration likelihood, we observe that with $c = 3$, $\hat{A}(t)$ yields the lowest loss over time and thus the best performance. Hence, we choose $V = 0.1$ and $c = 3$ for our experiments. Further details on selecting V and c are in Appendices B.1 and B.2, respectively.

We look at four different scenarios: (I) choosing the smallest model only (TinyLlama-1.1B), (II) choosing the largest model only (Llama-2 13B), (III) choosing a model randomly while satisfying the constraints, and (IV) applying MESS+. The results are shown in Table 1. In the case of (I), we do not satisfy the SLA in either task, while for (II), we observe the highest accuracy and comply with the SLAs, but the energy consumption is also the highest. Case (III) also complies with the SLAs and reduces the energy consumption by $1.9\times$ and $1.8\times$ for the translation and summarization tasks, respectively, when compared to (II). However, (IV) provides the same average accuracy over time as (III) while reducing the energy consumption by $3.5\times$ and $4.6\times$ for the translation and summarization tasks, respectively, when compared to (II). Thus, we see that MESS+ strictly ensures SLA compliance and reliably reduces energy consumption compared to randomly choosing an available model that satisfies the SLA requirements. Since energy costs are directly proportional to energy usage, the energy savings brought by MESS+ are directly related to cost reduction. With this, MESS+ can help reduce the operating costs for model serving.

Table 1: Comparison of baseline model selection strategies and MESS+ with $V = 0.1$ and $c = 3$. Our approach satisfies the SLA with requirement α while consuming the least energy among all compliant strategies.

Model	**WMT14** ($\alpha = 0.52$)			**CNNDailyMail** ($\alpha = 0.315$)		
	Accuracy (BLEU)	**Energy** (Joules)	**Meets** α	**Accuracy** (ROUGE1)	**Energy** (Joules)	**Meets** α
TinyLlama	49.1 $\pm$ 0.6	44.639 $\pm$ 0.6	No	30.9 $\pm$ 0.3	142.080 $\pm$ 1.4	No
Llama-2 13B	55.1 $\pm$ 0.4	527.870 $\pm$ 5.1	Yes	32.2 $\pm$ 0.3	750.285 $\pm$ 7.5	Yes
Random with constraint	52.0 $\pm$ 0.0	280.426 $\pm$ 3.0	Yes	31.5 $\pm$ 0.0	416.466 $\pm$ 4.3	Yes
MESS+ ($V = 0.1, c = 3$)	52.2 $\pm$ 0.2	**149.399** $\pm$ 1.3	Yes	31.5 $\pm$ 0.1	**163.836** $\pm$ 1.5	Yes

4 Conclusions

We present MESS+, a novel method for automatic model selection in model zoos for language inference services on a per-request basis. The approach is particularly useful for inference providers that cater to users with quality-sensitive workloads as it enables service level guarantees. The experimental evaluation of MESS+ demonstrates the effectiveness in routing requests to the most appropriate model while achieving a minimum required accuracy and reducing energy consumption at the same time. This will lead to overall lower operating costs for inference service providers.

In future work, we will simplify the approximation mechanism that yields $\hat{A}_m(t)$ by removing the need for multiple linear classifiers as this would otherwise establish a new bottleneck with growing model zoos.

Acknowledgements

This work is partially funded by the Bavarian Ministry of Economic Affairs, Regional Development and Energy (Grant: DIK0446/01). We would like to thank the PANDORA project (https://pandora-heu.eu/) for our fruitful discussions.

References

[1] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5: 135–146, 2017. ISSN 2307-387X.

[2] Ondřej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, et al. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the ninth workshop on statistical machine translation*, pages 12–58, 2014.

[3] Abhimanyu Dubey, Abhinav Jauhri, et al. The Llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

[4] Clémentine Fourrier, Nathan Habib, Alina Lozovskaya, Konrad Szafer, and Thomas Wolf. Open llm leaderboard v2. https://huggingface.co/spaces/open-llm-leaderboard/open_llm_leaderboard, 2024.

[5] Leo Gao, Jonathan Tow, et al. A framework for few-shot language model evaluation, September 2021. URL https://doi.org/10.5281/zenodo.5371628.

[6] IBM Granite Team. Granite 3.0 language models, Oct. 2024. URL https://github.com/ibm-granite/granite-3.0-language-models/blob/main/paper.pdf.

[7] Connor Holmes, Masahiro Tanaka, Michael Wyatt, Ammar Ahmad Awan, Jeff Rasley, Samyam Rajbhandari, Reza Yazdani Aminabadi, Heyang Qin, Arash Bakhtiari, Lev Kurilenko, and Yuxiong He. Deepspeed-fastgen: High-throughput text generation for llms via mii and deepspeed-inference, 2024. URL https://arxiv.org/abs/2401.08671.

[8] Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mixtral of experts, 2024. URL https://arxiv.org/abs/2401.04088.

[9] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431. Association for Computational Linguistics, April 2017.

[10] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, SOSP '23, page 611–626, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400702297. doi: 10.1145/3600006.3613165. URL https://doi.org/10.1145/3600006.3613165.

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

[12] Weijie Liu, Peng Zhou, Zhiruo Wang, Zhe Zhao, Haotang Deng, and Qi Ju. FastBERT: a self-distilling BERT with adaptive inference time. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6035–6044, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.537. URL https://aclanthology.org/2020.acl-main.537.

[13] Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. A comprehensive overview of large language models, 2023. URL https://arxiv.org/abs/2307.06435.

[14] Michael Neely. *Stochastic network optimization with application to communication and queueing systems*. Springer Nature, 2022.

[15] 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, 2002.

[16] Reuters. Microsoft deal propels three mile island restart, with key permits still needed. https://www.reuters.com/markets/deals/constellation-inks-power-supply-deal-with-microsoft-2024-09-20/, 2024. [Accessed 24-09-2024].

[17] Siddharth Samsi, Dan Zhao, Joseph McDonald, Baolin Li, Adam Michaleas, Michael Jones, William Bergeron, Jeremy Kepner, Devesh Tiwari, and Vijay Gadepally. From words to watts: Benchmarking the energy costs of large language model inference. In *2023 IEEE High Performance Extreme Computing Conference (HPEC)*, pages 1–9, 2023. doi: 10.1109/HPEC58863.2023.10363447.

[18] Abigail See, Peter J. Liu, and Christopher D. Manning. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1099. URL https://www.aclweb.org/anthology/P17-1099.

[19] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[20] Alexander Wettig, Aatmik Gupta, Saumya Malik, and Danqi Chen. QuRating: Selecting high-quality data for training language models. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 52915–52971. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/wettig24a.html.

[21] Bingyang Wu, Yinmin Zhong, Zili Zhang, Shengyu Liu, Fangyue Liu, Yuanhang Sun, Gang Huang, Xuanzhe Liu, and Xin Jin. Fast distributed inference serving for large language models, 2023. URL https://arxiv.org/abs/2305.05920.

[22] Ji Xin, Raphael Tang, Jaejun Lee, Yaoliang Yu, and Jimmy Lin. DeeBERT: Dynamic early exiting for accelerating BERT inference. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2246–2251, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.204. URL https://aclanthology.org/2020.acl-main.204.

[23] Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. Tinyllama: An open-source small language model. *arXiv preprint arXiv:2401.02385*, 2024.

[24] Zixuan Zhou, Xuefei Ning, Ke Hong, Tianyu Fu, Jiaming Xu, Shiyao Li, Yuming Lou, Luning Wang, Zhihang Yuan, Xiuhong Li, Shengen Yan, Guohao Dai, Xiao-Ping Zhang, Yuhan Dong, and Yu Wang. A survey on efficient inference for large language models, 2024. URL https://arxiv.org/abs/2404.14294.

Appendix

A Additional Experimental Details

Benchmark datasets. We use the WMT14 dataset [2] to evaluate how well our approach works with language translation and the CNNDailyMail [18] dataset for sequence-to-sequence modeling, specifically highlighting summarization from news articles.

Metrics. Since the users sending inference requests are primarily interested in retrieving high-quality results, we use the BLEU-1 [15] metric to evaluate the language translation task and the ROUGE-1 score [11] for the sequence-to-sequence task. To account for the service provider's cost-saving priority, we measure the energy consumption in total joules consumed.

Inference models. We populate M with TinyLlama 1.1B [23] and Llama-2 13B [19]. These models have shown strong performance across language modeling tasks and represent two different model classes: small and large.

MESS+ accuracy predictor. In our experiment, we use one linear model per inference model built with the `fasttext` library to learn $\mathbf{x}_m$ [1, 9]. We query each linear model for every request t to estimate the corresponding inference model performance.

Application scenarios. To demonstrate the sensitivity of control parameter V in MESS+, we explore different service levels as they would be found in commercial services. For the WMT14 translation task, users can choose from three different plans: *silver*, *gold*, and *platinum*, where the target accuracy is 0.49, 0.5, and 0.51, respectively. To discuss the energy efficiency implications, we aim to achieve $\alpha = 0.52$ for WMT14 and $\alpha = 0.315$ for CNNDailyMail. Both scores are considered high [11, 15].

B Additional Experimental Results

In this section, we discuss the impact of our control parameters V (the priority of energy efficiency) and c (the likelihood for exploration steps).

B.1 Varying the Priority for Energy Efficiency

The parameter V serves as a tuning knob that adjusts the trade-off between prioritizing accuracy and minimizing energy usage. By systematically adjusting V, we observe how the system behavior changes in response to the three service level plans. The results are shown in Figure 1.

As mentioned above, we introduce three different SLAs that each require a different α. Between the three α values, we vary the accuracy by exactly 1% to understand how much more energy we have to use in order to serve the next higher SLA.

For the *silver* level, where the minimum accuracy requirement is low, the system is less sensitive to variations of V. As we increase V from 0.01 to 100, the average BLEU score remains consistently above the silver threshold. In such a scenario, it is beneficial to set V to a high value as this prioritizes energy efficiency, which will route more requests to the smaller and faster model.

For the *gold* level, the sensitivity to V becomes more notable. if V is set too high (e.g. $V > 5$, the system excessively prioritizes energy savings, leading to a noticeable drop in the average BLEU score below the acceptable service requirement.

At the *platinum* level, the sensitivity of V increases further compared to gold since the quality requirements have increased as well. However, the viable configurations must now emphasize model accuracy, which leads to $V < 1$. Here, a 1% performance gain costs $1.9\times$ more energy than the gold plan.

Analyzing V reveals that its optimal value is inversely related to the minimum service requirement α. As α increases, the range of acceptable V narrows, and the system demands a greater emphasis on accuracy rather than cost reduction. Service providers can leverage this knowledge to adjust V dynamically based on operational priorities, such as reducing energy costs during non-peak hours and maximizing accuracy during critical periods.

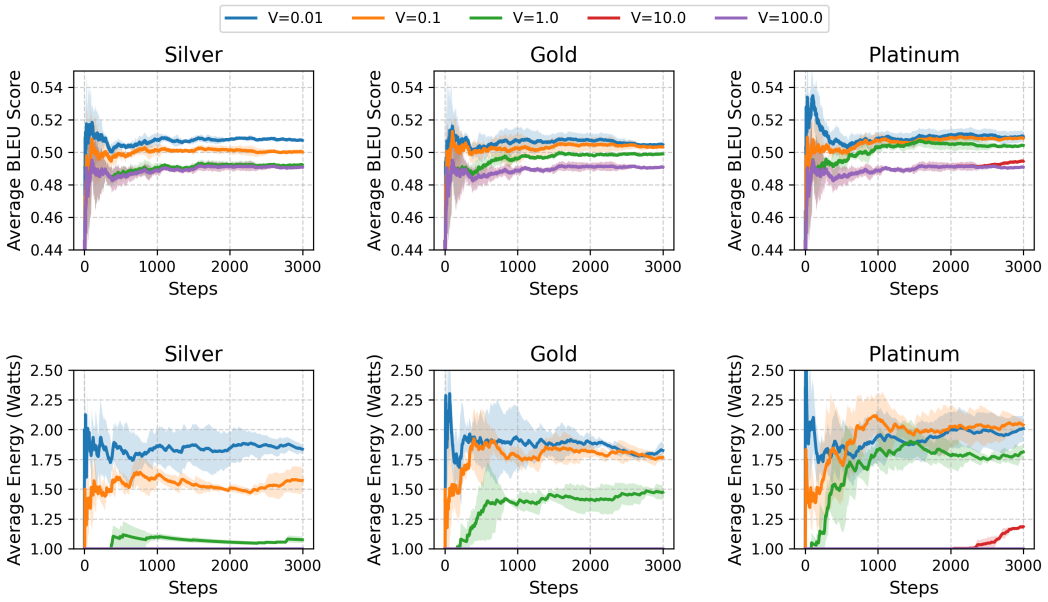
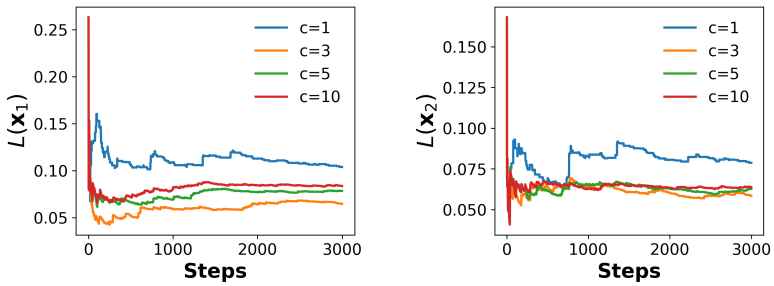


Figure 1: Service Level Guarantees and energy consumption with varying V across different minimum accuracy values α.



(a). Large model **(b).** Small model

Figure 2: We observe that $L(\mathbf{x}_m)$ converges to the lowest loss with $c = 3$.

Generally, for our experimental setup, we find that $V = 0.1$ ensures SLA compliance over time while also minimizing energy consumption.

B.2 Exploring the Likelihood of Exploration in MESS+

With parameter c, we control the exploration likelihood over time. The higher we set c, the higher the likelihood of exploring the model zoo, since $p_t = \min(1, \frac{c}{\sqrt[3]{t}})$, and the number of training steps K becomes larger. However, we decay p_t over time such that we increasingly rely on the model with parameter $\mathbf{x}_{m,t}$ to predict the accuracies of different models for each request t.

We explore the effects of varying levels of c ranging from $[1, 10]$ on the loss of the accuracy predictors, defined in (4), for different models. This is an indicator of how well $\hat{A}_m(t)$ approximates $A_m(t)$. The results are shown in Figure 2.

We find that balancing the level of exploration is important since $c = 1$ leads to less training; therefore, the training samples to learn $\mathbf{x}_m$ are significantly fewer than with $c = 10$, for instance. A large c implies more training of $\mathbf{x}_m$ but can also lead to overfitting on the incoming requests. While the convergence speed is similar for all values of c, we find that $c = 3$ leads to the lowest loss. Therefore, our systematic parameter search for c yields the best accuracy prediction performance with $c = 3$.

We need between 200 and 250 exploration steps for the classifier to fully converge. We also see that over time, each classifier overfits the data, which explains the need for decreasing p_t over time.

The consequence of increased latency is observed as we increase c. As shown in Table 2, the average time per inference request increases as c increases. While MESS+ aims to establish minimum service guarantees and optimize energy consumption, it is important to consider the trade-off with latency. When $c = 10$, multiple models are queried more often compared to $c = 1$. The additional processing introduces greater latency, as querying all M and training $\mathbf{x}_m$ requires more time and resources. Increased latency can negatively impact user experience, especially in real-time services where prompt response time is crucial.

Table 2: More exploration leads to a longer request processing latency. We evaluate the effects of choosing different c values on the average inference time in seconds.

	MESS+ Average Request Processing Duration	
c	**WMT14**	**CNNDailyMail**
1	0.99s	1.11s
3	2.86s	3.13s
5	4.02s	4.34s
10	5.38s	5.76s