# Revisiting Cascaded Ensembles for Efficient Inference

**Steven Kolawole** [1] **Don Dennis** [1] **Ameet Talwalkar** [1] **Virginia Smith** [1]

## Abstract

A common approach to make machine learning inference more efficient is to use *example-specific adaptive* schemes, which route or select models for each example at inference time. In this work we study a simple scheme for adaptive inference. We build a cascade of ensembles (CoE), beginning with resource-efficient models and growing to larger, more expressive models, where ensemble agreement serves as a data-dependent routing criterion. This scheme is easy to incorporate into existing inference pipelines, requires no additional training, and can be used to place models across multiple resource tiers—for instance, serving efficient models at the edge and invoking larger models in the cloud only when necessary. In cases where parallel inference is feasible, we show that CoE can improve accuracy relative to the single best model while reducing the average cost of inference by up to $7\times$, and provides Pareto-dominate solutions in accuracy and efficiency relative to existing adaptive inference baselines. These savings translate to an over $3\times$-reduction in total monetary cost when performing inference using a heterogeneous cluster of GPUs. Finally, for edge inference scenarios where portions of the cascade reside at the edge vs. in the cloud, CoE can provide a $14\times$ reduction in communication cost and inference latency without sacrificing accuracy.

## 1. Introduction

The cost of inference of large models poses a significant barrier to their adoption, particularly in resources-constrained settings (Strubell et al., 2020; Kaplan et al., 2020). As models become larger in pursuit of improved accuracy, practitioners may spend significant resources on improved models, or settle for lower performing models due to hard resource limits. Notably, in many such scenarios, a good portion of the data seen during inference time may be effectively evaluated using much smaller models (Chen et al., 2020; 2023; Jitkrittum et al., 2023; Gupta et al., 2024). This presents an opportunity to lower average inference cost per example—*if* we can determine the least expensive model from a set of models of increasing cost/accuracy to correctly evaluate a given data point. This well-studied problem is often referred to as *adaptive inference*, as the cost of inference adapts to the 'difficulty' of each example.

A natural approach for adaptive inference is to *cascade* over a set of models, starting from the least expensive and stopping based on some exit criterion or deferral rule. Cascades of various forms have been studied in the last few decades (Breiman, 1996; Freund and Schapire, 1997; Rowley et al., 1998; Viola and Jones, 2001; Soo, 2014). More recently, model cascades have seen increasing attention from within the ML community given the escalating size of models (e.g., Chen et al., 2023; Gupta et al., 2024). In its simplest form, a cascade can be constructed using a Pareto-efficient set of models combined with a deferral rule, a common choice being the confidence scores of the models predictions (Viola and Jones, 2001; 2004; Wang et al., 2018a; 2021). Prior work has taken a variety of approaches to enhance model cascades and deferral rules (see full discussion in Section 2). Common approaches include designing model architectures that require the training of new models from scratch (Cai et al., 2019; Devvrit et al., 2023; Khare et al., 2023), or routing schemes that require data-dependent training for every task considered (Chen et al., 2023; Wang et al., 2023; Yue et al., 2024). While these can potentially improve accuracy/efficiency trade-offs, they also incur significant computational overhead in setup costs with additional cost for every added task or data distribution.

In this work, we study a simple deferral rule based on the scores produced by an *ensemble of models* and a voting scheme, which we refer to as a "Cascade of Ensembles" (CoE). When scores from a smaller set of ensembles do not align, they trigger cascading, and inference is generated at the next tier of (larger) models. This scheme has been explored previously for specific applications of face detection (Zuo and de With, 2005; 2008; Susnjak et al., 2012). However, to the best of our knowledge, our work is the first to study the use of ensemble agreement as a generally applicable deferral mechanism in modern ML workloads.

[1]Carnegie Mellon University. Correspondence to: Steven Kolawole <skolawol@andrew.cmu.edu>.

Efficient Systems for Foundation Models. Workshop at the International Conference on Machine Learning (ICML) 2024.

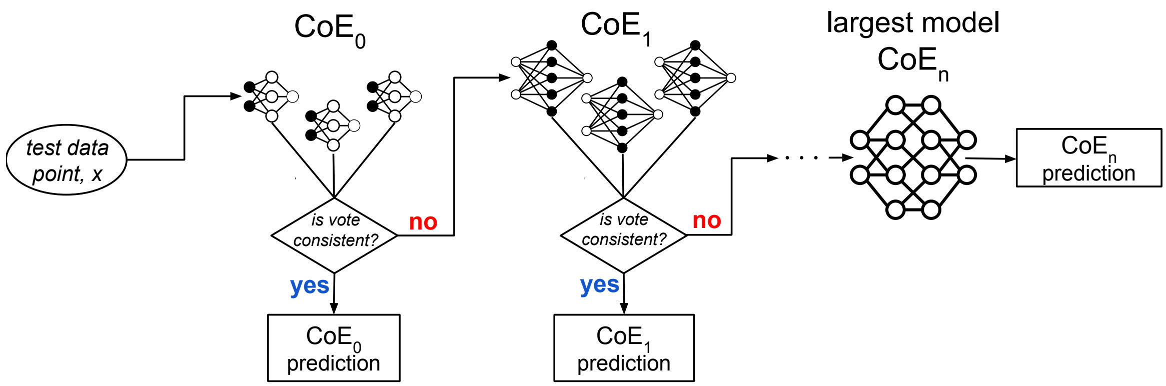


**Figure 1:** *Cascade of Ensembles (CoE)*: Cascading in CoE is triggered by inconsistency in the ensemble models' predictions. If the predictions of the smaller set of ensembles do not align, they trigger cascading, and inference is generated at the next tier of larger models.

We point to two trends that make CoE an increasingly feasible baseline in practice: (1) the growing ease of obtaining models of various sizes and accuracies due to the rise of model zoos (Wolf et al., 2019) as well as flexible compression schemes (Dettmers et al., 2022; Dennis et al., 2023; Dery et al., 2024); and (2) an increased ability to execute ensembles in parallel (or make parallel calls to black-box model APIs) by using intra-machine parallelism or a cluster of potentially heterogeneous machines (Fern and Givan, 2003; Kim et al., 2023; Miao et al., 2023).

With this motivation in mind, we revisit and rigorously study the performance of CoE as a baseline for adaptive inference in modern ML workloads. We provide a comprehensive analysis of how such cascading systems perform, considering not just accuracy and computational efficiency, but also communication costs and monetary expenses in practical settings. This holistic view of efficiency in practical deployment scenarios has been largely overlooked in previous adaptive inference literature. Our results across diverse language and vision tasks demonstrate that the simple idea of voting routinely outperforms—in both accuracy and efficiency metrics—the single, best available model, as well as traditional confidence-based approaches for model cascading. Additionally, CoE establishes judicious and cost-effective performance in hardware-aware environments, such as edge-to-cloud scenarios, where it not only retains its superior accuracy, but also achieves up to a 14-fold reduction in communication costs and inference latency, as well as more than a 3-fold reduction in total monetary cost when performing inference across heterogeneous GPU devices.

## 2. Related Work

**Cascading based on scores.** Various cascading methods, for instance, for object detection (Rowley et al., 1998; Viola and Jones, 2004; Wang et al., 2011; Cai et al., 2015; Angelova et al., 2015; Streeter, 2018), image classification (Wang et al., 2018a; 2021), and text classification (Li et al., 2021; Mamou et al., 2022; Varshney and Baral, 2022; Lebovitz et al., 2023) have been explored in the literature. In these settings, deferrals within cascading systems are usually performed using score-based methods using confidence scores, entropy, and probabilities (Gangrade et al., 2021; Geifman and El-Yaniv, 2019; Narasimhan et al., 2022). However, while using confidence scores of existing models is inexpensive, these scores are required to be well-calibrated, which is less common for off-the-shelf models (Guo et al., 2017; Enomoro and Eda, 2021). Existing methods employ a variety of techniques to work around this, such as difficulty-aware regularization (Li et al., 2021), explicitly adding deferrals as a prediction (Wang et al., 2018a), and calibration mechanisms (Nie et al., 2024). Recently, Jitkrittum et al. (2023) explored the specific practical settings under which confidence-based deferral in cascades could suffer; these settings include: scenarios where downstream models are specialists (the error probability of the downstream model is highly non-uniform across samples), when samples are subject to label noise, and in the presence of a distribution shift between the train and test set. Notably, in all these failure modes, where confidence-based cascades tend to be inadequate, our CoE approach is likely to offer improvements, as ensembles are known to help induce diversity, enable robustness to noise, and mitigate issues of distribution shift (Gontijo-Lopes et al., 2022; Sharkey, 1996; Dietterich, 2000; Džeroski and Ženko, 2004).

**Cascades with routing procedures.** Due to the downsides of confidence scores, certain methods avoid it entirely by training procedures independent of the model set to route instances as needed. Guan et al. (2018) developed a selection module trained to determine the best-fit classifiers for data instances. WILLUMP (Kraft et al., 2020) explored cascading based on feature representation, using cost models to identify important features in binary classification tasks and training approximate models that classify easy data inputs exhibiting these features. More recently, FrugalGPT (Chen et al., 2020; 2023) introduced an LLM cascade strategy that triages different queries in a dataset to different combinations of LLMs. To do this, it introduces two components: a newly trained DistilBERT-based LLM router and a scoring function. AutoMix (Madaan et al., 2023) offers a cascading method for black-box LLM APIs, where the prediction vector and scores are typically unavailable. They propose a few-shot self-verification mechanism and a Markov-based meta-verifier for cascading, which is tested on several context-grounded datasets. Relative to CoE, these more complex methods require additional data-dependent training and assume that the test data distribution is known beforehand, which leads to extra overhead and reduces applicability in practice.

**Dynamic/Adaptive networks.** Routing procedures typically train an independent routing mechanism, keeping the Pareto set of models untouched. However, for many medium-scale applications, methods have been explored that learn a Pareto-set of models and stopping criterion together. For example, early exit methods of (Huang et al., 2018; Wang et al., 2018b; Xin et al., 2020; Schuster et al., 2022), subnetworks extraction methods, (Yu et al., 2018; Yu and Huang, 2019; Chen et al., 2021; Hou et al., 2020; Han et al., 2022; Devvrit et al., 2023), composition-based methods (Suggala et al., 2020; Dennis et al., 2023; Du and Kaelbling, 2024) are some of the more recent examples. CoE does not fit into this category, as we use existing models as they are and do not require training specially designed architectures or training from scratch.

**Cascading and ensembles.** Finally, prior works have explored a specific variant of cascaded ensembles for the application of face detection (Zuo and de With, 2005; 2008; Susnjak et al., 2012). These earlier methods used ensembles of specialized face detectors in their cascade designs, each focusing on a specific sub-region of the face space, with decision networks combining the outputs of these component binary classifiers. In contrast, our work proposes a significantly more general and widely applicable approach—considering the use of cascaded ensembles for modern ML problems involving ensembles of pre-trained neural networks, without incurring setup costs or requiring extensive hyperparameter tuning.

# 3. Cascaded Ensembles for Adaptive Inference

We detail our problem setup here, first outlining standard model selection procedures, and then formalizing the ideas of adaptive inference (Section 3.1) and deferral rules (Section 3.2) before introducing the cascaded ensemble (CoE) approach.

**Standard model selection.** Let $\mathcal{X}$ denote an instance space and $\mathcal{Y} = [L] = \{1, 2, \dots, L\}$ denote the label space. In multi-class classification, we are provided with a data set $\mathcal{D} := \{(x_i, y_i)\}_i \subset \mathcal{X} \times \mathcal{Y}$ sampled according to some unknown distribution $\mathbb{P}$ over $\mathcal{X} \times \mathcal{Y}$. Our goal is to construct a classifier $h : \mathcal{X} \to \mathcal{Y}$ that minimizes the misclassification error or the risk $\mathcal{R}(h) = \mathbb{P}(y \neq h(x))$. Since the population distribution $\mathbb{P}$ is unavailable to us, we minimize the empirical risk over a subset of the data, the training data $\mathcal{D}_{\text{train}}$. In a typical workflow, we use many training runs with potentially different compression techniques to construct a set of candidate classifiers. The *validation error* of all classifiers is computed, by computing the error on an unseen split of the data $\mathcal{D}_{\text{val}}$. This is then used to perform *model selection*; picking the best model to deploy—the underlying theory being that the error on the $\mathcal{D}_{\text{val}}$ is a consistent, unbiased estimate of the error $\mathcal{R}(h) = \mathbb{P}(y \neq h(x))$ we can expect to see during inference. Oftentimes, we are also provided with a scoring function $s : \mathcal{X} \to [0, 1]$ for each classifier $h$, typically constructed from the unscaled classifier outputs.

## 3.1. Deferral Rule Based Adaptive Inference

We can contrast this standard model selection procedure to *adaptive inference*, where instead of picking a single 'best' model for inference, we wish to maintain a set of models and pick models at inference time in a data dependent manner. The focus of this work is on the case where a set containing models with varying cost of inference and performance is provided to us, e.g., potentially via an existing model zoo (Wolf et al., 2019).

Rule-based cascading is a simple, training-free approach to adaptive model selection for efficient inference. Here, the idea is to use the most resource efficient and (likely) least accurate model by default and use a deferral rule to determine if a model with lower error but higher resource cost should be used for the current sample. The intuition is that we can use the efficient classifier to classify 'easy' samples, and reserve the larger classifier for 'harder' samples.

Let $h$ be any classifier and $\mathcal{C} : \mathcal{H} \to \mathbb{R}_+$ be a function such that $\mathcal{C}(h)$ denotes the cost of inference of a classifier $h$. Consider the simple case of a cascade with two models, $\mathcal{M} = \{h_1, h_2\}$ such that $\mathcal{C}(h_1) \leq \gamma\mathcal{C}(h_2)$ for some $\gamma \geq 1$. Let $r : \mathcal{X} \to \{0, 1\}$ denote a deferral rule such that when $r(x) = 1$, we use $h_1$ to evaluate $x$ and defer otherwise. Let $r_{\text{ideal}}$ denote an *optimal deferral rule*, that minimizes the risk across all rules $r : \mathcal{X} \to \{0, 1\}$ under the assumed distribution $\mathbb{P}$,

$$r_{\text{ideal}} \in \arg\min_{r} \mathcal{R}(r; h_1, h_2). \tag{1}$$

While we rarely have access to an optimal rule during inference, we can construct optimal rules in an idealized setting, for instance assuming access to the true label $y$;

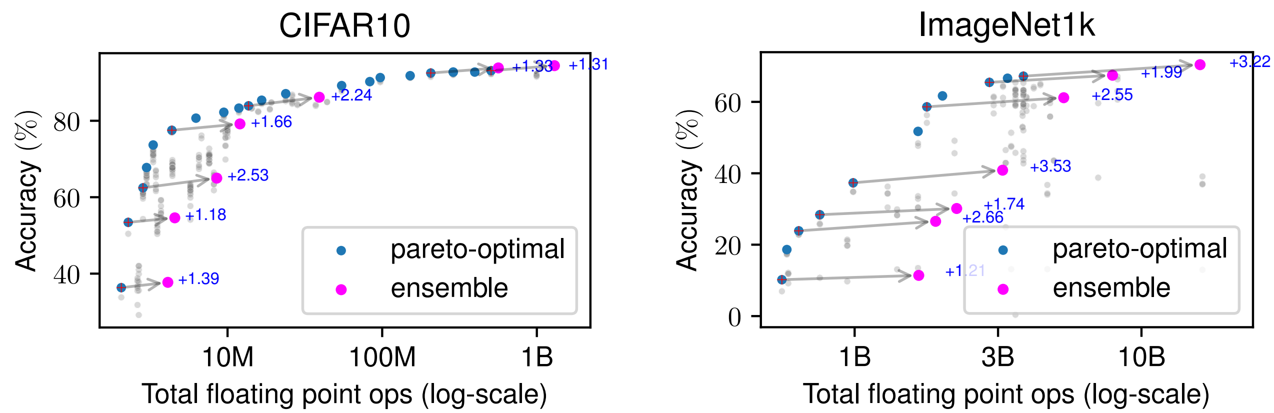


**Figure 2:** Comparing the cost and performance of ensembling to that of pareto set of single models. The change in accuracy over the best single model in the ensemble is marked with an arrow. Ensembling measurably improves accuracy. However, this is not Pareto-optimal; as we show, inference-time vs. accuracy results can be made Pareto-optimal by parallelizing ensemble inference.

**Proposition 1.** *Let $(x, y) \sim \mathbb{P}$ . Then, the deferral rule $r_{ideal}(x) = 1 - \mathbb{I}[h_1(x) = y]$ is optimal and attains $\min_r \mathcal{R}(r; h_1, h_2)$.*

Since we only have access to $x$ and not $y$, we cannot directly employ any of these rules. We instead try to approximate them, and analyze the approximations based on their excess risk in the subsequent subsections.

### 3.2. Approximating the Optimal Deferral Rule

The risk in the standard classification setting can be restated in terms of the conditional distribution $\mathbb{P}(y = h(x) \mid x)$ as

$$\begin{aligned} \mathcal{R}(h) &= \mathbb{P}(y \neq h(x)) = \mathbb{E}\big[\mathbb{I}[y \neq h(x)]\big] \\ &= 1 - \mathbb{E}_x\big[\mathbb{P}(y = h(x) \mid x)\big]. \end{aligned} \tag{2}$$

Consider the classifier, $h^*(x) = \arg\max_y \mathbb{P}(y \mid x)$ that given a data point $x$, assigns the label with the largest probability of being correct under $\mathbb{P}$. This classifier can be shown to attain the optimal risk in (2) and is known as the Bayes optimal classifier. Training is then thought of as constructing as estimator $\hat{p}(y \mid x)$ for $\mathbb{P}(y \mid x)$ and using it to construct the classifier $h(x) = \arg\max_y \hat{p}(y \mid x)$.

This point of view readily applies when using a cascade classifier with a deferral rule $r$. Consider the cascade with two models $\mathcal{M} = \{h_1, h_2\}$ and a deferral rule $r$. The risk can be stated as

$$\mathcal{R}(r; h_1, h_2) = 1 - \mathbb{E}_x\big[\mathbb{P}(y = h(x; r, h_1, h_2) \mid x)\big].$$

We can define the *Bayes optimal deferral rule* as the one that attains the minimum Bayes risk. The optimal rule $r_{\text{ideal}}$ defined in Proposition 1 can be used to construct Bayes optimal rule, if we assume access to the conditional distribution $\mathbb{P}(h(x) = y \mid x)$. In particular, the scoring function $s(x)$ associated with each classifier $h(x)$ can be argued to be an estimator of $\mathbb{P}(h(x) = y \mid x)$ and used to construct a deferral rule, as stated here

$$r_\theta(x) = \begin{cases} 1 & s_1(x) \geq \theta \\ 0 & s_1(x) < \theta. \end{cases}$$

This is the point of view taken by Jitkrittum et al. (2023), for instance, where they consider the excess risk and design various estimators for it based on insights from their analysis. We consider a simple modification of this deferral rule where, instead of using a single model $h_1$ at the first level, we use an ensemble of models. Ensembles often reduce the variance in prediction scores, resulting in an accuracy increase (e.g., Figure 2, (García-Pedrajas et al., 2005; Gontijo-Lopes et al., 2022)). We use the voted majority prediction of an ensemble of models and the number of votes for the prediction to construct our deferral rule.

**Algorithm 1** Deferral Based Cascade

**Require:** Set of ensembles $\mathcal{M} = \{H_1, H_2, \ldots, H_{n_E}\}$, voting threshold $\theta_v$
**Require:** A new inference data point $x$.
  Current cascade tier, $i \leftarrow 1$
  Cascaded prediction, $y \leftarrow \emptyset$
  **for** $i \in \{1, \ldots, n_E\}$ **do**
    $y \leftarrow H_i(x)$
    **if** vote$(x, H_i) \geq \theta_v$ **then**
      **break**
    **end if**
  **end for**
  **return** $y$

**CoE deferral rule.** Concretely, let $H(x)$ denote an ensemble classifier consisting of models $\{h_1, h_2, \ldots, h_{n_E}\}$.

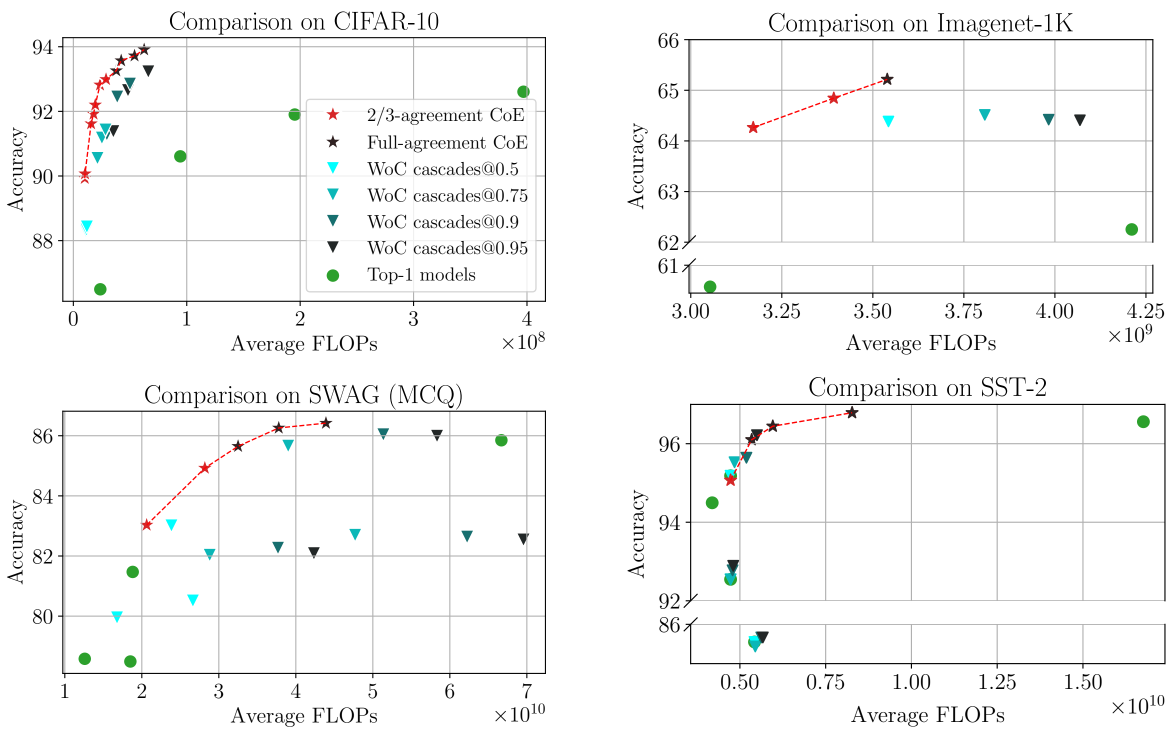


**Figure 3:** Pareto curves of CoE vs. confidence-based cascades (WoC) (Wang et al., 2021) and best single models on diverse tasks. We alternate between $\frac{2}{3}$ and 100% agreement consistency for CoE's tier models. For WoC, we tune its cascade configurations across the best four of its confidence thresholds and generate results from their most performant cascades. CoE maintains a Pareto-optimal curve, which consistently outperforms both methods in accuracy with lower FLOPs costs.

Let $\text{vote}(x; H) = \frac{1}{n_E} \sum_h \mathbb{I}[H(x) = h(x)]$ denote the fraction of votes received by the majority prediction, $H(x)$ and let $s(x; H)$ denote the average score of the majority prediction. Similar to before, we can now consider the cascade constructed from two ensembles $\mathcal{M} = \{H_1, H_2\}$ and the ensemble based deferral rule, parameterized by a voting threshold $\theta_v \in [0, n_E]$ ,

$$r_{\text{ensbl}}(x; \theta_v) = \begin{cases} 1 & \text{vote}(s; H_1) \geq \theta_v \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

While this deferral rule is designed for a two stage ensemble cascade, we can extend this to more tiers chosen in a problem specific manner as specified in Algorithm 1.

## 4. Experiments

**Experimental setup and procedures.** To evaluate the performance of CoE, we conduct a series of experiments using benchmark datasets for various image and language tasks. The datasets include CIFAR-10 (Krizhevsky and Hinton, 2009) and ImageNet-1K (Deng et al., 2009) for image classification, SST-2 (Socher et al., 2013) and Twitter Financial News for sentiment analysis, and SWAG (Zellers et al., 2018) for multiple-choice question answering. We use a diverse collection of models across both task modalities to form our ensembles. For language tasks, we use the `BASE` and `LARGE` models from the model families of BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), XLNet (Yang et al., 2019), and ELECTRA (Clark et al., 2020). All models are routinely picked from HuggingFace Zoo (Wolf et al., 2019) for inference generation without any additional training effort on our end.

As described in Algorithm 1, we construct CoE by starting with the most resource-efficient models and cascading tier by tier until the predictions of the models in a given tier are consistent or inference is generated at the last tier. We consider cascades where either all or a majority (e.g., $\frac{2}{3}$) of the models' predictions agree.

For evaluation, we measure inference accuracy and efficiency using key indicators such as floating point operations (FLOPs) and inference latency. The latter was particularly emphasized in hardware-aware scenarios where edge device constraints were simulated. All metrics are measured on the respective data test sets. For most of our experiments, we assume that inference can be generated in parallel in accordance with recent advances in intra- and inter-machine parallelism; hence, we take the numbers of the least efficient models in each ensemble tier for the efficiency metrics.

### 4.1. Adaptive Model Selection

As baselines, we compare CoE directly—as shown in Figure 3—with the best singular models and Wisdom-of-Committees (WoC) (Wang et al., 2021), a popular and representative confidence-based model cascading method.

Each ensemble tier within CoE has a number of models ranging from 2 to 5, which are flexible based on the users' resource constraints. We maintain a consistent number of models across all ensembles, with the last tier containing just the best singular model. We also enable a flexible agreement scheme where $\frac{2}{3}$ of the models is enough to ensure consistency for cascading, as opposed to having all models agree to pass the consistency check. Intuitively, this helps to establish a more flexible accuracy-efficiency pareto curve, given that cascading would be triggered less if a smaller number of models were needed to verify consistency.

For the WoC method, we consider a scenario that is advantageous to this approach by selecting the *best singular model* from each performance tier and cascading between them. Since the optimal confidence cascade threshold for a given task cannot be known beforehand, we perform hyperparameter tuning by cascading using a wide range of confidence thresholds.

For each run, we perform iterative experiments, varying the length of the cascade tiers based on the availability of distinct performance tiers. For example, we can cascade up to 5 tiers for the CIFAR-10 task, but we have a tier of two for language tasks since we typically only cascade between the `BASE` models and the best single `LARGE` model. This setting applies to CoE and the confidence-based WoC approach.

**Accuracy-FLOPs trade-offs.** As shown in Figure 3, both cascading approaches match or exceed the accuracy of the best singular models. However, it should be noted that for WoC, an optimal threshold that works for all data distributions or taks cannot be easily determined beforehand, as shown by the various locations of suboptimal WoC cascades in the results. Previous work usually selects thresholds by performing grid searches on held-out validation sets (Wang et al., 2021). On the other hand, CoE shows a stable Pareto curve that provides better accuracy than the singular models (and WoC cascades) available at similar costs can offer.

### 4.2. Model Cascading in Hardware-aware Settings

We additionally consider simulating scenarios that resemble (1) multi-GPU settings by measuring (both monetary and inference) the costs of utilizing cascades on GPUs with cost estimates from Lambda Cloud (Lambda, 2024) and (2) edge-to-cloud deployment by accounting for additional latency between 'edge' devices (e.g., smartphones) and 'cloud' servers to study the effect of cascading in a real-world distributed setting.

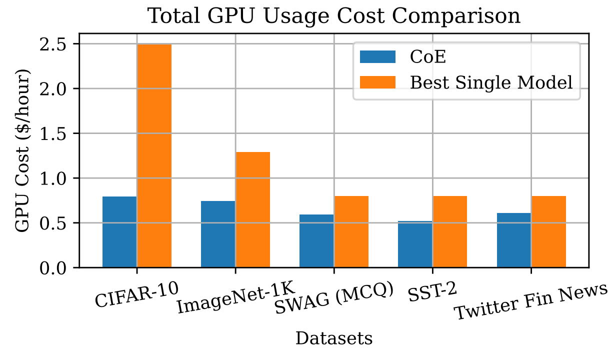


**Figure 4:** Total GPU usage costs of CoE vs. using the best model. Cascade of ensembles, at reduced costs of GPU usage, exceed the accuracy of the single best models in all task categories.

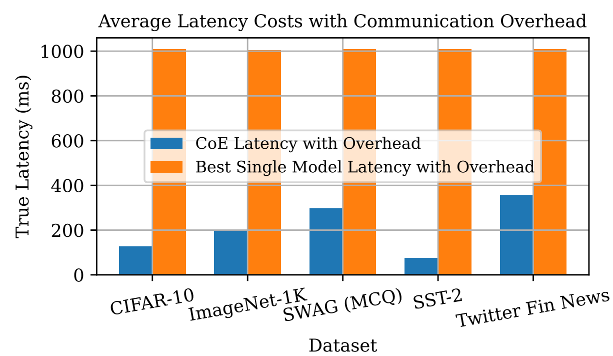


**Figure 5:** Edge-to-Cloud Cascading: We simulate a single-instance inference setup, as might be seen in real-time applications where predictions may need to be made as new data becomes available. Cascading systems, as shown with CoE, can enable small models to be served at the edge without sacrificing accuracy—leading to large savings in communication costs over the alternative of using only the highest accuracy/largest model residing in the cloud.

**GPU costs.** We retrieve the costs of GPU usage by hour from Lambda Cloud's pricing (Table 1). For simplicity, we assume that each ensemble tier is set up on the equivalent, distinct GPU type in increasing order of sophistication. We show a summary in Figure 4 and present the detailed results in Table 2. The fraction of test samples denotes the number of data points that exited (after consistent agreement) at each tier. It is worth noting in these results that while larger model tiers are also used, cascading ensures that they are used sparingly, as needed, and this helps to keep the costs (in terms of GPU costs, latency, and FLOPs) low. In fact, using CoE leads to outperforming the best singular model in each task on all three metrics for efficiency *while also* improving upon the best model in terms accuracy.

**Communication overhead.** To evaluate the real-world performance of our cascaded inference system, we must consider the impact of communication overhead, as would occur in practical settings. Since this overhead (latency delays incurred due to factors like network speed, serialization overhead, network congestion, etc.) can easily exceed inference latency, we use known latency setup numbers of edge devices (i.e., Raspberry Pis and smartphones) and cloud servers from Zhu et al. (2021); Lai et al. (2022)'s benchmarks. The delay parameters adopted range from small, medium, to large [1 us, 10 ms, 100 ms, 1000 ms], where near-instantaneous local communication (< 1 microsecond) can be expected to occur with base cascade tiers performing inference on-device, and substantial network delays might occur (> 1 second) in a worst-case edge-to-cloud transmission. Our experiments simulate these delays by applying them to the cascade exit points for each dataset, thus capturing the time cost of transitioning between tiers or devices in a practical setting. Specifically, we introduced the smallest delay at the first cascade tier—where inference might feasibly occur on-device with minimal latency—and the largest delay at the final cascade tier, which assumes the most significant latency scenario involving edge-to-cloud transmission.

Our results, as shown in Figure 5, reveal that even with the added complexity of cascading, judicious layering of models can lead to significant latency reductions along with superior accuracy compared to relying on a single, potentially resource-intensive model, particularly in scenarios where communication costs are large yet accuracy is of great importance. In particular, we see that cascading in these scenarios provides an even greater $14\times$ reduction in communication cost relative to performing inference using the best single model alone.

## 5. Discussion & Future Work

In this work, we explore a straightforward approach for implementing cascaded inference—using existing models in their current form and capitalizing on the concept of mutual prediction consistency through a *cascade of ensembles* to inform deferral decisions. We formulated the challenge of devising an effective deferral rule as a problem of statistical estimation and provided insights into the conditions under which cascading is likely to yield benefits. Our empirical evaluations spanned numerous tasks and cascading tiers, where our approach not only outperformed existing methods but did so without incurring the costs typically associated with constructing functional cascade frameworks. Additionally, we assessed our cascade methodology's deferral rates in on-device inference contexts, demonstrating potential communication and cost gains corresponding to various deferral rates.

Looking ahead, several avenues for advancing this work merit consideration. These include: (i) building more diverse CoE by extending our collection through the use of model compression techniques; (ii) extending the cascading approach to incorporate additional data and task modalities, e.g., audio; and (iii) addressing the complex problem of using cascading inference mechanisms on open-ended generation tasks (Gupta et al., 2024). These directions can potentially extend the robustness and efficiency of cascading inference systems, paving the way for broader and more adaptable applications in real-world settings.

## Acknowledgements

This work was supported in part by the National Science Foundation grants IIS1705121, IIS1838017, IIS2046613, IIS2112471, and funding from Meta, Morgan Stanley, Amazon, Google, and Scribe. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of any of these funding agencies.

## References

A. Angelova, A. Krizhevsky, V. Vanhoucke, A. S. Ogale, and D. Ferguson. Real-time pedestrian detection with deep network cascades. volume 2, page 4, 2015.

L. Breiman. Bagging predictors. *Machine Learning*, 24(2): 123–140, Aug. 1996. ISSN 1573-0565. doi: 10.1007/BF00058655.

H. Cai, C. Gan, T. Wang, Z. Zhang, and S. Han. Once-for-All: Train One Network and Specialize it for Efficient Deployment, Apr. 2019.

Z. Cai, M. Saberian, and N. Vasconcelos. Learning Complexity-Aware Cascades for Deep Pedestrian Detection. pages 3361–3369, 2015.

L. Chen, M. Zaharia, and J. Y. Zou. FrugalML: How to use ML Prediction APIs more accurately and cheaply. In *Advances in Neural Information Processing Systems*, 2020.

L. Chen, M. Zaharia, and J. Zou. FrugalGPT: How to use large language models while reducing cost and improving performance. *arXiv preprint arXiv:2305.05176*, 2023.

M. Chen, H. Peng, J. Fu, and H. Ling. Autoformer: Searching transformers for visual recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12270–12280, October 2021.

K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*, 2020.

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

D. Dennis, A. Shetty, A. P. Sevekari, K. Koishida, and V. Smith. Progressive ensemble distillation: Building ensembles for efficient inference. *Advances in Neural Information Processing Systems*, 36, 2023.

L. Dery, S. Kolawole, J.-F. Kagey, V. Smith, G. Neubig, and A. Talwalkar. Everybody prune now: Structured pruning of llms with only forward passes. *arXiv preprint arXiv:2402.05406*, 2024.

T. Dettmers, M. Lewis, Y. Belkada, and L. Zettlemoyer. Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale. *Advances in Neural Information Processing Systems*, 35:30318–30332, 2022.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics*, 2019.

Devvrit, S. Kudugunta, A. Kusupati, T. Dettmers, K. Chen, I. Dhillon, Y. Tsvetkov, H. Hajishirzi, S. Kakade, A. Farhadi, and P. Jain. MatFormer: Nested Transformer for Elastic Inference, Oct. 2023.

T. G. Dietterich. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer, 2000.

Y. Du and L. Kaelbling. Compositional Generative Modeling: A Single Model is Not All You Need, Feb. 2024. arXiv:2402.01103 [cs].

S. Džeroski and B. Ženko. Is combining classifiers with stacking better than selecting the best one? *Machine learning*, 54:255–273, 2004.

S. Enomoro and T. Eda. Learning to cascade: Confidence calibration for improving the accuracy and computational cost of cascade inference systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 7331–7339, 2021.

A. Fern and R. Givan. Online ensemble learning: An empirical study. *Machine Learning*, 53:71–109, 2003.

Y. Freund and R. E. Schapire. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Journal of Computer and System Sciences*, 55(1):119–139, Aug. 1997. ISSN 0022-0000. doi: 10.1006/jcss.1997.1504.

A. Gangrade, A. Kag, and V. Saligrama. Selective Classification via One-Sided Prediction. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, pages 2179–2187. PMLR, Mar. 2021. ISSN: 2640-3498.

N. García-Pedrajas, D. Ortiz-Boyer, R. del Castillo-Gomariz, and C. Hervás-Martínez. Cascade ensembles. In *Computational Intelligence and Bioinspired Systems: 8th International Work-Conference on Artificial Neural Networks. Proceedings 8*, pages 598–603. Springer, 2005.

Y. Geifman and R. El-Yaniv. SelectiveNet: A Deep Neural Network with an Integrated Reject Option. In *Proceedings of the 36th International Conference on Machine Learning*, pages 2151–2159. PMLR, May 2019. ISSN: 2640-3498.

R. Gontijo-Lopes, Y. Dauphin, and E. D. Cubuk. No One Representation to Rule Them All: Overlapping Features of Training Methods, Apr. 2022. arXiv:2110.12899 [cs].

J. Guan, Y. Liu, Q. Liu, and J. Peng. Energy-efficient Amortized Inference with Cascaded Deep Classifiers. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, pages 2184–2190, Stockholm, Sweden, July 2018. International Joint Conferences on Artificial Intelligence Organization. ISBN 978-0-9992411-2-7. doi: 10.24963/ijcai.2018/302.

C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger. On calibration of modern neural networks. In D. Precup and Y. W. Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330. PMLR, 06–11 Aug 2017.

N. Gupta, H. Narasimhan, W. Jitkrittum, A. S. Rawat, A. K. Menon, and S. Kumar. Language model cascades: Token-level uncertainty and beyond. In *International Conference on Learning Representations*, 2024.

Y. Han, G. Huang, S. Song, L. Yang, H. Wang, and Y. Wang. Dynamic Neural Networks: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11): 7436–7456, Nov. 2022. ISSN 1939-3539. doi: 10.1109/TPAMI.2021.3117837.

L. Hou, Z. Huang, L. Shang, X. Jiang, X. Chen, and Q. Liu. DynaBERT: Dynamic BERT with Adaptive Width and Depth. In *Advances in Neural Information Processing Systems*, volume 33, pages 9782–9793. Curran Associates, Inc., 2020.

G. Huang, D. Chen, T. Li, F. Wu, L. van der Maaten, and K. Q. Weinberger. Multi-Scale Dense Networks for Resource Efficient Image Classification, June 2018. arXiv:1703.09844.

W. Jitkrittum, N. Gupta, A. K. Menon, H. Narasimhan, A. Rawat, and S. Kumar. When Does Confidence-Based Cascade Deferral Suffice? *Advances in Neural Information Processing Systems*, 36:9891–9906, Dec. 2023.

J. Kaplan, S. McCandlish, T. J. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei. Scaling laws for neural language models. *ArXiv*, abs/2001.08361, 2020.

A. Khare, D. Garg, S. Kalra, S. Grandhi, I. Stoica, and A. Tumanov. SuperServe: Fine-Grained Inference Serving for Unpredictable Workloads, Dec. 2023. arXiv:2312.16733 [cs].

S. Kim, S. Moon, R. Tabrizi, N. Lee, M. W. Mahoney, K. Keutzer, and A. Gholami. An llm compiler for parallel function calling. *ArXiv*, abs/2312.04511, 2023.

P. Kraft, D. Kang, D. Narayanan, S. Palkar, P. Bailis, and M. Zaharia. Willump: A statistically-aware end-to-end optimizer for machine learning inference. *Proceedings of Machine Learning and Systems*, 2:147–159, 2020.

A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. 2009.

F. Lai, Y. Dai, S. Singapuram, J. Liu, X. Zhu, H. Madhyastha, and M. Chowdhury. Fedscale: Benchmarking model and system performance of federated learning at scale. In *International conference on machine learning*, pages 11814–11827. PMLR, 2022.

Lambda. GPU Cloud - VMs for Deep Learning | Lambda, 2024. URL `https://lambdalabs.com/service/gpu-cloud`.

L. Lebovitz, L. Cavigelli, M. Magno, and L. K. Muller. Efficient Inference With Model Cascades. *Transactions on Machine Learning Research*, 2023.

L. Li, Y. Lin, D. Chen, S. Ren, P. Li, J. Zhou, and X. Sun. CascadeBERT: Accelerating Inference of Pre-trained Language Models via Calibrated Complete Models Cascade. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 475–486, Punta Cana, Dominican Republic, Nov. 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-emnlp.43.

Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

A. Madaan, P. Aggarwal, A. Anand, S. P. Potharaju, S. Mishra, P. Zhou, A. Gupta, D. Rajagopal, K. Kappaganthu, Y. Yang, S. Upadhyay, Mausam, and M. Faruqui. AutoMix: Automatically Mixing Language Models, Oct. 2023. arXiv:2310.12963 [cs].

J. Mamou, O. Pereg, M. Wasserblat, and R. Schwartz. TangoBERT: Reducing Inference Cost by using Cascaded Architecture, Apr. 2022. arXiv:2204.06271 [cs].

X. Miao, G. Oliaro, Z. Zhang, X. Cheng, H. Jin, T. Chen, and Z. Jia. Towards efficient generative large language model serving: A survey from algorithms to systems. *ArXiv*, abs/2312.15234, 2023.

H. Narasimhan, W. Jitkrittum, A. K. Menon, A. Rawat, and S. Kumar. Post-hoc estimators for learning to defer to an expert. *Advances in Neural Information Processing Systems*, 35:29292–29304, Dec. 2022.

L. Nie, Z. Ding, E. Hu, C. Jermaine, and S. Chaudhuri. Online cascade learning for efficient inference over streams. *arXiv preprint arXiv:2402.04513*, 2024.

H. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(1):23–38, Jan. 1998. ISSN 1939-3539. doi: 10.1109/34.655647.

T. Schuster, A. Fisch, J. Gupta, M. Dehghani, D. Bahri, V. Tran, Y. Tay, and D. Metzler. Confident Adaptive Language Modeling. *Advances in Neural Information Processing Systems*, 35:17456–17472, Dec. 2022.

A. J. C. Sharkey. On combining artificial neural nets. *Connection Science*, 8(3-4):299–314, 1996. doi: 10.1080/095400996116785.

R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

S. Soo. Object detection using Haar-cascade Classifier. *Institute of Computer Science, University of Tartu*, 2(3): 1–12, 2014.

M. Streeter. Approximation Algorithms for Cascading Prediction Models. In *Proceedings of the 35th International Conference on Machine Learning*, pages 4752–4760. PMLR, July 2018. ISSN: 2640-3498.

E. Strubell, A. Ganesh, and A. McCallum. Energy and policy considerations for modern deep learning research. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 13693–13696, 2020.

A. Suggala, B. Liu, and P. Ravikumar. Generalized boosting. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 8787–8797, 2020.

T. Susnjak, A. L. C. Barczak, and K. A. Hawick. Adaptive cascade of boosted ensembles for face detection in concept drift. *Neural Computing and Applications*, 21: 671–682, 2012.

N. Varshney and C. Baral. Model Cascading: Towards Jointly Improving Efficiency and Accuracy of NLP Systems. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11007–11021, Abu Dhabi, United Arab Emirates, 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.756.

P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, Dec. 2001. doi: 10.1109/CVPR.2001.990517. ISSN: 1063-6919.

P. Viola and M. J. Jones. Robust Real-Time Face Detection. *International Journal of Computer Vision*, 57(2): 137–154, May 2004. ISSN 1573-1405. doi: 10.1023/B:VISI.0000013087.49260.fb.

H. Wang, F. M. Polo, Y. Sun, S. Kundu, E. Xing, and M. Yurochkin. Fusing models with complementary expertise. *arXiv preprint arXiv:2310.01542*, 2023.

L. Wang, J. Lin, and D. Metzler. A cascade ranking model for efficient ranked retrieval. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 105–114. ACM, 2011. doi: 10.1145/2009916.2009934.

X. Wang, Y. Luo, D. Crankshaw, A. Tumanov, F. Yu, and J. E. Gonzalez. IDK Cascades: Fast Deep Learning by Learning not to Overthink, June 2018a. arXiv:1706.00885 [cs].

X. Wang, F. Yu, Z.-Y. Dou, T. Darrell, and J. E. Gonzalez. SkipNet: Learning Dynamic Routing in Convolutional Networks. pages 409–424, 2018b.

X. Wang, D. Kondratyuk, E. Christiansen, K. M. Kitani, Y. Movshovitz-Attias, and E. Eban. Wisdom of committees: An overlooked approach to faster and more accurate models. In *International Conference on Learning Representations*, 2021.

T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

J. Xin, R. Tang, J. Lee, Y. Yu, and J. Lin. Deebert: Dynamic early exiting for accelerating bert inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2246–2251, 2020.

Z. Yang, Z. Dai, Y. Yang, J. G. Carbonell, R. Salakhutdinov, and Q. V. Le. Xlnet: Generalized autoregressive pretraining for language understanding. In *Neural Information Processing Systems*, 2019.

J. Yu and T. S. Huang. Universally Slimmable Networks and Improved Training Techniques. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1803–1811, 2019.

J. Yu, L. Yang, N. Xu, J. Yang, and T. Huang. Slimmable Neural Networks, Dec. 2018. arXiv:1812.08928 [cs].

M. Yue, J. Zhao, M. Zhang, L. Du, and Z. Yao. Large language model cascades with mixture of thought representations for cost-efficient reasoning. In *The Twelfth International Conference on Learning Representations*, 2024.

R. Zellers, Y. Bisk, R. Schwartz, and Y. Choi. SWAG: A large-scale adversarial dataset for grounded commonsense inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 93–104, Oct.-Nov. 2018. doi: 10.18653/v1/D18-1009.

L. Zhu, H. Lin, Y. Lu, Y. Lin, and S. Han. Delayed gradient averaging: Tolerate the communication latency for federated learning. *Advances in Neural Information Processing Systems*, 34:29995–30007, 2021.

F. Zuo and P. H. N. de With. Fast face detection using a cascade of neural network ensembles. In *Advanced Concepts for Intelligent Vision Systems Conference*, 2005.

F. Zuo and P. H. N. de With. Cascaded face detection using neural network ensembles. *EURASIP Journal on Advances in Signal Processing*, 2008:1–13, 2008.

# A. Model Cascading

## A.1. Cost Benefits

Based on Section 4.2, Table 1 shows the GPU pricings across several tiers retrieved from Lambda Cloud; Table 2 shows a detailed analysis of costs across all cascade depths, associated with the number of cascade exits at each cascade depths, to provide holistic efficiency comparisons of the aggregated cascade costs against using the best and most resource-intensive model with cascading dominating in every single aspect; Figure 6 certifies the assumption that larger models are significantly better, ensuring a pareto curve, which must be a guarantee for cascading to be functional, and; Figure 7 shows an in-depth analysis of how often cascade exits occur across depths, given the number of models participating in the decision-making process in each cascade tier. Typically, most cascade exits occur in the early rounds (as also shown in Table 2), ensuring that the more cost-intensive cascade tiers featuring more complex models are reserved for the harder test instances.

| GPU | Cost per Hour (USD) |
|---|---|
| V100 | 0.5 |
| A6000 | 0.8 |
| A100 | 1.29 |
| H100 | 2.49 |

**Table 1:** GPU usage costs estimates from Lambda Cloud (Lambda, 2024).

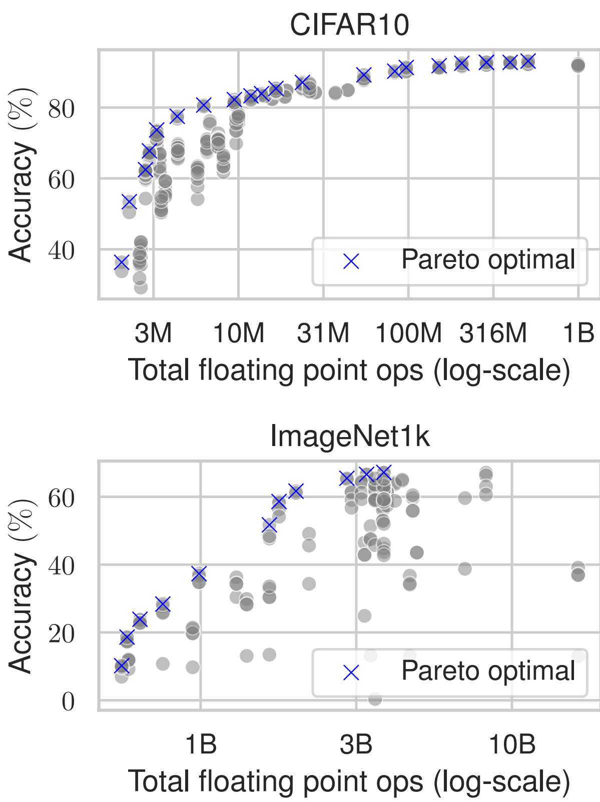


**Figure 6:** Pareto optimal set of models for two datasets.

### A.2. Parallel vs. sequential inference execution

Figure 2 shows that ensembling measurably improves accuracy, although at higher FLOPs. But this is made pareto optimal by parallelizing ensembles, as we did with CoE, given the increased ability to easily execute ensembles in parallel by leveraging intra-machine parallelism. In Figure 8, we show the superiority of parallel inference execution for cascading over using the best single model. We also show that in the (unlikely) worst-case scenarios in which every single inference has to be sequentially produced over each ensemble and cascade, there are considerable (albeit much reduced) savings over the largest single models at still better accuracy.

### A.3. Error Analysis

To analyze how we can improve our current system even better, we investigate the cases where the ensemble strategy yields incorrect predictions despite the consensus among the models. This means that cascading is not triggered, even though the smaller models are wrong. To understand why and what the best singular model would do in these cases, we examine the relationship between the test instances that produce incorrect agreements and how the large model performs in those cases, as shown in Table 3. The numbers indicate that the proportions of incorrect large model predictions in instances where incorrect agreements occur range from half to $\frac{2}{3}$ of the incorrect agreements. This signifies that the difficulty of the test instances may be the primary reason for wrong agreements, since the large model is failing at most of these instances as well. In particular, ensuring that all models agree mitigates less reliable consensus to a good extent. Future work can explore how to completely eliminate this scenario without additional overhead (metaverifiers usually require task-specific training, and this complexity scales linearly with every new task considered).

**Table 2:** Table describing various metrics at different cascade tiers for each dataset. Each dataset's row consists of the following metrics: the fraction of samples processed at each depth, the cost of Total GPU usage per hour at each tier, the average latency of the specific tier (in milliseconds), and the average FLOPs at each tier (floating-point operations). The CoE column for each metric represents the entire cascading system's average across all cascade depths.

| Dataset | Metric | Tier 1 | Tier 2 | Tier 3 | Tier 4 | CoE | Best Single Model |
|---|---|---|---|---|---|---|---|
| **CIFAR-10** | Frac. Samples (total=10,000) | 0.73 | 0.09 | 0.08 | 0.10 | 1.00 | 1.00 |
| | Total GPU Cost ($ / hour) | 0.36 | 0.07 | 0.11 | 0.24 | 0.79 | 2.49 |
| | Avg. Latency (ms) | 3.11 | 3.79 | 7.76 | 9.07 | 4.13 | 9.07 |
| | Avg. FLOPs | 5.42e6 | 2.32e7 | 1.16e8 | 2.47e8 | 3.97e7 | 2.48e8 |
| **ImageNet-1K** | Frac. Samples (total=50,000) | 0.52 | 0.29 | 0.19 | - | 1.00 | 1.00 |
| | Cost ($ / hour) | 0.26 | 0.23 | 0.25 | - | 0.74 | 1.29 |
| | Avg. Latency (ms) | 2.45 | 2.88 | 3.17 | - | 2.71 | 3.17 |
| | Avg. FLOPs | 2.15e9 | 3.90e9 | 4.30e9 | - | 3.07e9 | 4.30e9 |
| **SWAG (MCQ)** | Frac. Samples (total=20,006) | 0.71 | 0.29 | - | - | 1.00 | 1.00 |
| | Cost ($ / hour) | 0.36 | 0.23 | - | - | 0.59 | 0.80 |
| | Avg. Latency (ms) | 4.52 | 8.05 | - | - | 5.53 | 8.05 |
| | Avg. FLOPs | 1.88e10 | 6.67e10 | - | - | 3.25e10 | 6.67e10 |
| **SST-2** | Frac. Samples (total=872) | 0.93 | 0.07 | - | - | 1.00 | 1.00 |
| | Cost ($ / hour) | 0.46 | 0.06 | - | - | 0.52 | 0.80 |
| | Avg. Latency (ms) | 3.88 | 7.22 | - | - | 4.13 | 7.22 |
| | Avg. FLOPs | 5.43e9 | 1.68e10 | - | - | 6.26e9 | 1.68e10 |
| **Twitter Fin News** | Frac. Samples (total=822) | 0.65 | 0.35 | - | - | 1.00 | 1.00 |
| | Cost ($ / hour) | 0.32 | 0.28 | - | - | 0.61 | 0.80 |
| | Avg. Latency (ms) | 4.05 | 7.26 | - | - | 5.19 | 7.26 |
| | Avg. FLOPs | 6.83e9 | 2.42e10 | - | - | 1.30e10 | 2.42e10 |

**Table 3:** Wrong agreements vs. incorrect large model predictions: The percentage of large model incorrect predictions for test instances that produced incorrect agreements tends to range between $\frac{1}{2}$ and $\frac{3}{4}$ of the incorrect agreement numbers.

| | SWAG (20,000 samples) | | | SST-2 (872 samples) | | | Twitter Financial News (822 samples) | | |
|---|---|---|---|---|---|---|---|---|---|
| **CoE config** | **Wrong agreement** | **Big model** | **Rate %** | **Wrong agreement** | **Big model** | **Rate %** | **Wrong agreement** | **Big model** | **Rate %** |
| 2 models, threshold=1.0 | 1672 | 992 | 59.33 | 29 | 21 | 72.41 | 31 | 19 | 61.29 |
| 3 models, threshold=0.66 | 3198 | 1647 | 51.5 | 43 | 24 | 55.81 | 92 | 41 | 44.57 |
| 3 models, threshold=1.0 | 1119 | 738 | 65.95 | 24 | 19 | 79.17 | 23 | 16 | 69.57 |
| 4 models, threshold=0.66 | 2141 | 213 | 56.66 | 38 | 24 | 63.16 | 70 | 34 | 48.57 |
| 4 models, threshold=1.0 | 781 | 544 | 69.65 | 16 | 14 | 87.50 | 17 | 12 | 70.59 |

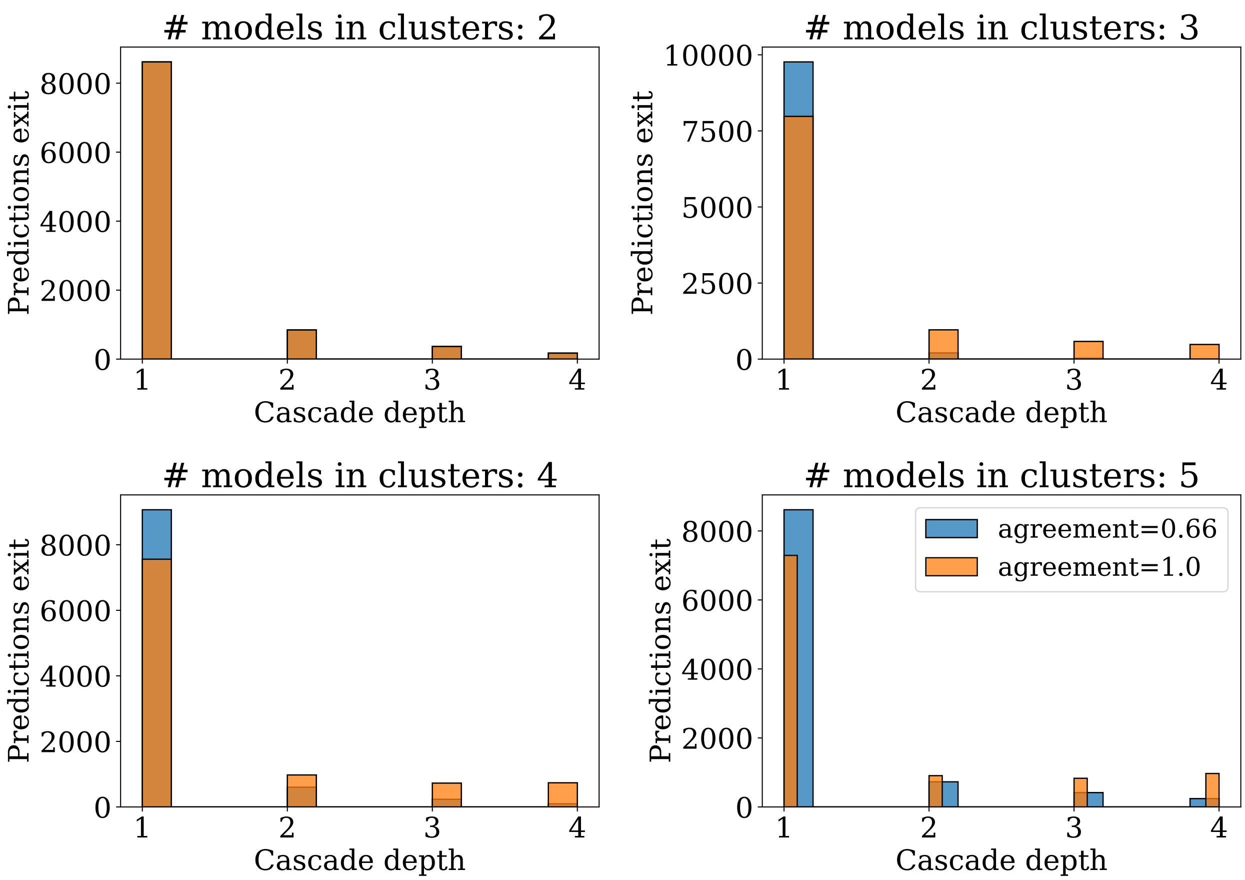


**Figure 7: where the 'average savings' come from** The above plot shows the distribution of CoE's performance tiers on CIFAR-10. Note how the distribution flattens out as we increase the threshold.

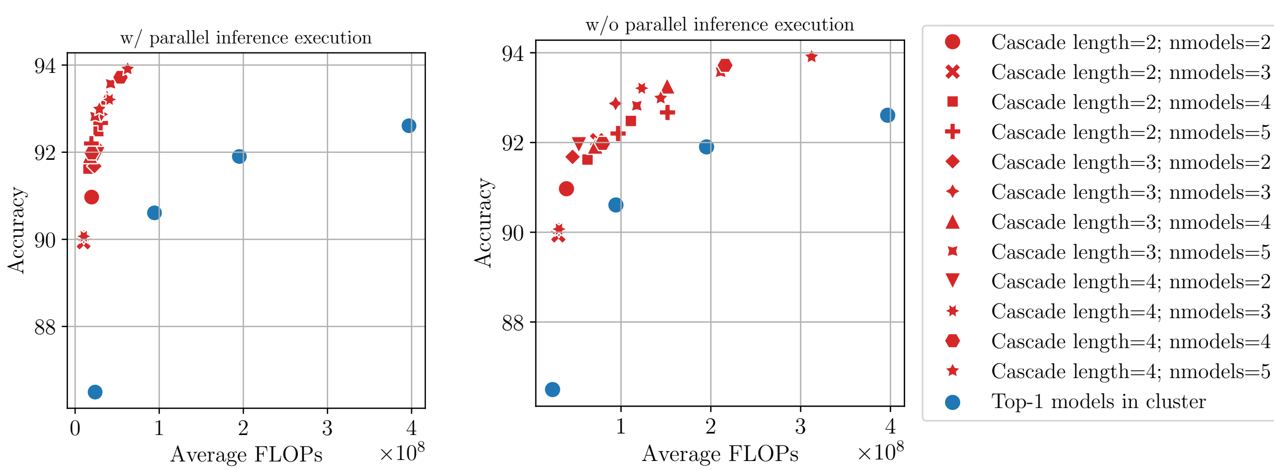


**Figure 8:** Inference execution in parallel vs non-parallel on CIFAR-10.