Online Cascade Learning for Efficient Inference over Streams

Lunyiu Nie[1] **Zhimin Ding**[2] **Erdong Hu**[2] **Christopher Jermaine**[2] **Swarat Chaudhuri**[1]

Abstract

Large Language Models (LLMs) have a natural role in answering complex queries about data streams, but the high computational cost of LLM inference makes them infeasible in many such tasks. We propose *online cascade learning* as an approach to address this challenge. The objective here is to learn a "cascade" of models, starting with lower-capacity models (such as logistic regression) and ending with a powerful LLM, along with a *deferral policy* that determines the model to be used on a given input. We formulate the task of learning cascades online as an imitation-learning problem, where smaller models are updated over time imitating the LLM expert demonstrations, and give a no-regret algorithm for the problem. Experimental results across four benchmarks show that our method parallels LLMs in accuracy while cutting down inference costs by as much as 90% with strong robustness against input distribution shifts, underscoring its efficacy and adaptability in stream processing.

1. Introduction

Large language models (LLMs) (Bommasani et al., 2021; Touvron et al., 2023b; Brown et al., 2020) hold great promise as a means of one-pass query answering over text streams. For example, suppose we have a stream of movie reviews posted on the internet and would like to retrieve the ones that are positive (Figure 1). To address this without human annotations, we can create a prompt for the LLM to identify the sentiment in a review. The LLM can then use this prompt to process each statement in sequence.

However, LLM inference can be extremely expensive, especially in a streaming setup where queries arrive continuously. Based on our benchmarking (Appendix B.1), a GPU server with eight A100 GPUs takes 3.6 seconds to process a document containing 8,192 tokens using the largest Llama (Touvron et al., 2023a). To process one million such documents per hour would thus require 1,000 A100 servers. Using Amazon Web Services, this computation would cost more than 30,000 USD per hour, if it were even possible to procure the machines required.

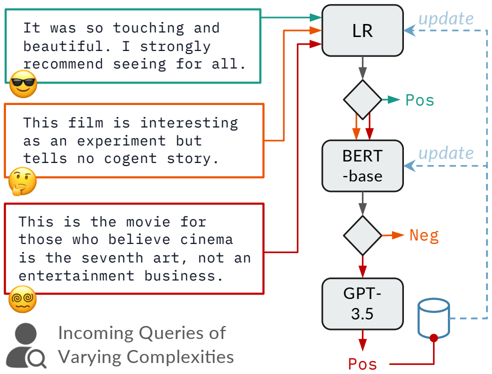
It was so touching and beautiful. I strongly recommend seeing for all.
This film is interesting as an experiment but tells no cogent story.
This is the movie for those who believe cinema is the seventh art, not an entertainment business.
Incoming Queries of Varying Complexities
LR
update
Pos
BERT -base
update
Neg
GPT-3.5
Pos

Figure 1: A sentiment analysis task over a stream of IMDB movie reviews (Maas et al., 2011). We use the cheapest logistic regression model (green lines) to process simpler queries and defer more complex queries to the larger models (orange & red lines). When the cascade proceeds to the LLM, the annotations are collected to update the smaller models online (blue lines).

There are two popular ways to reduce the cost of LLM inference. The first is to *distill* a large language model into a smaller model that can process a document using less computation (Hinton et al., 2015; Gu et al., 2023; Hsieh et al., 2023). The second is to use a *cascade* of models that use smaller models to process "easier" inputs and reserve the largest models for the most difficult inputs (Varshney & Baral, 2022; Chen et al., 2023). However, these existing proposals assume a model of learning in which a labeled training set is available beforehand, making them unsuitable for a streaming setting. In contrast, the streaming setting requires *online learning* (Hoi et al., 2021).

In this paper, we propose *online cascade learning* to bridge this gap in the literature. Our goal here is to develop a cascade of models, arrayed from the least to the most complex, that can process queries in a data stream with optimal cost-performance trade-offs. The critical difference from prior

[1]The University of Texas at Austin [2]Rice University. Correspondence to: Lunyiu Nie <lynie@utexas.edu>, Swarat Chaudhuri <swarat@cs.utexas.edu>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

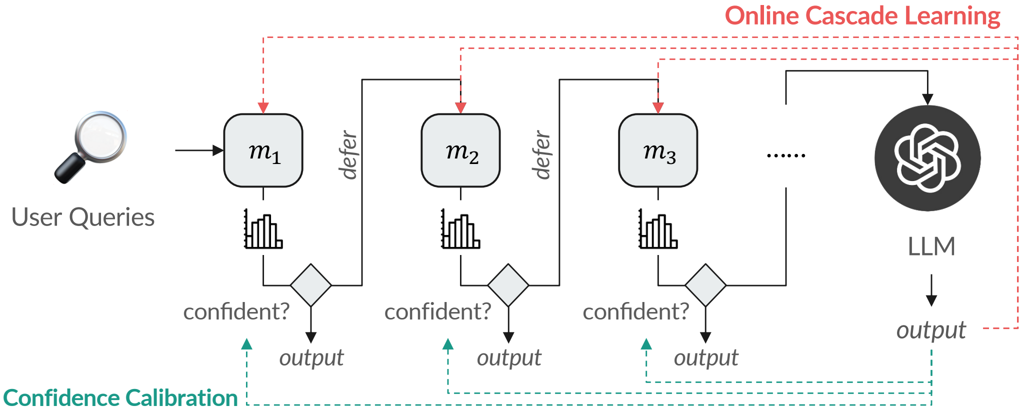

Figure 2: The proposed online cascade learning framework, where smaller models with monotonically increasing capacities and costs ($c_1 < c_2 < ... < c_N$) can progressively learn from the ongoing outputs of an LLM (as denoted in red arrows). Meanwhile, the deferral policy and corresponding confidence scores are also calibrated online (in green arrows).

work is that our cascades are trained in a fully online manner and do not require any human-labeled training data. The smaller models in the cascade would continuously evolve and improve over time by imitating the LLM's demonstrations on the harder queries, enabling them to handle an expanding range of queries with increasing proficiency.

A key component of our cascades is the *deferral policy* that decides, given an input, the best "level" of the cascade that should handle the input (Figure 2). At startup, the policy keeps its "gates" open, allowing all initial inputs to flow through the cascade and be processed by the most expensive model (an LLM). These processed inputs then become training labels for updating the smaller models and the deferral policy within the cascade. Over time, as the model sees more data, the system stabilizes at a state where the smaller, less expensive models can handle the majority of the new inputs. The framework also incorporates a set of learning hyperparameters that adjust the trade-off between accuracy and cost based on user needs.

We formalize the problem of online cascade learning in terms of an episodic Markov decision process (MDP) that considers both prediction loss and computational costs for co-optimization. We assume an expert policy — a high-capacity LLM — for this MDP. We learn the various components of the cascade through imitation learning (Ross et al., 2011) based on the LLM demonstrations. We show that our algorithm comes with a theoretical no-regret guarantee. Our experimental results, on four tasks of various complexity, show that our proposed method can achieve accuracy comparable to LLM at a vastly reduced inference cost.

To summarize the main contributions of our work:

- We introduce *online cascade learning*, a new framework for learning model cascades in resource-intensive streaming analytics settings. The framework enables systematic trade-offs between prediction accuracy and resource usage, and allows learning without any human annotations.
- We offer a formulation of the online learning of cascades in terms of episodic MDPs and give a no-regret imitation learning method for solving this problem.
- We present rigorous experiments showing that our proposed algorithm can achieve comparable accuracy as LLMs while saving up to 90% of the inference costs. Our source code is available at https://github.com/flitternie/online_cascade_learning.

2. Problem Formulation

Inference over Streams as an MDP Problem. We consider stream processing scenarios that have as input a fixed infinite stream $X = \langle x_1, \ldots, x_t, \ldots \rangle$ of user queries. The t-th query x_t is associated with a ground-truth label $y_t \in Y$, where Y is a label set. Our goal is to predict the label for each x_t using an N-level model cascade.

We formulate our problem using an episodic Markov decision process (MDP) $(S, A, \mathcal{T}, C)$. Here:

- S is a set of states. A state in the t-th episode is either a pair $\langle x_t, i \rangle$, where $i \in \{1, ..., N\}$ indicates the current cascade level, or a special terminal state `exit` that ends the episode. The initial state of the t-th episode is $\langle x_t, 1 \rangle$. For clarity, we abbreviate $\langle x_t, i \rangle$ by $s_{t,i}$.
- A is a set of actions, consisting of:
  - The label set Y, representing the potential predictions if the cascade chooses to output at the current state. For instance, in a binary classification task, $Y = \{0, 1\}$.
  - A special action `defer` that activates the next level of the cascade.

- $\mathcal{T}(s_{t,i}, a)$ is a deterministic transition function, consisting of transitions of the form:
  - $\mathcal{T}(s_{t,i}, a) = \texttt{exit}$ for $a \in Y$.
  - $\mathcal{T}(s_{t,i}, \texttt{defer}) = s_{t,i+1}$.
- $C(s_{t,i}, a)$ is a cost function defined as:

$$C(s_{t,i}, a) = \begin{cases} \mathcal{L}(a|y_t) & \text{if } a \in Y, \\ \mu c_{i+1} & \text{if } a = \texttt{defer}. \end{cases}$$

  Here, $\mathcal{L}(a|y_t)$ is a *prediction loss* that measures the accuracy of the cascade's prediction. c_{i+1} represents the penalty we pay for a deferral — intuitively, this penalty captures the computational overheads of going one level deeper into the cascade. The adjustable constant μ guides the trade-off between computational cost and accuracy.

Online Cascade Learning. We now formulate online cascade learning as the problem of solving the above episodic MDP. Let a *policy* π be a stochastic map from states to actions. We use $\pi(s_{t,i}, \texttt{defer})$ to represent the probability that π chooses to defer in state $s_{t,i}$, and $\pi(s_{t,i}, y)$ to represent the probability that π chooses $y \in Y$, conditioned on no deferral having occurred.

The probability of a policy π entering state $s_{t,i}$ in episode t is denoted as $p_\pi^{s_{t,i}}$. For $i > 1$, $s_{t,i}$ can be reached if and only if the policy chooses the $\texttt{defer}$ action in all preceding states $s_{t,1}, ..., s_{t,i-1}$ within the current episode. Thus,

$$p_\pi^{s_{t,1}} = 1, \qquad p_\pi^{s_{t,i}} = \prod_{j=1}^{i-1} \pi(s_{t,j}, \texttt{defer}).$$

Then, the cost of executing π over T episodes is computed by summing over all the episodes and cascade levels:

$$J(\pi, T) = \sum_{t=1}^{T} \left[\sum_{i=1}^{N} p_\pi^{s_{t,i}} C_\pi(s_{t,i}) \right] \tag{1}$$

Here, $C_\pi(s_{t,i})$ is the expected, immediate cost of applying policy π at state $s_{t,i}$. This is computed as:

$$\begin{aligned} C_\pi(s_{t,i}) = \pi(s_{t,i}, \texttt{defer}) \cdot \mu c_{i+1} \\ + (1 - \pi(s_{t,i}, \texttt{defer})) \cdot \sum_{y \in Y} \pi(s_{t,i}, y) \cdot \mathcal{L}(y|y_t). \end{aligned}$$

After having seen the first T queries in the input stream X, our learning goal is to find a policy that minimizes $J(\pi, T)$. When T is clear from the context, we often abbreviate $J(\pi, T)$ by $J(\pi)$.

Policy Representations. We represent policies in a factorized way using a set of *classification models* $\langle m_1, \ldots, m_N \rangle$ that constitute the different levels of the cascade, and a set of *deferral functions* $\langle f_1, \ldots, f_{N-1} \rangle$ that decide whether the current level can perform a high-confidence classification or to defer. We assume each m_i to produce a vector of probabilities, with one probability for each label, and each f_i to produce a probability of deferral. Then the overall policy has the form:

$$\begin{aligned} \pi(s_{t,i}, \texttt{defer}) = f_i(m_i(x_t)), \\ \pi(s_{t,i}, y) = (m_i(x_t))\,[y] \text{ for } y \in Y. \end{aligned}$$

We assume the two parameterized function representations for each level i: the classification model m_i and the deferral function f_i, are both characterized by a crucial property: they guarantee uniform computational costs for evaluating the functions, regardless of the specific function parameters and function inputs. This means that for any instantiation of the parameters in m_i and f_i, the inference costs remain constant, irrespective of the specific input query. This assumption is reasonable in the practical scenarios we target. For example, the inference cost of a BERT-base model is approximately the same, no matter how it is parameterized. This uniform cost assumption also underpins the use of fixed cost penalties c_i in our MDPs.

3. Learning Algorithm

To learn cascades online without human annotations, we propose an imitation learning algorithm that assumes an *expert policy* (an LLM) that can demonstrate ground-truth labels. The algorithm iteratively updates the classification models and deferral functions in the cascade by imitating the expert as in DAgger (Ross et al., 2011). However, unlike traditional imitation learning, our goal here is to balance computational efficiency and accuracy.

We incorporate such an expert into our cascades by assuming that the *final* classification model m_N is the expert LLM. When invoked on a query x_t, m_N always outputs a vector whose largest value is associated with ground-truth label y_t. However, it may not always be the case that choosing to invoke m_N leads to optimal cost. There may be a smaller model m_i for $i < N$ that also produces the correct label y_t, without the cost of the additional $\texttt{defer}$ actions. Indeed, an optimal policy may occasionally incur prediction errors to manage the cost of incurring $\texttt{defer}$ actions. Designing an algorithm that can train a policy in an online fashion to manage those trade-offs is at the core of the paper.

Our overall algorithm is detailed in Algorithm 1. Here, $m_1, ..., m_N$ are the classification models (with m_N being the expert for imitation) and $f_1, ..., f_{N-1}$ are the deferral functions. The smaller models $m_1, ..., m_{N-1}$ are initialized either randomly or with their respective pretrained weights. For each incoming query x_t (outer loop), we sequentially utilize the i-th model in the cascade to make a prediction (inner

Algorithm 1 Online Cascade Learning.

```
Initialize models $m_1, ..., m_N$, deferral functions $f_1, ..., f_{N-1}, \beta_1, \mathcal{D} \leftarrow \emptyset$
for $x_t$ in stream $X$ do
    for $m_i$ in $m_1$ to $m_N$ do
        At probability $\beta_t$:
            Jump to $m_N$ // like DAgger
        $pred_i = m_i(x_t)$ // probability vector
        $action_i \sim f_i(pred_i)$ // whether to defer
        if $m_i$ is $m_N$ or $action_i \neq$ defer then
            $\hat{y}_t = \arg\max(pred_i)$ // discretization
            $\mathcal{D} \leftarrow \mathcal{D} \cup \{x_t, \hat{y}_t\}$ if $m_i$ is $m_N$
            Break
        end if
    end for
    Output $\hat{y}_t$
    Update $m_1$ to $m_{N-1}$ on $\mathcal{D}$ via OGD // imitating expert
    Compute loss $J(\pi, t)$ and update $f_1$ to $f_{N-1}$ via OGD
    Decay $\beta_{t+1}$
end for
```

loop). Specifically, the model generates a probability vector *pred*$_i$, which is processed by the deferral function f_i. This yields a deferral probability score that can be discretized into a binary *action*$_i$ to determine whether the prediction at the current cascade level is reliable.

If the action is `defer`, the query would be navigated to the cascade's next $(i+1)$-th model. Otherwise, it will make a prediction $\hat{y}_t = \arg\max(pred_i)$, and break the inner loop (succeeding models will not be activated for current query x_t). If the query has been deferred to the LLM expert m_N at the last cascade level, its annotation $\hat{y}_t$ is regarded as the ground truth y_t and aggregated to the dataset $\mathcal{D}$.

Throughout the inner iteration, at cascade level i, it may optionally skip the rest layers and jump to the LLM expert m_N at a non-zero decaying probability β_i to directly obtain its demonstration and aggregate it to dataset $\mathcal{D}$, similar to DAgger (Ross et al., 2011), for faster convergence.

After processing each query, the algorithm outputs the prediction y_t, then updates the small models $m_1, ..., m_{N-1}$ to mimic the expert demonstrations on the collected trajectories $\mathcal{D}$. Similarly, the deferral functions $f_1, ..., f_{N-1}$ are also updated based on the loss computed by Equation (1). The algorithm continuously collects annotations from the LLM expert (*e.g.*, at a decaying probability β_t or when the query is deferred to m_N) and updates the policy via online gradient descent (OGD). Practically, the user can change the cost weighting factor μ in the loss function $J(\pi)$ and the initial decaying factor β_1 for adjusting cost budgets.

Theoretical Analysis. In online learning, a policy's *regret* over time T is its total cost minus the cost of the best fixed policy in hindsight. In our setting, the regret of a learned policy π is:

$$\gamma = J(\pi, T) - \min_{\pi \in \Pi} J(\pi, T) \tag{2}$$

$$= \sum_{t=1}^{T} \sum_{i=1}^{N} p_{\pi}^{s_{t,i}} C_{\pi}(s_{t,i}) - \min_{\pi \in \Pi} \sum_{t=1}^{T} \sum_{i=1}^{N} p_{\pi}^{s_{t,i}} C_{\pi}(s_{t,i}), \tag{3}$$

where Π denotes the whole possible policy space. An algorithm is defined to have *no-regret* if it can produce a sequence of policies $\pi_1, \ldots, \pi_T$ such that the expected average regret γ/T goes to 0 as $T \to \infty$ (Ross et al., 2011).

To aid our no-regret analysis of online cascade learning, we start by constructing a simplified *online ensemble learning* algorithm under the same stream processing setting that comprises the linear combination of a series of classification models $m_1, ..., m_N$, each with a static operating probability $\sum_{i=1}^{N} w_i = 1$, without any deferral functions. Let us denote the model parameters of m_i at time t by m_i^t. Assuming a convex, differentiable cost function c^t for all t that can evaluate m_i^t, and bounded, closed, nonempty model spaces $||M_i||$ for all m_i, we analyze the regret of this algorithm.

Theorem 3.1. *With online gradient descent and a learning rate $\eta_t = t^{-1/2}$, the total regret γ of the **online ensemble learning** algorithm is bounded as follows:*

$$\gamma = \sum_{t=1}^{T} \sum_{i=1}^{N} w_i \cdot c^t(m_i^t) - \min_{m_i \in M^i} \sum_{t=1}^{T} \sum_{i=1}^{N} w_i \cdot c^t(m_i^t)$$

$$\leq \frac{||M||^2 \sqrt{T}}{2} + (\sqrt{T} - \frac{1}{2}) ||\nabla c||^2. \tag{4}$$

Therefore, $\lim_{T \to \infty} \gamma/T \leq 0$.

The proof of Theorem 3.1 can be constructed as an extension of Theorem 1 in Zinkevich (2003). Here, $||M||$ denotes the maximum distance within model spaces, and $||\nabla c||$ represents the largest gradient magnitude of the cost function across models. We defer its proof to Appendix A.

Theorem 3.2. *For **online cascade learning** with $\eta_t = t^{-1/2}$, the algorithm's total regret is $o(T)$, implying $\lim_{T \to \infty} \gamma/T \leq 0$, i.e., the average regret approaches zero as T grows ∞.*

Having established the no-regret property for online ensemble learning, we extend this to our online cascade learning algorithm. By replacing the static probabilities w_i in Equation (4) with dynamic probabilities based on preceding model actions, and demonstrating convergence of these dynamic probabilities to optimal values as $T \to \infty$ (proof in Appendix A, Lemma A.2), we mirror the total regret of online cascade learning to that of the online ensemble learning algorithm, completing the proof for Theorem 3.2. Further details are in Appendix A.

Confidence Calibration. A reliable deferral rule is crucial for a cascade system to determine whether to invoke the next model (*i.e.*, to defer) or to output the predictions. Currently, most existing works make deferral decisions based on a confidence score, typically measured either by the maximum predictive probability across all classes (Wang et al., 2022; Varshney & Baral, 2022), or the Shannon entropy of the predictive distribution (Stogiannidis et al., 2023).

However, for online cascade learning, where the model capabilities are dynamically updated, and the annotations from LLMs may be noisy, confidence-based deferral rules have been shown to be inadequate (Jitkrittum et al., 2023). To aid the learning of deferral functions (*i.e.*, $f_1, ..., f_{N-1}$), we adopt a post-hoc approach to calibrate the confidence estimate of a certain model's prediction $m_i(x_t)$. It is implemented using a multi-layer perceptron (MLP) that takes the corresponding model's predictive probabilities as input, and the functions can be updated with the following objective:

$$\min_{\pi'_i:\mathbb{R}^{|Y|}\to\mathbb{R}} \sum_{x_t\in X} \sum_{i=1}^{N-1} L(f_i(m_i(x_t)), z_i), \tag{5}$$

where $z_i = \mathbb{1}[\arg\max m_i(x_t) \neq y_t^*]$ (*i.e.*, $z_i = 1$ if model m_i's prediction is not equal to the annotation y_t^*, otherwise $z_i = 0$) and L is the mean-squared error loss function.

Since we treat the expert LLM predictions (*i.e.*, $\arg\max(m_N(x_t))$) as the ground truth labels y_t, calibration is only performed on those input queries where the expert LLM is invoked. During the calibration, the learning of f_i would only optimize the parameters of the MLP, not the models m_i in the cascade. During inference time, the post-hoc deferral function would choose to `defer` if $f_i(m_i(x_t)) > 0.5$. Otherwise, it would output the prediction of the current model.

4. Experimental Setup

We construct a cascade system using three models: (i) a logistic regression model, (ii) a pretrained BERT-base model with 110M parameters (Kenton & Toutanova, 2019), and (iii) GPT-3.5 Turbo[1]. This diverse set of models allows us to evaluate the effectiveness and robustness of our approach across various complexities and types of queries.

To further test the adaptability and scalability of our system, we also conduct supplementary experiments where (a) a Llama 2 70B Chat (Touvron et al., 2023b) is used as the alternative LLM, and (b) a BERT-large with 340M parameters is incorporated to create a larger cascade. These variations aim to demonstrate the flexibility of our framework in accommodating different cascade sizes and structures.

Benchmarks. We evaluate our approach on four benchmarks that reflect the demands of real-world streaming applications in a wide range of commercial services, from customer feedback analysis to content moderation:

- ***IMDB.*** A binary sentiment classification benchmark with 50,000 movie reviews (Maas et al., 2011). The dataset has an even distribution of `positive` and `negative` samples. We use the official training split for our experiments, which contains 25,000 samples.
- ***HateSpeech.*** A binary classification dataset consisting of posts from an online forum, annotated with `hate` and `noHate` labels (de Gibert et al., 2018). After filtering, the dataset contains 10,703 samples with a pronounced class imbalance (1:7.95 ratio) between hatespeech and non-hatespeech examples. This imbalance mirrors a realistic challenge in streaming data environments, particularly in detecting harmful content. Our evaluation on this benchmark focuses on both *accuracy* and *recall*.
- ***ISEAR.*** A multi-class emotion detection benchmark encompassing 7,666 samples across seven categories (`Joy`, `Fear`, `Anger`, `Sadness`, `Disgust`, `Shame`, `Guilt`) (Shao et al., 2015). Each category is well-represented, providing a balanced label distribution across the dataset.
- ***FEVER.*** A fact-checking dataset with 6,512 claims manually verified against Wikipedia, labeled as `Supported` or `Refuted` (Thorne et al., 2018). It tests our framework's ability to perform complex reasoning and information verification, a crucial aspect for real-time truth assessment in streaming data applications.

Baselines. We compare online cascade learning against several baselines to establish its effectiveness:

- ***LLMs in the Cascade:*** This includes GPT-3.5 Turbo and Llama 2 70B Chat with zero-shot task prompting (details in Appendix B.2). The LLM outputs are also used as the annotations for online cascade learning and distillation.
- ***Knowledge Distillation:*** We fine-tune smaller models using different portions of LLM annotations. To ensure fairness, datasets are split equally, with 50% prepared for training (as distillation labels) and the remaining 50% for testing. All methods are evaluated on the identical test sets. In our experiments, the distilled smaller models are used in isolation without any ensemble or cascade.
- ***Online Ensemble Learning:*** We employ all available models in an ensemble with learned predetermined probabilities. The smaller models are also continuously updated based on LLM annotations. This serves as an ablation of our method by excluding the deferral policy learning component.

[1] https://platform.openai.com/docs/models/gpt-3-5

	IMDB			**HateSpeech** (Accuracy | Recall)			**ISEAR**			**FEVER**		
	$\mathcal{N}$=1300	$\mathcal{N}$=3800	$\mathcal{N}$=5200	$\mathcal{N}$=600	$\mathcal{N}$=2700	$\mathcal{N}$=4900	$\mathcal{N}$=1200	$\mathcal{N}$=1500	$\mathcal{N}$=2700	$\mathcal{N}$=700	$\mathcal{N}$=2000	$\mathcal{N}$=2800
GPT-3.5 Turbo	94.15			83.34 | 83.28			70.34			79.98		
Distilled LR	82.61	83.60	87.01	80.18 | 37.94	82.23 | 49.25	**85.03** | 45.59	44.97	47.46	48.92	56.51	57.80	57.13
Distilled BERT-base	85.28	90.18	90.19	80.49 | 64.39	80.71 | 73.88	79.35 | 77.37	**61.49**	62.62	63.37	61.70	63.64	70.82
Online Ensemble Learning	86.73	88.80	89.95	82.61 | 76.75	77.48 | 76.89	81.55 | 80.30	56.56	60.42	61.78	61.69	69.78	76.67
Online Cascade Learning	**87.95**	**92.48**	**93.01**	**82.66** | **82.36**	**85.35** | **77.20**	83.26 | **81.03**	60.78	**65.34**	**69.75**	**61.95**	**71.86**	**78.49**
Llama 2 70B Chat	93.33			77.81 | 82.19			68.23			77.15		
Distilled LR	82.17	85.80	86.88	67.94 | 66.56	**79.71** | 61.73	**81.46** | 49.91	46.78	47.56	51.76	57.46	61.24	58.42
Distilled BERT-base	85.39	85.59	85.44	75.84 | **78.87**	79.18 | 75.54	80.27 | 72.21	**62.18**	61.84	65.12	**65.88**	65.66	67.54
Online Ensemble Learning	87.14	88.66	89.61	75.99 | 60.36	70.79 | **79.16**	76.82 | 81.84	54.74	57.35	60.19	63.48	71.27	76.46
Online Cascade Learning	**87.58**	**92.14**	**92.63**	**78.30** | 63.06	78.32 | 76.54	78.32 | **82.03**	59.24	**63.34**	**67.25**	63.81	**72.47**	**77.73**

Table 1: Comparison of accuracy (and recall for HateSpeech dataset) among different methods under various cost budgets. The upper part of the table uses GPT-3.5 Turbo as the LLM in the cascade, while the lower part employs Llama 2 70B Chat. To ensure fairness, the same annotation cost budgets (*i.e.*, the maximum allowable LLM calls, denoted as $\mathcal{N}$, controlled via adjusting the cost weighting factor μ and decaying factor β for online cascade learning) are applied across all methods.

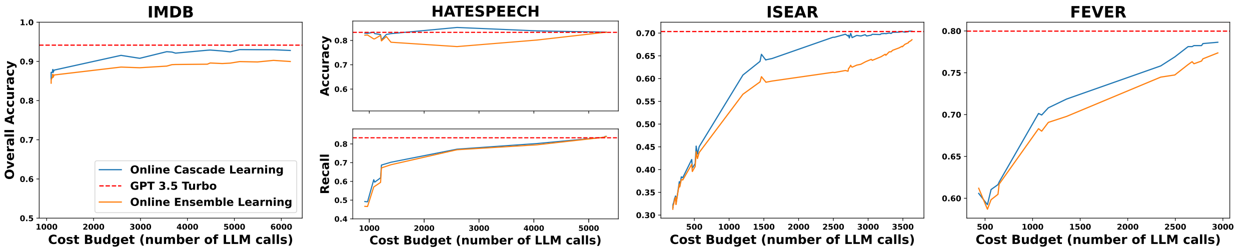

Figure 3: Accuracy curve (and Recall curve for HateSpeech) with respect to costs, using GPT-3.5 Turbo as the LLM in a cascade that also comprises logistic regression and BERT-base.

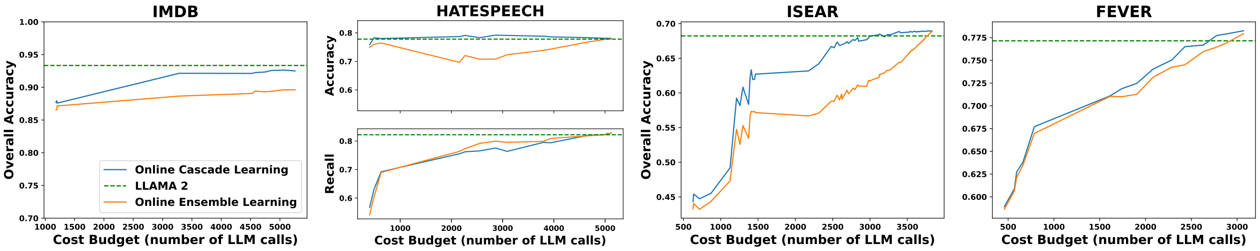

Figure 4: Accuracy curve (and Recall curve for HateSpeech) with respect to costs, using Llama 2 70B Chat as the LLM in a cascade that also comprises logistic regression and BERT-base.

5. Experimental Results

5.1. Overall Performance and Cost Trade-offs

IMDB. The results presented in Table 1, Figure 3 and 4 demonstrate that our proposed online cascade learning system can consistently achieve higher accuracies compared to knowledge distillation and online ensemble learning baselines on the IMDB dataset across different cost budgets, regardless of whether using GPT-3.5 Turbo or Llama 2 70B Chat in the cascade. Notably, Figure 4 highlights our system's ability to closely rival the performance of Llama 2 70B Chat while achieving a 60% reduction in inference costs (*i.e.*, calling LLM ∼5200 times in processing a total of 12500 queries). This effectively demonstrates the system's efficiency in balancing cost with performance.

HateSpeech. The results on the HateSpeech dataset further reveal the strengths of our online cascade learning system in handling datasets with significant class imbalance. Most models may face a trade-off between accuracy and recall due to the imbalanced nature of HateSpeech. However, the accuracy-cost and recall-cost trade-offs, respectively depicted in the upper and lower subplots of Figure 3 and 4, demonstrate that our system effectively improves recall with minimal impact on accuracy as the cost budget increases. Although the recall rate of online cascade learning is marginally lower than the baselines under certain budgets, it can achieve a better balance between recall and precision, as evidenced by its consistently higher F1 scores in

Appendix Figure 10. In particular, as demonstrated in Table 1 where Llama 2 70B Chat is the LLM and the cost budget $\mathcal{N} = 4900$, our system even outperforms the LLM with a similar recall (ours: 82.03% vs. LLM: 82.19%) and a better accuracy (78.32% vs. 77.81%), underlining its effectiveness in handling imbalanced data streams.

ISEAR. On the ISEAR benchmark, our online cascade learning system also effectively balances cost and accuracy in complex multi-class classification. As indicated in Figure 3 and 4, the system's performance gradually aligns with that of GPT-3.5 Turbo and even surpasses Llama 2 70B Chat as the cost budget increases. This success underscores the advantages of smaller models in adapting to complex classifications by learning from the LLM annotations, enabling them to potentially outshine zero-shot LLMs. Moreover, the notable performance gap between online ensemble learning and online cascade learning also confirms the benefit of co-optimizing model learning with deferral policy learning for optimal cost-performance equilibrium.

FEVER. FEVER is a significantly more complex dataset compared to the previous benchmarks. It demands models to reason over the statements and validate their factuality based on parametric knowledge. Therefore, small models of limited capacities, such as logistic regression, struggle to perform effectively on FEVER even after several iterations of update, as evident in Table 1 where distilled LR can perform only slightly better than random guess (*i.e.*, 50%). Recognizing these limitations, our online cascade learning system smartly adapts by prioritizing more capable models, such as BERT-base and the LLM, for processing most of the queries, leading to a favorable accuracy-cost trade-off. Remarkably, when using Llama 2 70B Chat as the LLM with a cost budget of $\mathcal{N} = 2800$, our system slightly outperforms the LLM in accuracy (77.73% vs. 77.15%), showcasing the system's proficiency at navigating intricate reasoning tasks with enhanced cost-efficiency.

5.2. Case Analysis

To examine our approach's online learning process more closely, we run the online cascade learning at specific cost budgets and conduct a detailed case analysis.

IMDB. Figure 5 illustrates the online cascade system performance at a specific cost budget ($\mathcal{N} = 3671$) throughout the inference of the IMDB dataset. Initially, for the first 160 samples, all queries are processed exclusively by the LLM, as indicated by the stacked plot in the background. However, with the arrival of more queries over time, both the logistic regression and BERT-base models, denoted by the green and orange dashed lines, dynamically improve by learning from the GPT-3.5 Turbo annotations and increasingly contribute to query processing. When the incoming number of samples approaches 4000, most queries are processed by BERT-base, leaving only 30% of the queries deferred to GPT-3.5 Turbo. Meanwhile, the system can achieve an overall accuracy consistently close to or even slightly higher than the LLM (marked by the blue dashed line), which affirms the effectiveness of online cascade learning in saving inference costs with minimum performance degradation.

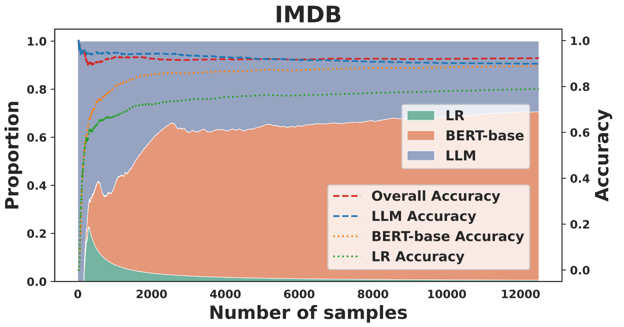

Figure 5: Inference results on IMDB when $\mathcal{N} = 3671$. Online cascade learning system performs similarly to GPT-3.5 Turbo while saving ∼70% of the inference costs.

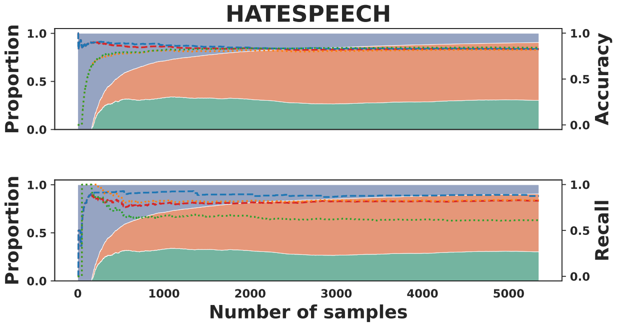

Figure 6: Inference results on HateSpeech when $\mathcal{N} = 507$. Online cascade learning system performs similarly to GPT-3.5 Turbo while saving ∼90% of the inference costs.

HateSpeech. Similarly, on HateSpeech dataset, when $\mathcal{N} = 507$, as shown in Figure 6, as the number of samples approaches 5000, 30.31% of the queries are handled by logistic regression, while 60.24% are handled by BERT-base, which altogether cut down the LLM inference costs by more than 90%. At the same time, the online cascade system can still achieve an overall accuracy of 82.66% and recall of 82.36%, which aligns closely with the performance of GPT-3.5 Turbo (Accuracy: 83.34%, Recall: 83.28%)

ISEAR. Our analysis of the ISEAR dataset under a specific cost budget ($\mathcal{N} = 2517$) also confirms the dynamic adaptability and cost-saving features of our online cascade system. As the number of processed samples increases, the system demonstrates an impressive ability to gradually shift query processing from the expensive GPT-3.5 Turbo to the more economical models. This transition is evident in the

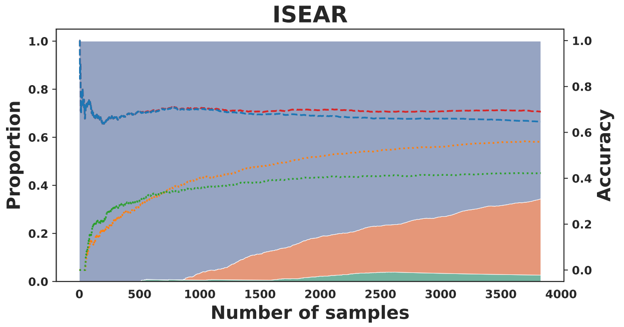

Figure 7: Inference results on ISEAR when $\mathcal{N} = 2517$. Online cascade learning system performs very close to GPT-3.5 Turbo while saving ∼30% of the inference costs.

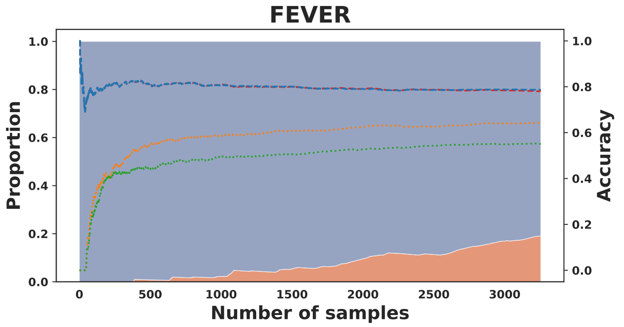

Figure 8: Inference results on FEVER when $\mathcal{N} = 2635$. Online cascade learning system performs similarly to GPT-3.5 Turbo while saving ∼20% of the inference costs.

increasing proportions of queries handled by BERT-base over time, as shown in Figure 7. Most importantly, since the proportion of queries handled by BERT-base is not yet converged, the whole system can further cut inference costs while sustaining high accuracy as more samples come in.

FEVER. On the FEVER dataset, our system's performance at a cost budget ($\mathcal{N} = 3671$) further validates its usefulness even under complex reasoning task settings. The system's learning curve, as depicted in Figure 8, illustrates a steady increase in the number of queries processed by the BERT-base models. Given logistic regression's limited capability in fact-checking, our online cascade learning system smartly shifts its reliance towards more capable models, such as BERT-base and the larger LLMs. The overall performance in terms of accuracy aligns closely with that of the GPT-3.5 Turbo, achieving significant cost savings (17%) without compromising on the quality of the results.

5.3. Adaptability to Larger Cascade

To further validate the adaptability of our online cascade learning system, we have also explored scaling up the cascade system by further integrating a BERT-large model (4 models in total). The results of this integration are encouraging, detailed in Appendix Figure 11. The expanded cascade system, particularly when trained on Llama 2 70B Chat annotations, demonstrates equivalent or even superior results compared to the standalone LLM in most cases, with the exception of the HateSpeech dataset, a simpler task setting where a larger cascade might complicate the deferral policy learning and thus degrading overall performance. Therefore, it is important to align cascade size with task complexity to optimize performance and avoid overfitting. A more detailed analysis is in Appendix C.2

The success of this larger cascade system signals the huge potential in our work's extensions. It showcases our system's inherent flexibility and scalability, enabling it to accommodate and efficiently utilize more powerful models as part of its architecture. Future investigations on online cascade systems may further scale up the system by incorporating LLMs of different sizes and specialties, allowing for refined tailoring of responses to the specific characteristics of different queries. However, there are also several factors worth noting when scaling up the cascade, as outlined in our analysis in Appendix C.1 and C.3.

5.4. Robustness against Distributional Shifts

Distribution Shift in Input Length. Longer inputs typically involve more complex semantics. For example, on IMDB, the average accuracy of GPT-3.5 Turbo is notably lower on longer movie reviews, as evidenced in Appendix Table 5. Therefore, to verify our method's robustness against input distribution shifts, we rearrange the IMDB benchmark with length ascending order to simulate a distribution shift over the inputs' semantic complexity. The results are shown in Figure 9 and Table 2. Despite the distribution shift in input length, online cascade learning demonstrates good performance with minimal accuracy drops across different cost budgets, regardless of the LLM adopted in the cascade.

Distribution Shift in Input Category. We further validate our approach with a distribution shift in input semantic categories, by filtering all the input samples regarding "`Comedy`" movies in the IMDB dataset (8,140 out of 25,000), and feeding them into the cascade as the last part of the input stream. This means that the system had not seen any comedy movie reviews in the first 2/3 of the inputs before processing comedy movie reviews in the last 1/3 of the inputs. As shown in Figure 9 and Table 2, our approach also performs reliably under the distribution shift in input categories, with a slight increase in average accuracy.

Based on the results, we conclude that our proposed approach can well utilize the advantages of online learning, quickly adapt to unseen inputs, and perform robustly against distribution shifts in the data streams.

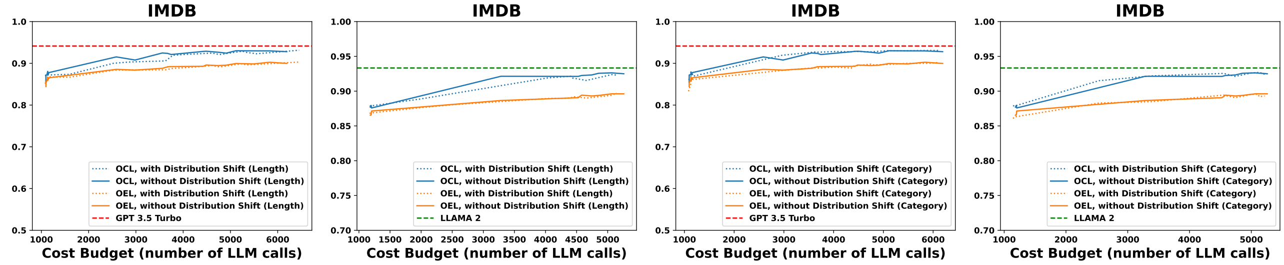

Figure 9: Cost-accuracy trade-off curves on two input distributional shift scenarios, respectively using GPT-3.5 Turbo and Llama 2 70B Chat as the LLM with logistic regression and BERT-base in a cascade. “OCL” refers to online cascade learning, and “OEL” means online ensemble learning.

	GPT-3.5 Turbo	**Llama 2 70B**
Without Any Distribution Shift	90.77%	90.97%
With Distribution Shift in Input Length ⇒ Difference	90.23% -0.54%	90.64% -0.33%
With Distribution Shift in Input Category ⇒ Difference	90.85% +0.08%	91.46% +0.49%

Table 2: Average accuracy of our approach across different cost budgets with distribution shifts in input length or input category, compared to the default setting.

6. Related Work

6.1. Knowledge Distillation

Knowledge distillation, originally conceptualized by Hinton et al. (2015), emerged as a technique to transfer knowledge from a large, complex model (teacher) to a smaller, more efficient one (student), intending to retain performance while reducing computational costs. Notable advancements include the works of Sanh et al. (2019), who demonstrated the effectiveness of distilling the capabilities of BERT into smaller models, and Gu et al. (2023) who successfully applied distillation to LLMs, achieving comparable performance with significantly reduced model sizes. However, the significant difference in capabilities between the teacher and student model can lead to challenges, particularly when dealing with complex queries that require advanced reasoning or involve intricate subject matter (Cho & Hariharan, 2019; Zhang et al., 2022), thereby highlighting a performance gap that distillation alone cannot overcome (Rawat et al., 2021).

6.2. Boosting LLMs with Small Models

Recent advancements in combining LLMs with smaller models have suggested promising avenues for improving model performance and efficiency. For classification tasks, SuperICL improves LLMs’ performance by utilizing the predictions of smaller models to enrich the prompting context (Xu et al., 2023). For generative tasks, speculative decoding techniques (Leviathan et al., 2023; Miao et al., 2023; Liu et al., 2023) are proposed to accelerate LLM inference by using smaller language models to predict token sequences. Our work differs from these works in that small models are separately trained as plugins, and the performance of LLMs is prioritized. Instead, we focus on the overall performance of the entire cascade.

6.3. Model Cascade

By orchestrating multiple models with varying complexities, model cascade has been widely adopted in both CV and NLP tasks to enhance system efficiency. An early theme in this field is the early exiting from a neural network’s intermediate layers (Liu et al., 2020; Xin et al., 2020; Schwartz et al., 2020). These works have later inspired the cascade of complete models (Li et al., 2020; Khalili et al., 2022). Among them, Varshney & Baral (2022) systematically examines the trade-off between accuracy and cost in a cascade of variants of BERT. Chen et al. (2023) further extends the cascade to cover multiple LLM APIs by incorporating a scoring function. Compared to the previous works, our work expands the model cascade by allowing small models to learn online and improve rather than having fixed capabilities, similar to the recent idea of “neural caching” (Ramírez et al., 2023; Stogiannidis et al., 2023). Moreover, by formulating the model confidence as part of the learning objective, our work eliminates the need to set confidence thresholds manually.

7. Conclusion

In this work, we address the challenge of managing streaming queries with LLMs in a cost-efficient way. We propose an online cascade learning framework that adapts to evolving queries by improving smaller models in the cascade through imitation of LLM behaviors. Our theoretical analysis provides a no-regret performance guarantee for the algorithm. Extensive experiments confirmed the effectiveness of our approach, showing that it can achieve performance levels comparable to LLMs while significantly reducing inference costs, with potential savings of up to 90%.

Acknowledgements

We thank the anonymous reviewers for their valuable comments and suggestions in enhancing the paper. This research was supported by the NSF under grant numbers CCF-1918651, 2008240, 2131294, and 2212557, ARO award #W911NF-21-1-0009, DARPA award #HR00112320018, NIH CTSA award No. UL1TR003167, and US DOT Tier-1 UTC CYBER-CARE grant #69A3552348332.

Impact Statement

This work presents an online cascade learning framework for enhancing the efficiency of LLMs in handling streaming queries. While primarily advancing the field of machine learning, it also has broader societal implications. Notably, by significantly reducing computational demands (up to 90% in inference costs), our framework addresses environmental concerns related to the energy consumption of large-scale computing. This reduction in resource utilization aligns with global efforts towards sustainable technology use.

Furthermore, our work has the potential to democratize access to advanced machine learning technologies. By lowering operational costs, smaller organizations and researchers with limited resources can leverage the power of LLMs, fostering inclusivity and diversity in research and application development. Additionally, our approach could contribute to sectors like healthcare, finance, and public policy, where real-time, cost-effective data processing is crucial.

References

Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M. S., Bohg, J., Bosselut, A., Brunskill, E., et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901, 2020.

Chen, L., Zaharia, M., and Zou, J. Frugalgpt: How to use large language models while reducing cost and improving performance. *CoRR*, abs/2305.05176, 2023. doi: 10.48550/arXiv.2305.05176. URL https://doi.org/10.48550/arXiv.2305.05176.

Cho, J. H. and Hariharan, B. On the efficacy of knowledge distillation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 4794–4802, 2019.

de Gibert, O., Perez, N., García-Pablos, A., and Cuadros, M. Hate Speech Dataset from a White Supremacy Forum. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pp. 11–20, Brussels, Belgium, October 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-5102. URL https://www.aclweb.org/anthology/W18-5102.

Gu, Y., Dong, L., Wei, F., and Huang, M. Knowledge distillation of large language models. *arXiv preprint arXiv:2306.08543*, 2023.

Gupta, N., Narasimhan, H., Jitkrittum, W., Rawat, A. S., Menon, A. K., and Kumar, S. Language model cascades: Token-level uncertainty and beyond. *arXiv preprint arXiv:2404.10136*, 2024.

Hinton, G., Vinyals, O., and Dean, J. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Hoi, S. C., Sahoo, D., Lu, J., and Zhao, P. Online learning: A comprehensive survey. *Neurocomputing*, 459:249–289, 2021.

Hsieh, C.-Y., Li, C.-L., Yeh, C.-K., Nakhost, H., Fujii, Y., Ratner, A., Krishna, R., Lee, C.-Y., and Pfister, T. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. *arXiv preprint arXiv:2305.02301*, 2023.

Jitkrittum, W., Gupta, N., Menon, A. K., Narasimhan, H., Rawat, A. S., and Kumar, S. When does confidence-based cascade deferral suffice? *CoRR*, abs/2307.02764, 2023. doi: 10.48550/arXiv.2307.02764. URL https://doi.org/10.48550/arXiv.2307.02764.

Kenton, J. D. M.-W. C. and Toutanova, L. K. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pp. 4171–4186, 2019.

Khalili, L., You, Y., and Bohannon, J. Babybear: Cheap inference triage for expensive language models. *arXiv preprint arXiv:2205.11747*, 2022.

Leviathan, Y., Kalman, M., and Matias, Y. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning*, pp. 19274–19286. PMLR, 2023.

Li, L., Lin, Y., Chen, D., Ren, S., Li, P., Zhou, J., and Sun, X. Cascadebert: Accelerating inference of pre-trained language models via calibrated complete models cascade. *arXiv preprint arXiv:2012.14682*, 2020.

Li, X. and Orabona, F. On the convergence of stochastic gradient descent with adaptive stepsizes. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 983–992. PMLR, 2019.

Liu, W., Zhou, P., Wang, Z., Zhao, Z., Deng, H., and Ju, Q. Fastbert: a self-distilling bert with adaptive inference time. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 6035–6044, 2020.

Liu, X., Hu, L., Bailis, P., Stoica, I., Deng, Z., Cheung, A., and Zhang, H. Online speculative decoding. *arXiv preprint arXiv:2310.07177*, 2023.

Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., and Potts, C. Learning word vectors for sentiment analysis. In Lin, D., Matsumoto, Y., and Mihalcea, R. (eds.), *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA*, pp. 142–150. The Association for Computer Linguistics, 2011. URL https://aclanthology.org/P11-1015/.

Miao, X., Oliaro, G., Zhang, Z., Cheng, X., Wang, Z., Wong, R. Y. Y., Chen, Z., Arfeen, D., Abhyankar, R., and Jia, Z. Specinfer: Accelerating generative llm serving with speculative inference and token tree verification. *arXiv preprint arXiv:2305.09781*, 2023.

Ramírez, G., Lindemann, M., Birch, A., and Titov, I. Cache & distil: Optimising api calls to large language models. *arXiv preprint arXiv:2310.13561*, 2023.

Rawat, A. S., Zaheer, M., Menon, A. K., Ahmed, A., and Kumar, S. When in doubt, summon the titans: Efficient inference with large models. *arXiv preprint arXiv:2110.10305*, 2021.

Ross, S., Gordon, G., and Bagnell, D. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 627–635. JMLR Workshop and Conference Proceedings, 2011.

Sanh, V., Debut, L., Chaumond, J., and Wolf, T. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

Schwartz, R., Stanovsky, G., Swayamdipta, S., Dodge, J., and Smith, N. A. The right tool for the job: Matching model and instance complexities. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 6640–6651, 2020.

Shao, B., Doucet, L., and Caruso, D. R. Universality versus cultural specificity of three emotion domains: Some evidence based on the cascading model of emotional intelligence. *Journal of Cross-Cultural Psychology*, 46(2): 229–251, 2015.

Stogiannidis, I., Vassos, S., Malakasiotis, P., and Androutsopoulos, I. Cache me if you can: an online cost-aware teacher-student framework to reduce the calls to large language models. *arXiv preprint arXiv:2310.13395*, 2023.

Thorne, J., Vlachos, A., Christodoulopoulos, C., and Mittal, A. Fever: a large-scale dataset for fact extraction and verification. *arXiv preprint arXiv:1803.05355*, 2018.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023a.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023b.

Varshney, N. and Baral, C. Model cascading: Towards jointly improving efficiency and accuracy of NLP systems. In Goldberg, Y., Kozareva, Z., and Zhang, Y. (eds.), *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pp. 11007–11021. Association for Computational Linguistics, 2022. doi: 10.18653/v1/2022.emnlp-main.756. URL https://doi.org/10.18653/v1/2022.emnlp-main.756.

Wang, X., Kondratyuk, D., Christiansen, E., Kitani, K. M., Movshovitz-Attias, Y., and Eban, E. Wisdom of committees: An overlooked approach to faster and more accurate models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=MvO2t0vbs4-.

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

Xu, C., Xu, Y., Wang, S., Liu, Y., Zhu, C., and McAuley, J. Small models are valuable plug-ins for large language models. *arXiv preprint arXiv:2305.08848*, 2023.

Zhang, C., Yang, Y., Liu, J., Wang, J., Wu, W., Wang, B., and Song, D. Lifting the curse of capacity gap in distilling large language models. 2022.

Zinkevich, M. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pp. 928–936, 2003.

A. Detailed Proofs and Theoretical Analysis

This appendix section provides detailed proofs and explanations for the theoretical analysis presented in the main paper Section 3.

Preliminaries

To ensure clarity, we begin by defining the concept of regret in online learning.

Definition A.1. Given an online learning algorithm operating over time steps $1, ..., T$, the regret γ is defined as the difference between the total loss incurred by the algorithm and the loss of the best fixed policy in hindsight. For online cascade learning, regret γ is formally expressed as:

$$\begin{aligned}\gamma &= J(\pi, T) - \min_{\pi \in \Pi} J(\pi, T) && (6)\\ &= \sum_{t=1}^{T}\sum_{i=1}^{N} p_{\pi}^{s_{t,i}} C_{\pi}(s_{t,i}) - \min_{\pi \in \Pi} \sum_{t=1}^{T}\sum_{i=1}^{N} p_{\pi}^{s_{t,i}} C_{\pi}(s_{t,i}) && (7)\end{aligned}$$

As the value of t is clear from the context, we abbreviate states $s_{t,i} = \langle x_t, i \rangle$ by s_i. We denote the best fixed policy as $\pi^* = \arg\min_{\pi \in \Pi} J(\pi, T)$ and $\pi(s_{t,i}, \texttt{defer})$ as $p(\pi, s_i)'$. Furthermore, since the prediction loss is computed as the aggregation over a class probability distribution, we simplify $\sum_{y \in Y} \pi(s_{t,i}, y) \cdot \mathcal{L}(y|y_t)$ as $\mathcal{L}(a_i|y_t)$ by using a_i to represent the output probability vector, which leads to the following representation of regret:

$$\begin{aligned}\gamma &= \sum_{t=1}^{T}\sum_{i=1}^{N} p_{\pi}^{s_i} C_{\pi}(s_i) - \sum_{t=1}^{T}\sum_{i=1}^{N} p_{\pi^*}^{s_i} C_{\pi^*}(s_i) && (8)\\ &= \sum_{t=1}^{T}\sum_{i=1}^{N}\prod_{j=1}^{i-1} p(\pi, s_j)' \cdot \Big((1 - p(\pi, s_i)') \cdot \mathcal{L}(a_i|y_t) + p(\pi, s_i)' \cdot \mu c_{i+1}\Big)\\ &\quad - \sum_{t=1}^{T}\sum_{i=1}^{N}\prod_{j=1}^{i-1} p(\pi^*, s_j)' \cdot \Big((1 - p(\pi^*, s_i)') \cdot \mathcal{L}(a_i^*|y_t) + p(\pi^*, s_i)' \cdot \mu c_{i+1}\Big). && (9)\end{aligned}$$

Online Ensemble Learning Analysis

To contextualize the no-regret analysis of our proposed online cascade learning algorithm, we first analyze a simplified *online ensemble learning* algorithm under the same stream processing setting that comprises the linear combination of a series of classification models $m_1, ..., m_N$, each with a static operating probability $\sum_{i=1}^{N} w_i = 1$. Let us denote the model parameters of m_i at time t by m_i^t. Assuming a convex, differentiable cost function c^t for all t that can evaluate m_i^t, and bounded, closed, nonempty model spaces $||M_i||$ for all m_i, we define $||x|| = \sqrt{x \cdot x}$ and $d(x, y) = ||x - y||$ to establish:

$$\begin{aligned}||M|| &= \max_{x,y \in M_i, i \in [1, N-1]} d(x, y)\\ ||\nabla c|| &= \max_{m_i \in M_i, i \in [1, N-1], t \in [1, T]} ||\nabla c^t(m_i)||.\end{aligned}$$

Theorem 3.1. *With online gradient descent and a learning rate $\eta_t = t^{-1/2}$, the total regret γ of the* ***online ensemble learning*** *algorithm is bounded as follows:*

$$\begin{aligned}\gamma &= \sum_{t=1}^{T}\sum_{i=1}^{N} w_i \cdot c^t(m_i^t) - \min_{m_i \in M^i} \sum_{t=1}^{T}\sum_{i=1}^{N} w_i \cdot c^t(m_i^t)\\ &\le \frac{||M||^2\sqrt{T}}{2} + (\sqrt{T} - \frac{1}{2})||\nabla c||^2. && (4)\end{aligned}$$

Therefore, $\lim_{T \to \infty} \gamma/T \le 0$.

Proof. First, we show that, without loss of generality, for all t there exists a $g_i^t \in \mathbb{R}^n$ such that for all models $m_i \in M_i$, $c^t(m_i) = g_i^t \cdot m_i$.

By defining $g_i^t = \nabla c^t(m_i)$, because $c^t(m_i)$ is convex, for all $m_i \in M_i$:

$$c^t(m_i) \geq (\nabla c^t(m_i^t)) \cdot (m_i - m_i^t) + c^t(m_i^t). \tag{10}$$

Set m_i^* to be the best-fixed model in hindsight. Then, because $m_i^* \in M_i : c^t(m_i^*) \geq g^t \cdot (m_i^* - m^t) + c^t(m_i^t)$. Thus,

$$\begin{aligned} c^t(m_i^t) - c^t(m_i^*) &\leq c^t(m_i^t) - \Big(g_i^t \cdot (m_i^* - m_i^t) + c^t(m_i^t)\Big) \qquad (11)\\ &\leq g_i^t m_i^t - g_i^t m_i^* \qquad (12) \end{aligned}$$

Thus we show that for all m_i, $g_i^t m_i^t - g_i^t m_i^*$ is the upper bound of $c^t(m_i^t) - c^t(m_i^*)$:

$$\begin{aligned} \gamma &= \sum_{t=1}^{T}\sum_{i=1}^{N} w_i \cdot c^t(m_i^t) - \min_{m_i \in M^i} \sum_{t=1}^{T}\sum_{i=1}^{N} w_i \cdot c^t(m_i^t) \qquad (13)\\ &= \sum_{t=1}^{T}\sum_{i=1}^{N} w_i \Big(c^t(m_i^t) - c^t(m_i^*)\Big) \qquad (14)\\ &\leq \sum_{t=1}^{T}\sum_{i=1}^{N} w_i \Big(g^t m_i^t - g^t m_i^*\Big). \qquad (15) \end{aligned}$$

We define for all t, ${\hat{m}_i}^{t+1} = m_i^t - \eta_t g_i^t$ (gradient descent). Note that $m_i^{t+1} = P({\hat{m}_i}^{t+1}) = \arg\min_{m_i \in M_i} ||m_i - {\hat{m}_i}^{t+1}||$ (greedy projection). We will attempt to bound the regret of not playing action m_i^* on round t:

$$\begin{aligned} {\hat{m}_i}^{t+1} - m_i^* &= m_i^t - \eta_t g_i^t - m_i^* \qquad (16)\\ ({\hat{m}_i}^{t+1} - m_i^*)^2 &= (m_i^t - m_i^*)^2 - 2\eta_t (m_i^t - m_i^*) \cdot g_i^t + \eta_t^2 ||g_i^t||^2 \qquad (17) \end{aligned}$$

Since by definition, for all $\hat{m}_i \in \mathbb{R}^n$, for all $m_i \in M_i$, $(\hat{m}_i - m_i)^2 \geq (P(\hat{m}_i) - m_i)^2$. Also, $||g_i^t|| \leq ||\nabla c||$. So

$$\begin{aligned} (m_i^{t+1} - m_i^*)^2 &\leq ({\hat{m}_i}^{t+1} - m_i^*)^2 \qquad (18)\\ &\leq (m_i^t - m_i^*)^2 - 2\eta_t (m_i^t - m_i^*) \cdot g_i^t + \eta_t^2 ||\nabla c||^2, \qquad (19) \end{aligned}$$

which can be converted into:

$$g_i^t m_i^t - g_i^t m_i^* \leq \frac{1}{2\eta_t}\Big[(m_i^t - m_i^*)^2 - (m_i^{t+1} - m_i^*)^2\Big] + \frac{\eta_t}{2}||\nabla c||^2. \tag{20}$$

By summing Equation (15) and Equation (20) we get:

$$\begin{aligned} \gamma &\leq \sum_{t=1}^{T}\sum_{i=1}^{N} w_i \Big(g^t m_i^t - g^t m_i^*) \qquad (21)\\ &\leq \sum_{t=1}^{T}\sum_{i=1}^{N} w_i \Bigg[\frac{1}{2\eta_t}\Big[(m_i^t - m_i^*)^2 - (m_i^{t+1} - m_i^*)^2\Big] + \frac{\eta_t}{2}||\nabla c||^2\Bigg] \qquad (22)\\ &\leq \sum_{i=1}^{N} \frac{w_i}{2\eta_t}(m_i^1 - m_i^*)^2 - \sum_{i=1}^{N} \frac{w_i}{2\eta_t}(m_i^{T+1} - m_i^*)^2 \qquad (23)\\ &+ \frac{1}{2}\sum_{t=2}^{T}\sum_{i=1}^{N} w_i (\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}})(m_i^t - m_i^*)^2 + \frac{||\nabla c||^2}{2}\sum_{t=1}^{T}\sum_{i=1}^{N} w_i \cdot \eta_t \qquad (24)\\ &\leq \sum_{i=1}^{N} w_i ||M||^2 \Big(\frac{1}{2\eta_1} + \frac{1}{2}\sum_{t=2}^{T}(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}})\Big) + \frac{||\nabla c||^2}{2}\sum_{t=1}^{T}\sum_{i=1}^{N} w_i \cdot \eta_t \qquad (25) \end{aligned}$$

Since $\sum_{i=1}^{N} w_i = 1$,

$$\gamma \leq ||M||^2 \Big(\frac{1}{2\eta_1} + \frac{1}{2}\sum_{t=2}^{T}(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}})\Big) + \frac{||\nabla c||^2}{2}\sum_{t=1}^{T}\eta_t \tag{26}$$

$$\leq ||M||^2 \frac{1}{2\eta_T} + \frac{||\nabla c||^2}{2}\sum_{t=1}^{T}\eta_t \tag{27}$$

Now, if we define $\eta_t = \frac{1}{\sqrt{t}}$, then

$$\sum_{t=1}^{T}\eta_t = \sum_{t=1}^{T}\frac{1}{\sqrt{t}} \tag{28}$$

$$\leq 1 + \int_{t=1}^{T}\frac{dt}{\sqrt{t}} \tag{29}$$

$$\leq 1 + [2\sqrt{t}]_1^T \tag{30}$$

$$\leq 2\sqrt{T} - 1 \tag{31}$$

Plugging this into Equation (27) yields

$$\gamma \leq \frac{||M||^2\sqrt{T}}{2} + \frac{||\nabla c||^2}{2}(2\sqrt{T} - 1) \tag{32}$$

$$\leq \frac{||M||^2\sqrt{T}}{2} + (\sqrt{T} - \frac{1}{2})||\nabla c||^2 \tag{33}$$

Therefore, when the number of iterations T approaches infinity, the average regret $\lim_{T\to\infty} \gamma/T \leq 0$. □

Extension to Online Cascade Learning

The above theorem establishes that the online ensemble learning algorithm achieves no regret as $T \to \infty$. Transitioning to online cascade learning, we extend this analysis by substituting the fixed model probabilities w_i in Equation (4) with dynamic probabilities influenced by preceding model actions (*i.e.*, $p_\pi^{s_i} = \prod_{j=1}^{i-1} p(\pi, s_j)'$) in Equation (9). Thus, we now analyze the convergence of the deferral policies $p_\pi^{s_i}$.

Lemma A.2. *For online cascade learning, for all $i \in [1, N]$, $p_\pi^{s_i} - p_{\pi^*}^{s_i} \xrightarrow{T\to\infty} 0$.*

Proof. We begin by revisiting Equation (9), transforming its first term to facilitate our analysis:

$$\sum_{t=1}^{T}\sum_{i=1}^{N}\prod_{j=1}^{i-1} p(\pi, s_j)' \cdot \Big((1 - p(\pi, s_i)') \cdot \mathcal{L}(a_i|y_t) + p(\pi, s_i)' \cdot \mu c_{i+1}\Big) \tag{34}$$

$$= \sum_{t=1}^{T}\sum_{i=1}^{N}\prod_{j=1}^{i-1} p(\pi, s_j)' \cdot \Big(p(\pi, s_i)'\big(\mu c_{i+1} - \mathcal{L}(a_i|y_t)) + \mathcal{L}(a_i|y_t)\Big). \tag{35}$$

Given that $p(\pi, s_i)'$ lies within the range $(0, 1)$ for all t and i, the coefficient preceding $\mathcal{L}(a_i|y_t)$, namely $\prod_{j=1}^{i-1} p(\pi, s_j)'(1 - p(\pi, s_i)')$, also falls within $(0, 1)$. Assuming $\mathcal{L}$ is convex, and following the convergence arguments in Li & Orabona (2019), we establish that as $T \to \infty$, $\mathcal{L}(a_i|y_t)$ converges to a minimal loss value ϵ_i for each model.

Integrating this convergence into our transformed regret expression of Equation (35), we arrive at:

$$\sum_{t=1}^{T}\sum_{i=1}^{N}\prod_{j=1}^{i-1} p(\pi, s_j)' \cdot \Big(p(\pi, s_i)'\big(\mu c_{i+1} - \epsilon_i) + \epsilon_i\Big). \tag{36}$$

As discussed in Jitkrittum et al. (2023) Proposition 3.1, to minimize the aggregated costs, the optimal deferral rule $p(\pi^*, s_i)'$ should be:

$$p(\pi^*, s_i)' = \begin{cases} 0 & \text{if } \mu c_{i+1} - \epsilon_i > 0, \\ 1 & \text{otherwise.} \end{cases}$$

With gradient descent, as $T \to \infty$, we can show that for all i, $p(\pi, s_i)'$ gradually approaches $p(\pi^*, s_i)' = \mathbb{1}[\mu c_{i+1} - \epsilon_i \leq 0]$ by pointwise convergence. Therefore, in the base case of $N = 2$, we have $p_\pi^{s_1} - p_{\pi^*}^{s_1} = p(\pi, s_1)' - p(\pi^*, s_1)' = 0, p_\pi^{s_2} - p_{\pi^*}^{s_2} = 1 - 1 = 0$.

Assuming the lemma holds for $N = k$, *i.e.*, $p_\pi^{s_i} - p_{\pi^*}^{s_i} = 0$ for all $i \in [1, k]$, we extend this to $N = k + 1$:

$$\begin{aligned} & p(\pi, s_{k+1})' - p(\pi^*, s_{k+1})' \qquad (37) \\ &= \prod_{j=1}^{k} p(\pi, s_j)' - \prod_{j=1}^{k} p(\pi^*, s_j)' \qquad (38) \\ &= p_\pi^{s_k} \cdot p(\pi, s_k)' - p_{\pi^*}^{s_k} \cdot p(\pi^*, s_k)' \qquad (39) \\ &= p_{\pi^*}^{s_k} \Big(p(\pi, s_k)' - p(\pi^*, s_k)' \Big) = 0 \qquad (40) \end{aligned}$$

Hence by mathematical induction, for online cascade learning, when $T \to \infty$, for all $i \in [1, N]$, $p_\pi^{s_i} - p_{\pi^*}^{s_i} = 0$. □

With the conclusions of Theorem 3.1 and Lemma A.2, we can now analyze the performance of our proposed online cascade learning algorithm.

Theorem 3.2. *For* ***online cascade learning*** *with* $\eta_t = t^{-1/2}$*, the algorithm's total regret is* $o(T)$*, implying* $\lim_{T\to\infty} \gamma/T \leq 0$*, i.e., the average regret approaches zero as* T *grows* ∞*.*

Proof. Leveraging the findings of Lemma A.2, assuming $\mathcal{L}$ is a convex function and $T \to \infty$, we can now reformulate the regret expression in Equation (8) as:

$$\begin{aligned} \gamma &= \sum_{t=1}^{T} \sum_{i=1}^{N} p_\pi^{s_i} C_\pi(s_i) - \sum_{t=1}^{T} \sum_{i=1}^{N} p_{\pi^*}^{s_i} C_{\pi^*}(s_i) \qquad (41) \\ &= \sum_{t=1}^{T} \sum_{i=1}^{N} p_{\pi^*}^{s_i} C_\pi(s_i) - \sum_{t=1}^{T} \sum_{i=1}^{N} p_{\pi^*}^{s_i} C_{\pi^*}(s_i) \qquad (42) \end{aligned}$$

The transformation of the regret expression to Equation (42) is significant as it bridges our understanding of the regret in online cascade learning with the established results from online ensemble learning. Specifically, by treating $p_{\pi^*}^{s_i}$ as analogous to the fixed model probabilities w_i and $C_\pi(s_i)$ as equivalent to the costs $c^t(m_i^t)$ in the ensemble learning context, we establish a parallel between the two regret formulations.

Therefore, following the conclusion of Theorem 3.1, we can infer that, with a learning rate $\eta_t = t^{-1/2}$, the online cascade learning algorithm is guaranteed can achieve no regret, *i.e.*, $\lim_{T\to\infty} \gamma/T \leq 0$.

□

B. Detailed Experimental Setups

B.1. A Simple Prefill Experiment

To determine time required to process a document-plus-prompt in a realistic scenario, we performed the following, simple experiment. Using the 65B parameter LLaMA model (Touvron et al., 2023a) and PyTorch 2.1.2, on an Amazon Web Services '`m6in.16xlarge`' machine with eight, A100 GPUs, we prepare a sequence of 10 prompts each consisting of 8192 tokens, and perform "first token" inference on each in sequence. That is, we process the prompt and obtain the first output token, and record the total time taken, which is 36.2 seconds, for an average of 3.6 seconds per prompt.

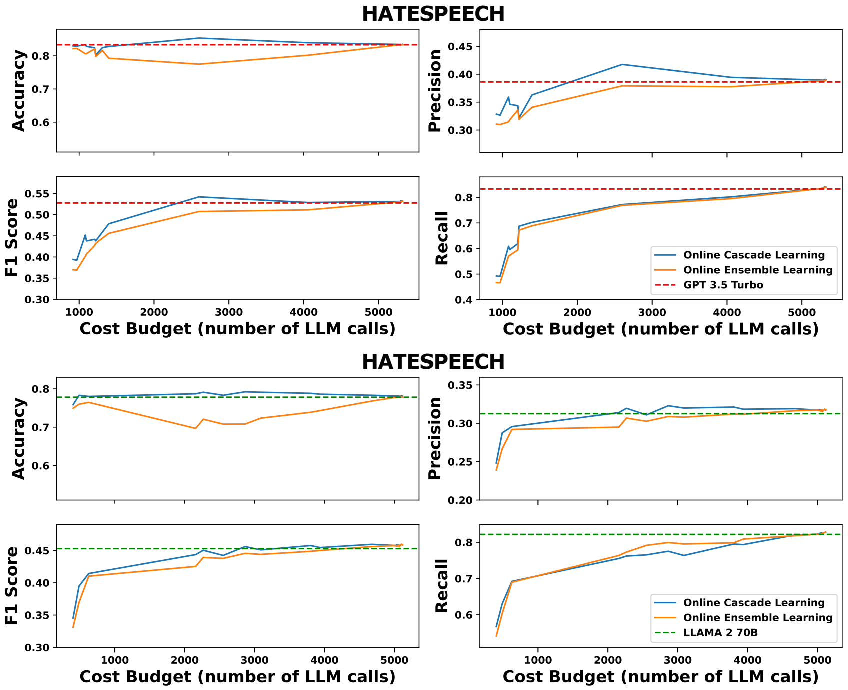

Figure 10: A more detailed cost-performance trade-off plot with accuracy, F1-score, recall, and precision curves, respectively using GPT-3.5 Turbo and Llama 2 70B Chat as the LLM.

The reason we perform first-token inference is that relative to the first output token—which requires a quadratic all-to-all attention computation—subsequent tokens are relatively costless, taking a fraction of a second. For our application, where the full response sequence is expected to be short, first token inference is by far the most costly step. Note that in our experiment, inference is performed separately on each prompt, rather than as a batch. This is necessary to avoid out-of-memory errors (there is not enough GPU memory to process more than one prompt at a time, due to the memory requirements of the all-to-all attention computation).

B.2. LLM Prompts

IMDB

System Prompt: You are a helpful, respectful and honest assistant. The user has given you a movie review to help them make their decision. You should read the review and tell the user whether the review overall shows a positive or negative sentiment towards the movie. Return the answer in one word.
User Prompt: Here is the movie review: {REVIEW} \\Tell me whether the above review overall shows a positive or negative sentiment towards the movie. Return the answer in one word.

HateSpeech

System Prompt: You are given a post from an online forum and you need to check whether the post contains any hate speech. Return your answer in one word (yes or no) without any explanations.
User Prompt: Post: {POST}

ISEAR

System Prompt: In this task, you will be performing a classification exercise aimed at identifying the underlying emotion conveyed by a given sentence. The emotions to consider are as follows:
Anger: Anger is a strong feeling of displeasure, hostility, or frustration.
Joy: Joy is a positive and uplifting emotion characterized by happiness, elation, and a sense of contentment.
Sadness: Sadness is a feeling of sorrow, unhappiness, or despondency.
Guilt: Guilt is a self-directed emotion that arises from a sense of wrongdoing or moral transgression.
Shame: Shame is a powerful emotion associated with feeling embarrassed, humiliated, or unworthy.

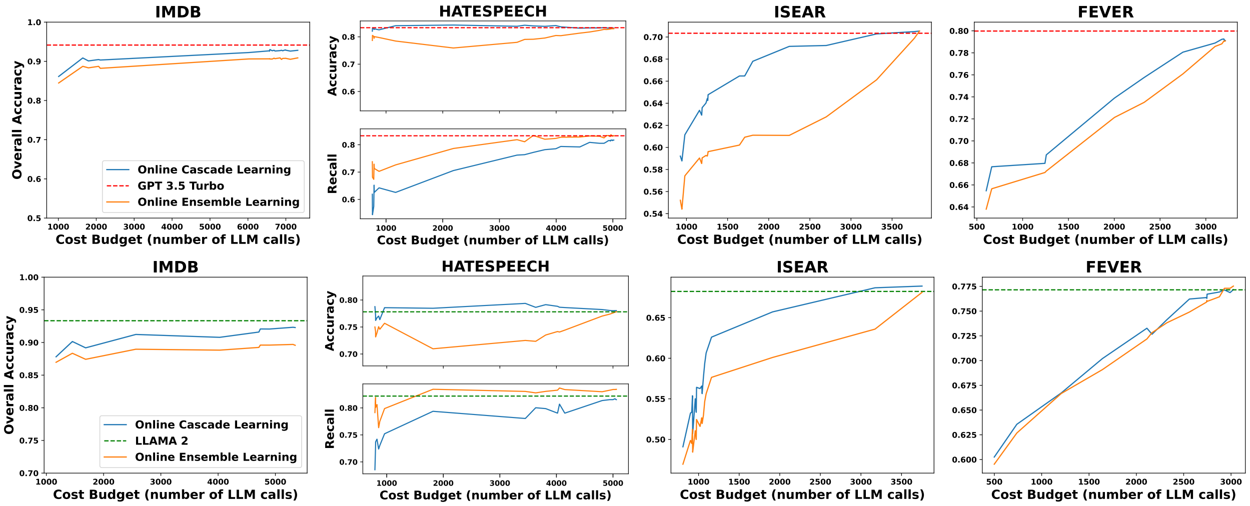

Figure 11: Accuracy curve (and Recall curve for HateSpeech) with respect to costs, respectively using GPT-3.5 Turbo and Llama 2 70B Chat as the LLM in a larger cascade that also comprises logistic regression, BERT-base, and BERT-large.

Fear: Fear is an emotion triggered by a perceived threat or danger.
Disgust: Disgust is an aversive emotion linked to feelings of revulsion, repulsion, or strong distaste. It arises in response to things that are offensive or unpleasant.
Your task is to analyze each sentence provided and categorize it into one of these emotions based on the dominant feeling conveyed by the text.
This classification will require an understanding of the nuances of human emotions and the context in which the sentences are presented. Remember, you have to classify the sentences using only anger, joy, sadness, guilt, shame, fear or disgust. Please respond with only the word and nothing else.
User Prompt: {`SENTENCE`} \\Classify the emotion of this hypothetical sentence. Respond in exactly one word in all lowercase with a response in the exact format requested by the user. Do not acknowledge my request with "sure" or in any other way besides going straight to the answer. Only answer in exactly one word.

FEVER
System Prompt: You are a helpful, respectful and honest assistant. This is a fact-checking task. Use your knowledge to determine whether a given claim is true or false. Answer only in "true" or "false" without providing any explanations.
User Prompt: In June 2017, the following claim was made: {`CLAIM`}.

B.3. Experimental Configurations

The experiments involved querying Llama 2 70B Chat utilized a single machine equipped with 8 NVIDIA A40 GPUs, each with 48GB of memory, running CUDA 12.0. All the other experiments were conducted on a machine with 4 NVIDIA Quadro RTX 8000 GPUs (48GB memory each) on CUDA 12.2.

The detailed hyperparameter settings for online cascade learning are listed in Table 3 and 4. We tuned the hyperparameters using a grid search method on a separate validation set, which is a standard practice to avoid overfitting. In particular, we used the training set prepared for the knowledge distillation (as mentioned in Section 4) as our online cascade learning method's validation set.

Specifically, for the hyperparameters β and μ, we observed that our experimental results are notably robust to variations in β. Regarding μ, we tuned it specifically in the context of different cost budgets, which was essential for plotting Figure 3 and Figure 4 in our paper. By adjusting μ, we were able to effectively manage the cost budgets to evaluate our method's performance on cost-accuracy trade-offs, which is a critical aspect of our research objective.

Note that the "learning rate" in the table refers to the learning rates of the MLPs (in Section 3: Confidence Calibration), not the models'. We used a consistent configuration for both online cascade learning and the distillation baselines by setting BERT-base's batch size to 8, the learning rate to 0.00001, and the number of epochs to 5.

Table 3: Hyperparameter settings for online cascade learning experiments with GPT-3.5 Turbo as the LLM.

	Model Cost	Cache Size	Batch Size	Learning Rate	Decaying Factor	Calibration Factor
IMDB, Small Cascade						
LR	1	8	8	0.0007	0.97	0.4
BERT-base	1182	16	8	0.0007	0.95	0.3
IMDB, Large Cascade						
LR	1	8	8	0.0007	0.99	0.45
BERT-base	3	16	8	0.0007	0.97	0.4
BERT-large	1182	32	16	0.0007	0.95	0.4
HateSpeech, Small Cascade						
LR	1	8	8	0.001	0.97	0.4
BERT-base	1182	16	8	0.0007	0.9	0.4
HateSpeech, Large Cascade						
LR	1	8	8	0.001	0.99	0.45
BERT-base	3	16	8	0.0007	0.97	0.45
BERT-large	1182	32	16	0.0007	0.95	0.45
ISEAR, Small Cascade						
LR	1	8	8	0.0007	0.8	0.15
BERT-base	1182	16	8	0.0007	0.9	0.45
ISEAR, Large Cascade						
LR	1	8	8	0.0007	0.99	0.4
BERT-base	3	16	8	0.0007	0.97	0.35
BERT-large	1182	32	16	0.0007	0.95	0.3
FEVER, Small Cascade						
LR	1	8	8	0.0007	0.97	0.4
BERT-base	1182	16	8	0.0007	0.95	0.3
FEVER, Large Cascade						
LR	1	8	8	0.0007	0.97	0.4
BERT-base	3	16	8	0.001	0.95	0.4
BERT-large	1182	32	16	0.0001	0.93	0.4

C. Discussion

C.1. Analysis of Training & Inference Cost Equilibrium

We conduct a thorough analysis of the training overhead for the models involved in our cascade to quantify the computational costs incurred in our approach. Below are the computation costs for training or inference over one sample (the computational costs of the confidence calibration MLP in Section 3, inference: $897\ Flops$, training: $1794\ Flops$, are negligible):

- Logistic Regression training: $33.8 \times 10^4 Flops$ (floating-point operations).
- Logistic Regression inference: $16.9 \times 10^4 Flops$.
- BERT-base training: $18.5 \times 10^7 Flops$.
- BERT-base inference: $9.2 \times 10^7 Flops$.
- BERT-large training: $55.5 \times 10^7 Flops$.
- BERT-large inference: $27.7 \times 10^7 Flops$.

Comparatively, Llama 2 70B's inference costs for generating one token is approximately $39.86 \times 10^{15} Flops$ (we have no access to GPT-3.5 Turbo's running statistics). Even if all the small models are **consistently updated per sample**, the per-sample training costs of a large cascade $\underset{\textit{LR}}{33.8 \times 10^4} + \underset{\textit{BERT-base}}{18.5 \times 10^7} + \underset{\textit{BERT-large}}{55.5 \times 10^7} \approx 7.4 \times 10^8 Flops$ is still 5.3×10^7 times smaller than the per-sample Llama 2 70B inference costs.

Therefore, the computational costs regarding the deferral policy's inference and training are minimal when comparing against the enormous LLM inference costs. In the real world, as the smaller models' capabilities grow over time and the

Table 4: Hyperparameter settings for online cascade learning experiments with Llama 2 70B Chat as the LLM.

	Model Cost	Cache Size	Batch Size	Learning Rate	Decaying Factor	Calibration Factor
IMDB, Small Cascade						
LR	1	8	8	0.0007	0.97	0.4
BERT-base	636	16	8	0.0007	0.95	0.3
IMDB, Large Cascade						
LR	1	8	8	0.0007	0.99	0.45
BERT-base	3	16	8	0.0007	0.97	0.4
BERT-large	636	32	16	0.0007	0.95	0.4
HateSpeech, Small Cascade						
LR	1	8	8	0.001	0.97	0.4
BERT-base	636	16	8	0.0007	0.9	0.4
HateSpeech, Large Cascade						
LR	1	8	8	0.001	0.99	0.45
BERT-base	3	16	8	0.0007	0.97	0.45
BERT-large	636	32	16	0.0007	0.95	0.45
ISEAR, Small Cascade						
LR	1	8	8	0.0007	0.8	0.15
BERT-base	636	16	8	0.0007	0.9	0.45
ISEAR, Large Cascade						
LR	1	8	8	0.0007	0.99	0.4
BERT-base	3	16	8	0.0007	0.97	0.35
BERT-large	636	32	16	0.0007	0.95	0.3
FEVER, Small Cascade						
LR	1	8	8	0.0007	0.97	0.4
BERT-base	636	16	8	0.0007	0.95	0.3
FEVER, Large Cascade						
LR	1	8	8	0.0007	0.97	0.4
BERT-base	3	16	8	0.001	0.95	0.4
BERT-large	636	32	16	0.0001	0.93	0.4

need for training decreases, the incurred model training overhead is also negligible compared to the tremendous LLM inference costs saved with our approach.

More formally, we can formulate the theoretical cost equilibrium as follows:

$$100\% \cdot C = x \cdot M + (1 - x) \cdot (M + 2M + C),$$

where the LHS refers to the inference overheads for using the LLM to process all queries, and the RHS comprises the maximum inference costs forusing the small models to handle $x\%$ of the queries (which are not deferred to the LLM), and the LLM inference costs, plus the inference & training costs for updating the small models when handling the rest $(1 - x\%)$ queries. This equation can be further simplified to

$$M = \frac{xC}{3 - 2x},$$

where C represents the LLM inference cost, x is the proportion of queries handled by small models, and M indicates small models' aggregated inference costs.

For example, assuming $C = 39.86 \times 10^{15} Flops$, $x = 0.5$, then $M \approx 9.95 \times 10^{15} Flops$,typically refers to a total number of parameters around 17.5 Billion. That means, when using Llama 2 70B as the LLM, even if the smaller models can only handle 50% of the input queries, as long as the smaller models' total number of parameters does not exceed 17.5B, our approach can still save costs.

Length	Count	Average Length	GPT-3.5 Turbo Accuracy
52-664	4975	481.92	95.54%
664-843	5018	745.86	95.08%
843-1160	5003	985.05	93.96%
1160-1852	5001	1453.49	93.74%
1852-13704	5002	2953.95	92.44%
Total	25000	1325.07	94.15%

Table 5: GPT-3.5 Turbo's classification accuracy across different IMDB review lengths. Longer inputs are typically more complex and thus have lower average accuracies.

C.2. Potential Overfitting in Larger Cascades

When scaling up the cascade, the task complexity may influence the appropriate cascade size, as suggested by the performance dynamics observed in Section 5.3. There are several factors that may affect this:

Task Complexity vs. Cascade Size. The HateSpeech dataset, despite its class imbalance, represents arelatively simple binary classification task where hate speech is often identifiable through specific keywords. Our findings, illustrated in Figure 6, show that a basic cascade of logistic regression and BERT-base can already handle 90% of the queries effectively, matching the performance of more complex models like GPT3.5 Turbo. This indicates that for simpler tasks, adding more models to the cascade might introduce unnecessary noise, complicating the deferral policy learning and thus degrading overall performance.

Capability Gap in Models. The performance impact of adding more models to a cascade also depends onthe capability gap between these models. In our larger cascade setup, we incorporated BERT-large alongside BERT-base. However, both models exhibit similar performance on the HateSpeech dataset, meaning the addition of BERT-large brought minimal benefit. This redundancy shows that the effectiveness of a cascade does not solely rely on adding more models, but rather on ensuring that each model contributes unique capabilities to the given task.

Improved Performance on More Complex Dataset. In contrast, for the more complex ISEAR dataset, the larger cascade that includes BERT-large outperformed the smaller cascade, despite the ISEAR dataset's smaller size (3833) compared to HateSpeech (5352). This outcome supports the idea that as task complexity increases, a larger cascade can still better balance the cost-performance trade-off, leveraging the distinct strengths of each model in the cascade.

C.3. Challenges in Scaling Up Cascade

Scaling up the cascade for more complex task processing is technically feasible and compatible with our proposed framework, but can present notable challenges:

Confidence Measurement in Generative Tasks. For complex generative tasks, such as conversation, the measurement of confidence scores is also challenging. For example, traditional confidence estimation methods may not be well-suited for measuring LLMs' token-by-token generation (Gupta et al., 2024). This difficulty in accurately assessing model confidence complicates the deferral policy learning within cascades, potentially leading to suboptimal routing of queries and affecting overall system efficiency.

Constraints with API-Based LLMs. More practically, many current LLMs, especially those accessible only via APIs, do not support fine-tuning. This restriction hinders our ability to tailor each LLM in the cascade to specific tasks in an online learning setting, presenting a significant limitation for customizing LLM-only cascades.

Computational Costs. The most important hurdle in a more complex cascade is the significant computational expense associated with online learning. For example, in an LLM-only cascade, updating LLMs in real-time (even if using efficient training techniques like LoRA) demands substantial resources, making it impractical for many applications. This computational burden not only affects scalability but also limits the frequency and extent to which LLMs can be updated, impacting the cascade's adaptability and performance. As analyzed in Appendix C.1's cost equilibrium, when the learnable model space grows too large, the training costs would offset the saved inference costs, which is against our initial motivation of enhancing cost-efficiency.