ThriftLLM: On Cost-Effective Selection of Large Language Models for Classification Queries

Keke Huang*
kk.huang@ubc.ca
University of British Columbia

Yimin Shi
yiminshi@u.nus.edu
National University of Singapore

Dujian Ding
dujian.ding@gmail.com
University of British Columbia

Yifei Li
yfli@student.ubc.ca
University of British Columbia

Yang Fei
yfei11@u.nus.edu
National University of Singapore

Laks Lakshmanan
laks@cs.ubc.ca
University of British Columbia

Xiaokui Xiao
xkxiao@nus.edu.sg
National University of Singapore
CNRS@CREATE, Singapore

ABSTRACT

Recently, large language models (LLMs) have demonstrated remarkable capabilities in understanding and generating natural language content, attracting widespread attention in both industry and academia. An increasing number of services offer LLMs for various tasks via APIs. Different LLMs demonstrate expertise in different domains of queries (e.g., text classification queries). Meanwhile, LLMs of different scales, complexities, and performance are priced diversely. Driven by this, several researchers are investigating strategies for selecting an ensemble of LLMs, aiming to decrease overall usage costs while enhancing performance. However, to our best knowledge, none of the existing works addresses the problem, how to find an LLM ensemble subject to a cost budget, which maximizes the ensemble performance with guarantees.

In this paper, we formalize the performance of an ensemble of models (LLMs) using the notion of correctness probability, which we formally define. We develop an approach for aggregating responses from multiple LLMs to enhance ensemble performance. Building on this, we formulate the Optimal Ensemble Selection (OES) problem of selecting a set of LLMs subject to a cost budget that maximizes the overall correctness probability. We show that the correctness probability function is non-decreasing and non-submodular and provide evidence that the OES problem is likely to be NP-hard. By leveraging a submodular function that upper bounds correctness probability, we develop an algorithm, ThriftLLM, and prove that it achieves an instance-dependent approximation guarantee with high probability. Our framework functions as a data processing system that selects appropriate LLM operators to deliver high-quality results under budget constraints. It achieves state-of-the-art performance for text classification and entity matching queries on multiple real-world datasets against various baselines in our extensive experimental evaluation, while using a relatively lower cost budget, strongly supporting the effectiveness and superiority of our method.

PVLDB Reference Format:

Keke Huang, Yimin Shi, Dujian Ding, Yifei Li, Yang Fei, Laks Lakshmanan, and Xiaokui Xiao. ThriftLLM: On Cost-Effective Selection of Large Language Models for Classification Queries. PVLDB, 18(11): 4410 - 4423, 2025.
doi:10.14778/3749646.3749702

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/kkhuang81/thriftLLM.

1 INTRODUCTION

The evolution of large language models (LLMs) has significantly transformed the field of natural language processing. These models have enabled AI systems to generate human-like texts based on input instructions with remarkable accuracy. An increasing number of companies (e.g., OpenAI, Anthropic, and Google) have introduced a wide range of services powered by LLMs, such as GPT-4 [37], Claude-3 [1], and Gemini [21]. These services, accessible via APIs, have achieved notable success across various applications, such as text classification, question answering, summarization, and natural language inference [31, 44, 45, 57]. In data management, various LLMs have been adopted for text-to-SQL generation [16, 20, 56], query performance optimization [50, 51, 61], schema matching [40], and entity matching [32, 39, 41]. In those applications, we can regard each LLM as a standalone operator that takes unstructured data as input and computes derived attributes (e.g., sentiment, stance). From this perspective, an LLM ensemble acts as a data processing system to deliver high-quality results under budget constraints. More broadly, this strategy can be seen as extending the capabilities of traditional DBMSs by leveraging the power of LLM operators over unstructured data. This is particularly relevant for modern

*Work partially done when the author was with NUS.

Proceedings of the VLDB Endowment, Vol. 18, No. 11 ISSN 2150-8097.
doi:10.14778/3749646.3749702

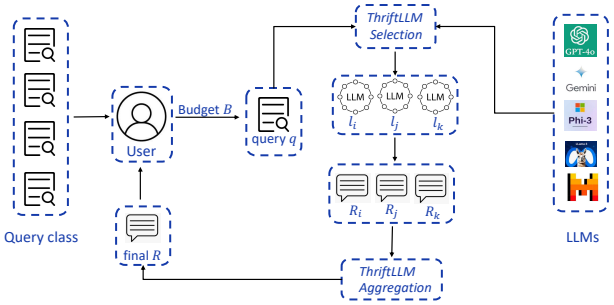


Figure 1: Overview of ThriftLLM: R_i, R_j, R denote responses.

data management workloads, where unstructured data, e.g., text, images, increasingly plays a central role.

While LLMs offer significant improvements in performance, they come with substantial costs, particularly for high-throughput applications. For example, GPT-4 charges $30 per million input tokens and $60 per million output tokens [23], while Gemini-1.5 [34] costs $7 and $21 per million input and output tokens, respectively. Typically, models with higher performance charge a higher price. However, it is well recognized that expensive models with larger number of parameters do not necessarily dominate the cheaper models across all applications [10, 12, 42, 46, 53]. Different models may excel in different domains, and it has been observed that smaller LLMs can surpass larger LLMs on specific tasks [10, 14, 18].

This phenomenon raises an important question: *can we leverage a set of cost-effective LLMs, within a specified budget, to achieve performance comparable to that of a more expensive LLM?* The literature offers two main strategies to address this: *LLM ensemble* [10, 27, 42] and *tiny LM* [9, 24, 47]. The ensemble approach aims to optimize performance by combining outputs from a carefully selected set of LLMs. In contrast, the tiny LM approach applies advanced techniques to reduce the parameter counts of models without significantly compromising the performance. Specifically, FrugalGPT [10], a recent LLM ensemble work, derives an LLM cascade from a ground set optimized for given queries under a cost budget constraint. As a representative from the tiny LM family, Octopus v2 [11] fine-tunes the small model Gemma-2B to exploit fine-tuned functional tokens for function calling, achieving comparable performance with GPT-4 in accuracy and latency. Among the frameworks for LLM usage cost reduction and LLM performance enhancement, FrugalGPT is the most pertinent to our research. However, when FrugalGPT applies the derived LLM cascade to queries, it adopts the single output from the *last executed* model in the sequence rather than taking advantage of all generated responses from the model cascade to produce an optimal solution. Moreover, the budget constraint is applied to queries in an *expected* sense, allowing the incurred costs on individual queries to exceed the budget in practice, a phenomenon confirmed by our experiments. In addition, FrugalGPT does not provide any performance guarantee for the selected LLM ensemble.

Inspired by these observations, we aim to combine the responses generated by a collection of LLMs in a non-trivial manner to deliver high-quality output, with a focus on classification queries. To this end, we devise a novel aggregation approach and quantify the quality of the aggregated response via a new metric of *correctness probability* by leveraging likelihood maximization. Building on this, we formalize an optimization problem, dubbed Optimal Ensemble Selection (OES for short): given a set of LLMs, a specific budget, and a query, the objective is to identify a subset of LLMs whose total cost is within the budget while the aggregated correctness probability on the query is maximized. To address this challenge, we propose an adaptive LLM selection algorithm ThriftLLM, designed specifically for this budget-constrained LLM selection scenario. As illustrated in Figure 1, when receiving a query and budget from a user, ThriftLLM selects an appropriate subset of LLMs from the LLM candidates without exceeding the budget. These LLMs independently process the query, and their responses are subsequently aggregated by ThriftLLM to produce a final answer of high quality, which is then returned to the user. Take a traditional data management task—entity matching—as an example. A global consulting firm may collect product information from different regions in varied formats and descriptions for market analysis. For example, Product 1: *Samsung Galaxy S21 Ultra, Phantom Silver, Factory Unlocked* and Product 2: *Samsung smartphone, 256GB storage, high-end camera, silver color*. They both refer to the same high-end Samsung Galaxy S21. To ensure reliable insights, it must identify records referring to the same real-world product across sources. In our framework, each LLM serves as an operator that takes a product pair and predicts whether they refer to the same entity. These LLM operators vary in capability. Given a fixed budget, ThriftLLM selects an optimal subset of them to form an ensemble that ensures high-quality matching.

We conduct an in-depth theoretical analysis of the correctness probability function in the OES problem and establish that it is non-decreasing and non-submodular. We also provide evidence of the hardness of the problem. Nevertheless, we leverage a surrogate objective function that upper bounds the correctness probability, show that it is non-decreasing and submodular, and devise an algorithm for OES that achieves an instance-dependent approximation guarantee with high probability, when LLM success probabilities are known. We show how our algorithms and approximation guarantees can be extended to the case when the success probabilities are unknown and are estimated within confidence intervals. We compare ThriftLLM with 3 baselines on 5 real-world datasets on various text classification queries across different domains as well as with 2 SOTA baselines across 5 real-world datasets on entity matching queries. The experimental results strongly demonstrate the superior performance of ThriftLLM in terms of accuracy respecting cost budgets and delivering high-quality results under given budgets.

In sum, we make the following contributions.

- We propose a new aggregation scheme for combining individual LLM responses and formalize the ensemble prediction quality using a novel notion of *correctness probability* (Sections 2 and 3). We show that correctness probability is non-decreasing but non-submodular and offer evidence of hardness of Optimal Ensemble Selection (Section 4.1).
- We develop a greedy algorithm, GreedyLLM, and show that it has an unbounded approximation factor. We then develop the ThriftLLM algorithm by leveraging a submodular upper bound function as a surrogate objective, and show that it offers an instance dependent approximation to the

optimum with high probability (Sections 4.2-4.3). When the success probabilities are unknown and are estimated within confidence intervals, our data-dependent approximation guarantees continue to hold (Section 4.4).

- We conduct extensive experiments on text classification tasks over 5 real-world datasets across diverse domains against 5 baselines and on entity matching over 5 real-world datasets against 4 baselines. Experiments show that ThriftLLM achieves comparable or superior performance on the tested datasets while respecting the budget constraints).

2 PRELIMINARIES

We denote matrices, vectors, and sets with bold uppercase letters (e.g., $\mathbf{T}$), bold lowercase letters (e.g., $\mathbf{x}$), and calligraphic letters (e.g., $\mathcal{S}$), respectively. The i-th row (resp. column) of matrix $\mathbf{T}$ is denoted $\mathbf{T}[i,\cdot]$ (resp. $\mathbf{T}[\cdot,i]$). We use $[n]$ to denote the set $\{1,2,\cdots,n\}$.

We refer to textual tasks submitted to LLMs as *queries*. Queries typically contain contexts, called *prompts*, preceding the actual questions. In general, we assume queries include necessary prompts. Let $\mathcal{Q}$ be a query class representing a specific category of queries in the real world and $\mathcal{L} = \{l_1, l_2, \cdots, l_L\}$ be a set of distinct LLMs ($L > 0$). Given a random query $q \in \mathcal{Q}$ (with its corresponding prompt), the query processing cost of a model consists of two components: the input and output costs, which are directly determined by the number of input and output tokens, respectively. We let $b_i(q) \in \mathbb{R}_+$ denote the total incurred cost of model l_i for processing query q. When the model l_i is deterministic, its cost $b_i(q)$ is solely determined by the query q. For simplicity, we refer to this cost as b_i when the query is clear from the context. The performance of the models from $\mathcal{L}$ on the query class $\mathcal{Q}$ is represented by the set of *success probabilities* $\mathcal{P} = \{p_1, p_2, \cdots, p_L\}$, where each p_i denotes the probability that the model l_i generates a correct response to a query selected uniformly at random from $\mathcal{Q}$.

Specifically, for a classification query $q \in \mathcal{Q}$, let $\mathcal{C} = \{C_1, \ldots, C_K\}$ denote the set of $K > 0$ possible classes. When model $l_i \in \mathcal{L}$ is applied to query q, it returns a prediction response denoted $R(l_i) \in \mathcal{C}$. For a subset $\mathcal{S} \subseteq \mathcal{L}$ of LLMs, their prediction responses yield an *observation* $\phi_{\mathcal{S}} = (R(l) \mid l \in \mathcal{S})$, a prediction sequence of the LLMs from $\mathcal{S}$. The set of all possible observations for $\mathcal{S}$ on the query class $\mathcal{Q}$ is termed as *observation space* $\Omega_{\mathcal{S}}$. We denote the prediction derived from observation $\phi_{\mathcal{S}} \in \Omega_{\mathcal{S}}$ as $C(\phi_{\mathcal{S}})$. The derivation procedure is elaborated in Section 3.2.

Given a set of LLMs $\mathcal{S}$ and considering a random query q from $\mathcal{Q}$ with an associated ground-truth class C_q, the probability of observing $\phi_{\mathcal{S}}$, denoted as $\Pr[\phi_{\mathcal{S}}]$, is computed as follows. Let $\mathcal{S}^T = \{l \in \mathcal{S} \mid R(l) = C_q\}$ (resp. $\mathcal{S}^F = \{l \in \mathcal{S} \mid R(l) \neq C_q\}$) be the subset of models whose prediction agrees (resp. disagrees) with C_q. Then $\Pr[\phi_{\mathcal{S}}]$ is $\Pr[\phi_{\mathcal{S}}] = \prod_{l_i \in \mathcal{S}^T} p_i \prod_{l_j \in \mathcal{S}^F} \frac{1-p_j}{K-1}$.

Notice that a model making a wrong prediction could predict any one of the rest $K-1$ wrong classes. For example, given a query class $\mathcal{Q}$, let the LLM set $\mathcal{S} = \{l_1, l_2, l_3\}$ with corresponding success probabilities $\mathcal{P} = \{0.9, 0.8, 0.8\}$ and the class set $\mathcal{C} = \{C_1, C_2, C_3\}$. Figure 2 demonstrates the observation space $\Omega_{\mathcal{S}}$. Suppose we sample a random query q from $\mathcal{Q}$ uniformly. When the ground-truth class $C_q = C_1$, the probability of observation $\phi_3 = (C_1, C_1, C_3)$ is

$$S = \{l_1, l_2, l_3\}, \quad \mathcal{C} = \{C_1, C_2, C_3\}$$

$$\Omega_S = \left\{ \begin{array}{ll} \phi_1 = (C_1, C_1, C_1), & \phi_2 = (C_1, C_1, C_2) \\ \phi_3 = (C_1, C_1, C_3), & \phi_4 = (C_1, C_2, C_1) \\ & \cdots \\ \phi_{26} = (C_3, C_3, C_2), & \phi_{27} = (C_3, C_3, C_3) \end{array} \right\}$$

Figure 2: Example of an observation space $\Omega_{\mathcal{S}}$.

$\Pr[\phi_3] = 0.9 \times 0.8 \times \frac{1-0.8}{2} = 0.072$. Similarly, the probability is updated to $\Pr[\phi_3] = 0.0005$ or $\Pr[\phi_3] = 0.004$ if $C_q = C_2$ or $C_q = C_3$, respectively. By following this procedure, the probability $\xi_{\mathcal{P}}(\mathcal{S})$ is aggregated over the entire observation space $\Omega_{\mathcal{S}}$.

By leveraging the realization probability $\Pr[\phi_{\mathcal{S}}]$, the fundamental notion of *correctness probability* $\xi(\mathcal{S})$ of $\mathcal{S}$, i.e., the probability that $\mathcal{S}$ makes the correct prediction on a random query from query class $\mathcal{Q}$, is formalized as follows.

Definition 1 (Correctness Probability). *Given a random query q sampled uniformly from class $\mathcal{Q}$ and a subset $\mathcal{S}$ of LLMs, let C_q be the ground-truth response. Let $\Omega_{\mathcal{S}}^T \subset \Omega_{\mathcal{S}}$ be the subset of observations with $C(\phi_{\mathcal{S}}) = C_q$. The correctness probability on query class $\mathcal{Q}$ is $\xi_{\mathcal{P}}(\mathcal{S}) = \sum_{\phi_{\mathcal{S}} \in \Omega_{\mathcal{S}}^T} \Pr[\phi_{\mathcal{S}}]$.*

When the success probabilities $\mathcal{P}$ are clear from the context, we will drop the subscript and denote correctness probability as $\xi(.)$. We next formally state the problem we study in the paper.

Definition 2 (Optimal Ensemble Selection). *Consider a query class $\mathcal{Q}$, a set $\mathcal{L}$ of LLMs, and a cost budget $B \in \mathbb{R}_+$. The* Optimal Ensemble Selection *(*OES*) problem is to find a subset $\mathcal{S}^\circ \subseteq \mathcal{L}$ whose total cost is under B, such that the correctness probability $\xi(\mathcal{S}^\circ)$ on $\mathcal{Q}$ is maximized, i.e.,*

$$\mathcal{S}^\circ = \underset{\mathcal{S}: \mathcal{S} \subseteq \mathcal{L}, c(\mathcal{S}) \leq B}{\arg\max} \xi(\mathcal{S}), \tag{1}$$

where $c(\mathcal{S}) := \sum_{l_i \in \mathcal{S}} b_i$ is the total cost of LLMs in $\mathcal{S}$.

3 PROBABILITY ESTIMATION AND RESPONSE AGGREGATION

3.1 Estimation of Success Probability

We assume that queries from the same query class $\mathcal{Q}$ exhibit semantic similarity. The success probability of a model l on query class $\mathcal{Q}$ is the probability that l returns the correct response for a query q randomly sampled from $\mathcal{Q}$. This probability is crucial in addressing the OES problem. However, success probabilities of LLMs are not known *a priori* but can be estimated from historical data.

Specifically, consider an input table $\mathbf{T}$ that records the historical performance of the L LLMs on N queries in a matrix format with $L, N \in \mathbb{N}_+$. The entries of $\mathbf{T}$ vary based on the type of query q. For classification tasks, $\mathbf{T}$ contains boolean entries, i.e., $\mathbf{T} \in \{0,1\}^{N \times L}$. Conversely, for generation or regression tasks, it contains real values in the interval $[0,1]$, i.e., $\mathbf{T} \in [0,1]^{N \times L}$, with $\mathbf{T}_{\ell,k}$ indicating the quality or accuracy score of the response from the ℓ-th model on the k-th query. In this paper, we focus on classification queries.

To accurately estimate the success probability for each query class, we first cluster the queries in $\mathbf{T}$ into distinct groups based on their semantic similarity. To this end, we convert all queries into

embeddings by leveraging the embedding API [38] provided by OpenAI. Subsequently, we employ the DBSCAN [19] algorithm to cluster the embeddings. The success probability p_l of the l-th model on one resultant cluster $\mathcal{Q}_k$ is estimated as $p_l = \frac{1}{|\mathcal{Q}_k|} \sum_{q_i \in \mathcal{Q}_k} \mathbf{T}[i,l]$.

3.2 Response Aggregation

Given a set $\mathcal{S}$ of LLMs and an observation $\phi_{\mathcal{S}} = (R(l_1), R(l_2), \cdots, R(l_{|\mathcal{S}|}))$, the aggregated prediction $C(\phi_{\mathcal{S}})$ of $\phi_{\mathcal{S}}$ is derived by combining all responses in observation $\phi_{\mathcal{S}}$. Since the ground-truth C_q of query q is unknown, we take the class with maximum likelihood as the aggregated prediction $C(\phi_{\mathcal{S}})$. Specifically, each response in observation $\phi_{\mathcal{S}}$ corresponds to a class from $\mathcal{C} = \{C_1, C_2, \cdots, C_K\}$, which includes the (unknown) ground truth C_q. We consider each of the K classes in turn as the ground truth and compute the likelihood of observing $\phi_{\mathcal{S}}$. The class with the highest likelihood is selected as the prediction $C(\phi_{\mathcal{S}})$.

Initially, the set $\mathcal{S}$ is partitioned into disjoint subsets as $\mathcal{S} = \{\mathcal{S}(C_1), \mathcal{S}(C_2), \cdots, \mathcal{S}(C_K)\}$ where $\mathcal{S}(C_k) = \{l : R(l) = C_k, l \in \mathcal{S}\}$ represents the subset of models that predicted class C_k. The likelihood function $f(C_q = C_k \mid \phi_{\mathcal{S}})$, which measures the probability of C_k being the ground truth, is defined as

$$f(C_q = C_k \mid \phi_{\mathcal{S}}) = \prod_{l_i \in \mathcal{S}(C_k)} p_i \prod_{l_j \in \mathcal{S} \setminus \mathcal{S}(C_k)} \frac{1-p_j}{K-1} \quad (2)$$

for $k \in [K]$. Based on this likelihood, we can derive the prediction $C(\phi_{\mathcal{S}})$ from observation $\phi_{\mathcal{S}}$ as $C(\phi_{\mathcal{S}}) = \arg\max_{C_k \in \mathcal{C}} f(C_q = C_k \mid \phi_{\mathcal{S}})$. If there are multiple classes with the same maximal likelihood, we break the tie randomly.

Computing $f(C_q = C_k \mid \phi_{\mathcal{S}})$ for all $k \in [K]$ is expensive. However, by identifying redundant calculations, we can speed it up. Specifically, computing $f(.)$ involves a substantial amount of repeated calculations, as $\prod_{l_j \in \mathcal{S}(C_k)} \frac{1-p_j}{K-1}$ for $k \in [K]$ is repeated $K-1$ times, leading to unnecessary overheads. We can choose $C(\phi_{\mathcal{S}})$ as the class with the highest likelihood by ranking the K likelihoods $f(C_q = C_k \mid \phi_{\mathcal{S}})$ *without calculating their exact values.*

Specifically, it holds that

$$\begin{aligned} f(C_q = C_k \mid \phi_{\mathcal{S}}) &= \prod_{l_i \in \mathcal{S}(C_k)} p_i \prod_{l_j \in \mathcal{S} \setminus \mathcal{S}(C_k)} \frac{1-p_j}{K-1} \\ &= \prod_{l_i \in \mathcal{S}(C_k)} \frac{p_i(K-1)}{1-p_i} \prod_{l_j \in \mathcal{S}} \frac{1-p_j}{K-1}. \end{aligned}$$

The factor $\prod_{l_j \in \mathcal{S}} \frac{1-p_j}{K-1}$ is independent of k because it is shared across all likelihood functions $f(C_q = C_k \mid \phi_{\mathcal{S}}), \forall k \in [K]$. Consequently, it is the term $\prod_{l_i \in \mathcal{S}(C_k)} \frac{p_i(K-1)}{1-p_i}$ that determines the prediction of a given observation. For clarity, we define

$$h(C_k \mid \phi_{\mathcal{S}}) = \prod_{l_i \in \mathcal{S}(C_k)} \frac{p_i(K-1)}{1-p_i} \quad (3)$$

as the *belief* in C_k being the ground truth, conditioned on the observation $\phi_{\mathcal{S}}$ for $k \in [K]$. We use this belief in place of the likelihood when deriving the prediction from an observation. Without loss of generality, we set $h(C_k \mid \phi_{\mathcal{S}}) = \frac{p_{\min}}{2(1-p_{\min})}$ heuristically if $\mathcal{S}(C_k) = \emptyset$ where $p_{\min} = \min\{p_1, p_2, \cdots, p_L\}$. Let $H_k(\phi_{\mathcal{S}})$ be the k-th highest belief value for $k \in [K]$ among all classes. This leads to the following easily proved fact.

Fact 1. *Given an observation $\phi_{\mathcal{S}}$, the class C_k corresponding to $H_1(\phi_{\mathcal{S}})$ is the prediction derived from $\phi_{\mathcal{S}}$. In formal terms, $C(\phi_{\mathcal{S}}) = \arg\max_{C_k \in \mathcal{C}} h(C_k \mid \phi_{\mathcal{S}})$.*

Therefore, when deriving the prediction from observations, it is sufficient to examine the belief value of the subset $\mathcal{S}(C_k)$ rather than evaluating the likelihood across the entire set $\mathcal{S}$. One complication is that the ground truth C_q is unknown. Our next result shows that we do not need to know the ground truth class C_q to calculate $\xi(\mathcal{S})$.

Proposition 1. *The correctness probability $\xi(\mathcal{S})$ is independent of the ground-truth class C_q of the random query q.*

The rationale is that correctness probability $\xi(\mathcal{S})$ is determined by the set of success probabilities $\mathcal{P}$ on query class $\mathcal{Q}$. As long as $\mathcal{P}$ is fixed, varying C_q of a random query $q \in \mathcal{Q}$ does not affect the overall correctness probability $\xi(\mathcal{S})$ on $\mathcal{Q}$, i.e., it does not matter what the actual ground truth class is! The intuition lies in the fact that observations are symmetrically distributed with respect to the underlying ground-truth labels. Hence, we can assume any class in $\mathcal{C}$ to be the ground truth, without affecting the calculation of $\xi(\mathcal{S})$.

4 ADAPTIVE LLM SELECTION

4.1 Correctness Probability and Problem Complexity

The correctness probability function $\xi(\mathcal{S})$ (see Definition 1) determines the probability of correctness of a given set of LLMs $\mathcal{S}$ on a query class $\mathcal{Q}$, i.e., the probability of obtaining the correct aggregated prediction from any observation when applying $\mathcal{S}$ on a random query $q \in \mathcal{Q}$. The ultimate objective of OES is to maximize this probability by identifying a subset of LLMs. To aid our analysis, we first analyze the properties of function $\xi(\cdot)$.

Let $\mathcal{P} = \{p_1, ..., p_L\}$ and $\mathcal{P}' = \{p'_1, ..., p'_L\}$ be two sets of success probabilities of the models in $\mathcal{L}$. We write $\mathcal{P} \preceq \mathcal{P}'$ iff $p_i \leq p'_i, i \in [L]$. We have the following lemmas.

Lemma 1. *The correctness probability function $\xi(\cdot)$ is non-decreasing. Specifically, (i) for any subset of models $\mathcal{S} \subseteq \mathcal{L}$ and success probability sets $\mathcal{P}, \mathcal{P}'$ such that $\mathcal{P} \preceq \mathcal{P}'$, we have $\xi_{\mathcal{P}}(\mathcal{S}) \leq \xi_{\mathcal{P}'}(\mathcal{S})$; (ii) for any sets of models $\mathcal{S} \subset \mathcal{S}' \subseteq \mathcal{L}$, and any success probability set $\mathcal{P}$, $\xi_{\mathcal{P}}(\mathcal{S}) \leq \xi_{\mathcal{P}}(\mathcal{S}')$.*

Lemma 2. *The correctness probability function $\xi(\cdot)$ is non-submodular.*

Given this, we cannot directly find an (even approximately) optimal solution for the Optimal Ensemble Selection problem. Before we proceed, we establish the following proposition.

Proposition 2. *For a set $\mathcal{S} = \{l_1, l_2\}$ consisting of two LLMs, we have $\xi(\mathcal{S}) = \max\{p_1, p_2\}$ where p_1 and p_2 are the success probabilities of l_1 and l_2 respectively.*

The intuition is that when only two models are employed and they give two distinct predictions, the one with the higher success probability results in a higher belief value, as Equation (3) indicates, dominating the weaker one. Hence, the correctness probability is equal to the higher success probability between the two. Our proof of Lemma 2 (see Appendix A) builds on this idea.

Hardness of Optimal Ensemble Selection**.** We offer evidence of the hardness of the OES problem. We can rephrase the OES problem as, "select for each query, a subset of LLMs (items) so as to

maximize the sum of success probabilities (value) while adhering to a pre-defined cost budget (weight limit)", which is a variant of the 0-1 Knapsack problem [35], a well-known NP-hard problem. It is also worth noting that, for a given subset of LLMs, its correctness probability sums over exponentially many possible observations, which is clearly harder to compute than the "total value" in the 0-1 Knapsack problem. This suggests that the OES problem should be at least as hard as the 0-1 Knapsack problem. While proving a formal reduction is challenging due to the complex calculation of $\xi(.)$, the above argument offers some evidence that OES is likely to be computationally intractable.

4.2 Our Surrogate Greedy Strategy

As proved above, Optimal Ensemble Selection is essentially a *budgeted non-submodular maximization* problem, which is substantially challenging. In the literature [4, 6, 26, 43], the *greedy strategy* is recognized as a canonical approach for addressing combinatorial optimization problems involving submodular and non-submodular set functions. In the sequel, we first present the *vanilla greedy* strategy and demonstrate its inability to solve budgeted non-submodular maximization with theoretical guarantees. To remedy this deficiency, we propose a novel *surrogate greedy*, which can provide a data-dependent approximation guarantee.

Vanilla Greedy. Algorithm 1, dubbed GreedyLLM, presents the pseudo-code of the greedy strategy applied to the OES problem. In general, GreedyLLM selects models from the ground set $\mathcal{L}$ that achieve the highest ratio of marginal correctness gain to the associated cost in each iteration (see Line 3). When there is a tie, i.e., $\mathcal{S}'$ contains more than one model with the same maximum ratio, the tie is broken in Line 4 by choosing the model $l_* \in S'$ with the largest probability/cost ratio, which is then added to $\mathcal{S}$ if the remaining budget allows; otherwise, l_* is omitted, and GreedyLLM proceeds to the next iteration. The process terminates if either the budget is exhausted or the set $\mathcal{L}$ becomes empty.

The selection mechanism of the greedy strategy is straightforward, making the algorithm efficient. However, the greedy strategy for this budgeted non-submodular maximization problem does not come with any approximation guarantees and can indeed lead to arbitrarily bad results. For example, consider a set of LLMs $\mathcal{L} = \{l_1, l_2\}$ subject to a budget B, each with associated probabilities $\{p_1, p_2\}$ and costs $\{b_1, b_2\}$. Assume that $b_1 = B$, $b_2 \ll B$, and $p_1 \gg p_2$, yet the ratio $\frac{p_1}{b_1} < \frac{p_2}{b_2}$. In this case, $\{l_1\}$ is the optimal solution while GreedyLLM would myopically choose $\{l_2\}$ as the solution. As a consequence, this misselection results in an approximation guarantee $\frac{p_2}{p_1}$, which can be arbitrarily small.

Our Surrogate Greedy. To derive a plausible approximation guarantee for this budgeted non-submodular maximization problem, we propose a *surrogate greedy* strategy. The idea is as follows. We design a surrogate set function $\gamma(\mathcal{S})$ to approximate the correctness probability function $\xi(\mathcal{S})$. This surrogate function is devised such that i) $\gamma(\mathcal{S})$ is *submodular* and ii) $\gamma(\mathcal{S}) \geq \xi(\mathcal{S})$ holds for every subset $\mathcal{S} \subseteq \mathcal{L}$. Subsequently, we establish an approximation guarantee for $\gamma(\mathcal{S})$ concerning budgeted submodular maximization. Built on this foundation, we derive a data-dependent approximation guarantee for $\xi(\mathcal{S})$ for the Optimal Ensemble Selection problem.

Algorithm 1: LLM Selection in Greedy - GreedyLLM

Input: LLM set $\mathcal{L}$, success probability set $\mathcal{P} = \{p_1, p_2, \cdots, p_L\}$, cost set $\{b_1, \cdots, b_L\}$, budget B, and the set function $\xi(\cdot)$
Output: Subset $\mathcal{S}$

1 $\mathcal{S} \leftarrow \emptyset$;
2 **while** $B > 0$ **and** $\mathcal{L} \neq \emptyset$ **do**
3 $\quad \mathcal{S}' \leftarrow \arg\max_{l_i \in \mathcal{L}} \frac{\xi(\mathcal{S} \cup \{l_i\}) - \xi(\mathcal{S})}{b_i}$;
4 $\quad l_* \leftarrow \arg\max_{l_i \in \mathcal{S}'} \frac{p_i}{b_i}$, $\mathcal{L} \leftarrow \mathcal{L} \setminus \{l_*\}$;
5 $\quad$ **if** $B < b_*$ **then**
6 $\quad\quad$ **continue**;
7 $\quad \mathcal{S} \leftarrow \mathcal{S} \cup \{l_*\}$, $B \leftarrow B - b_*$;
8 **return** $\mathcal{S}$;

Algorithm 2: Surrogate Greedy - SurGreedyLLM

Input: LLM set $\mathcal{L}$, success probability set $\mathcal{P} = \{p_1, p_2, \cdots, p_L\}$, cost set $\{b_1, \cdots, b_L\}$, budget B, correctness function $\xi(\cdot)$, and surrogate function $\gamma(\cdot)$
Output: Subset $\mathcal{S}$

1 $l^* \leftarrow \arg\max_{l_i \in \mathcal{L}, b_i \leq B} p_i$;
2 $\mathcal{S}_1 \leftarrow$GreedyLLM $(\mathcal{L}, \mathcal{P}, \{b_1, \cdots, b_L\}, B, \xi(\cdot))$;
3 $\mathcal{S}_2 \leftarrow$GreedyLLM $(\mathcal{L}, \mathcal{P}, \{b_1, \cdots, b_L\}, B, \gamma(\cdot))$;
4 $\mathcal{S}^* \leftarrow \arg\max\{\{l^*\}, \xi(\mathcal{S}_1), \xi(\mathcal{S}_2)\}$;
5 **return** $\mathcal{S}^*$;

Specifically, we define the surrogate set function as

$$\gamma(\mathcal{S}) := 1 - \textstyle\prod_{l_k \in \mathcal{S}}(1 - p_k) \tag{4}$$

Accordingly, we have the following result.

Lemma 3. *The set function $\gamma(\mathcal{S})$ in Equation (4) is submodular and $\gamma(\mathcal{S}) \geq \xi(\mathcal{S})$ holds for $\forall \mathcal{S} \subseteq \mathcal{L}$.*

Proof of Lemma 3. Consider two subsets $\mathcal{S}_1 \subseteq \mathcal{S}_2 \subseteq \mathcal{L}$ and an LLM $l_i \in \mathcal{L} \setminus \mathcal{S}_2$. We calculate the ratio of marginal gains

$$\frac{\gamma(\mathcal{S}_1 \cup \{l_i\}) - \gamma(\mathcal{S}_1)}{\gamma(\mathcal{S}_2 \cup \{l_i\}) - \gamma(\mathcal{S}_2)} = \frac{1 - \prod_{l_k \in \mathcal{S}_1 \cup \{l_i\}}(1-p_k) - 1 + \prod_{l_k \in \mathcal{S}_1}(1-p_k)}{1 - \prod_{l_k \in \mathcal{S}_2 \cup \{l_i\}}(1-p_k) - 1 + \prod_{l_k \in \mathcal{S}_2}(1-p_k)}$$
$$= \frac{\prod_{l_k \in \mathcal{S}_1}(1-p_k) - \prod_{l_k \in \mathcal{S}_1 \cup \{l_i\}}(1-p_k)}{\prod_{l_k \in \mathcal{S}_2}(1-p_k) - \prod_{l_k \in \mathcal{S}_2 \cup \{l_i\}}(1-p_k)} = \frac{p_i \prod_{l_k \in \mathcal{S}_1}(1-p_k)}{p_i \prod_{l_k \in \mathcal{S}_2}(1-p_k)}$$
$$= \frac{1}{\prod_{l_k \in \mathcal{S}_2 \setminus \mathcal{S}_1}(1-p_k)} \geq 1.$$

It follows that $\gamma(\mathcal{S}_1 \cup \{l_i\}) - \gamma(\mathcal{S}_1) \geq \gamma(\mathcal{S}_2 \cup \{l_i\}) - \gamma(\mathcal{S}_2)$ for $\mathcal{S}_1 \subseteq \mathcal{S}_2$, showing that $\gamma(.)$ is submodular.

To compare $\xi(\mathcal{S})$ with $\gamma(\mathcal{S})$, we analyze the difference between the failure probabilities $1 - \xi(\mathcal{S})$ and $1 - \gamma(\mathcal{S}) = \prod_{l_i \in \mathcal{S}}(1 - p_i)$. The probability $1 - \xi(\mathcal{S})$ captures all cases where the aggregated prediction is incorrect. Those cases fall into two disjoint categories: **Category I** contains cases where at least one model in $\mathcal{S}$ provides a correct prediction, whereas **Category II** comprises cases where all models in $\mathcal{S}$ yield incorrect predictions. Consequently, we can express $1 - \xi(\mathcal{S}) = \Pr[\textbf{Category I}] + \Pr[\textbf{Category II}]$ and $1 - \gamma(\mathcal{S}) = \prod_{l_i \in \mathcal{S}}(1 - p_i) = \Pr[\textbf{Category II}]$. Given that $\Pr[\textbf{Category I}] \geq 0$, it follows that $1 - \xi(\mathcal{S}) \geq \prod_{l_i \in \mathcal{S}}(1 - p_i)$, implying $\gamma(\mathcal{S}) \geq \xi(\mathcal{S})$, which completes the proof. □

Khuller et al. [29] study budgeted maximum set cover, a well-known submodular optimization problem, and propose a modified

greedy algorithm that has a $(1 - \frac{1}{\sqrt{e}})$-approximation guarantee[1]. Basically, the algorithm selects a single set $\mathcal{S}_1$ whose coverage is maximum within the budget; if the coverage of $\mathcal{S}_1$ is more than that of the vanilla greedy solution $\mathcal{S}_2$, then return $\mathcal{S}_1$; otherwise, return $\mathcal{S}_2$. By leveraging this as a building block, we now introduce our surrogate greedy approach, dubbed SurGreedyLLM, for the budgeted non-submodular LLM selection problem in Algorithm 2.

In particular, SurGreedyLLM first identifies the model l^* with the highest success probability under the budget B. Next, it derives solution sets $\mathcal{S}_1$ and $\mathcal{S}_2$ by leveraging GreedyLLM on set functions $\xi(\cdot)$ and $\gamma(\cdot)$ respectively. The one with the highest correctness probability among the three sets $\{\{l^*\}, \mathcal{S}_1, \mathcal{S}_2\}$ is returned as the final solution. The following theorem shows that SurGreedyLLM provides a data-dependent approximation guarantee.

Theorem 3. *Consider sets $\{l^*\}$, $\mathcal{S}_1$, $\mathcal{S}_2$, and $\mathcal{S}^*$ derived from SurGreedyLLM. It holds that*

$$\xi(\mathcal{S}^*) \geq \frac{\max\{\xi(S_1), \xi(S_2), p^*\}}{\max\{\gamma(S_2), p^*\}}(1 - \frac{1}{\sqrt{e}}) \cdot \xi(\mathcal{S}^\circ), \tag{5}$$

where p^* is the success probability of l^* and $\mathcal{S}^\circ$ is the optimal solution of Optimal Ensemble Selection.

4.3 Surrogate Greedy: Further Optimizations

In this section, we seek further optimizations on Algorithm 2. Specifically, we shall first show that it is possible to further optimize the solution $\mathcal{S}^*$ returned by Surrogate Greedy by identifying models in $\mathcal{S}^*$ that can be safely pruned without changing the final prediction. Eliminating models from $\mathcal{S}^*$ helps cut down the cost incurred by the final solution. The intuition why this works is because when we apply models in $\mathcal{S}^*$ successively on a given task, by observing the predictions obtained so far we may be able to determine that the remaining models cannot influence the final aggregated prediction.

Secondly, we note that calculating the correctness function $\xi(\cdot)$ exactly is expensive. Therefore, for practical implementation, we estimate $\xi(\cdot)$ using θ Monte Carlo simulations, where θ is determined by input parameters $\epsilon, \delta \in (0, 1)$. The principle for selecting appropriate values for ϵ and δ is discussed in Section 4.3.2.

4.3.1 Adaptive Selection. After obtaining $\mathcal{S}^*$ from Algorithm 2, it can be further reduced at model invocation time, in an adaptive manner tailored for practical scenarios. Specifically, when we apply models from $\mathcal{S}^*$ in sequence on a given query, a tipping point arises upon which the aggregated prediction from the models applied so far cannot be not influenced by subsequent models from $\mathcal{S}^*$ not yet used. At this juncture, the aggregated prediction can be deemed final and returned in response to the query. This procedure enables the derivation of a subset of LLMs $\mathcal{S}$ from $\mathcal{S}^*$ with a reduced cost by leveraging real-time observational feedback, while ensuring the same prediction as that of $\mathcal{S}^*$. Building on this pivotal insight, we have developed an adaptive LLM selection strategy and introduce ThriftLLM, detailed in Algorithm 3.

We first initialize $\mathcal{T}^* = \mathcal{S}^*$ obtained from Algorithm 2 and select models from $\mathcal{T}^*$, add them to $\mathcal{S}$ while removing them from $\mathcal{T}^*$. Let $\mathcal{S}$ be the current set of selected LLMs from $\mathcal{T}^*$, i.e., $\mathcal{S} = \mathcal{S}^* \setminus \mathcal{T}^*$, and $\phi_{\mathcal{S}}$ be any real-time observation of $\mathcal{S}$ on input random query q. In each iteration before the selection, the algorithm checks the termination condition $F(\mathcal{T}^*)H_2(\phi_{\mathcal{S}}) > H_1(\phi_{\mathcal{S}})$ where $F(\mathcal{T}^*) := \prod_{l_i \in \mathcal{T}^*} \frac{p_i(K-1)}{1-p_i}$ is the potential belief value of set $\mathcal{T}^*$. In particular, the potential belief $F(\mathcal{S})$ of a set $\mathcal{S}$ represents the maximum possible belief value that the set $\mathcal{S}$ could contribute to any class. Note that the set $\mathcal{T}^*$ is updated (Line 7 in Algorithm 3) during each selection and contains the remaining unselected LLMs.

Algorithm 3: Adaptive LLM Selection - ThriftLLM

Input: Set $\mathcal{L}$ of LLMs, set $\mathcal{P}$ of success probability, cost $b_1, \cdots, b_L$, budget B, parameters ϵ, δ, and a random query $q \in Q$
Output: Subset $\mathcal{S}$ and prediction on q

1 $p^* \leftarrow \max\{p_i : l_i \in \mathcal{L}, b_i \leq B\}$, $\theta := \frac{8+2\epsilon}{\epsilon^2 p^*} \ln(\frac{2L^2}{\delta})$;
2 $\mathcal{S}^* \leftarrow$ Apply Algorithm 2 with θ Monte Carlo simulations for $\xi(\cdot)$ estimation;
3 $\mathcal{T}^* \leftarrow \mathcal{S}^*$, $\mathcal{S} \leftarrow \emptyset$, $\phi_{\mathcal{S}} \leftarrow \emptyset$, $H_2(\phi_{\mathcal{S}}) \leftarrow 1$, $H_1(\phi_{\mathcal{S}}) \leftarrow 1$;
4 **while** $\mathcal{T}^* \neq \emptyset$ **do**
5 **if** $F(\mathcal{T}^*)H_2(\phi_{\mathcal{S}}) > H_1(\phi_{\mathcal{S}})$ **then**
6 $l^* \leftarrow \arg\max_{l_i \in \mathcal{T}^*} p_i$;
7 $\mathcal{S} \leftarrow \mathcal{S} \cup \{l^*\}$, $\mathcal{T}^* \leftarrow \mathcal{T}^* \setminus \{l^*\}$;
8 Apply l^* on query q and update observation $\phi_{\mathcal{S}}$;
9 **else**
10 **break**;
11 **return** $\mathcal{S}$ and the prediction with belief $H_1(\phi_{\mathcal{S}})$;

If the condition is satisfied (Line 5), this suggests the possibility that the application of the residual set $\mathcal{T}^*$ to query q can yield a prediction that differs from the existing prediction associated with the belief value $H_1(\phi_{\mathcal{S}})$. We formalize this observation in Proposition 4. In this scenario, we persist in picking the model l^* with the largest success probability from $\mathcal{T}^*$ into set $\mathcal{S}$. Subsequently, l^* is applied to query q, and observation $\phi_{\mathcal{S}}$ is updated accordingly. This procedure terminates if the condition is not met or $\mathbf{T}^*$ is empty.

Proposition 4. *If the condition $F(\mathcal{T}^*)H_2(\phi_{\mathcal{S}}) \leq H_1(\phi_{\mathcal{S}})$ holds in Algorithm 3, the prediction by set $\mathcal{S}$ is the same as the prediction by set $\mathcal{S}^*$.*

4.3.2 Approximation Guarantee and Time Complexity. In Algorithm 3, correctness probability values of all subsets examined in SurGreedyLLM are estimated using a sufficient number of Monte Carlo simulations. We quantify the confidence of the estimation interval in the following result, derived using Hoeffding's inequality.

Lemma 4. *Consider an arbitrary set $\mathcal{S} \subseteq \mathcal{L}$ with correctness probability $\xi(\mathcal{S})$. The correctness probability estimation $\tilde{\xi}(\mathcal{S})$ in Algorithm 3 satisfies:*

$$\Pr\left[|\xi(\mathcal{S}) - \tilde{\xi}(\mathcal{S})| \leq \frac{\epsilon p^*}{2}\right] \geq 1 - \delta/L^2, \tag{6}$$

where $\epsilon, \delta \in (0, 1)$, $p^* = \max\{p_i : l_i \in \mathcal{L}, b_i \leq B\}$, and $\tilde{\xi}(\mathcal{S})$ is averaged from $\theta = \frac{8+2\epsilon}{\epsilon^2 p^*} \ln(\frac{2L^2}{\delta})$ Monte Carlo simulations.

Lemma 4 directly follows from Lemma 3 in [48]. Building on Lemma 4, we establish the approximation guarantee of the subset of LLMs from ThriftLLM.

[1]They also propose a greedy algorithm with a $(1 - 1/e)$ approximation guarantee, but its prohibitive $O(n^5)$ complexity makes it impractical for use.

Theorem 5. *Given parameters $\epsilon, \delta \in (0,1)$, let $\mathcal{S}^*$ be the solution returned by ThriftLLM and $\mathcal{S}^\circ$ be the optimal solution to the* Optimal Ensemble Selection *problem. Then we have*

$$\Pr\left[\xi(\mathcal{S}^*) \geq \left(\frac{\max\{\xi(S_1), \xi(S_2), p^*\}}{\max\{\gamma(\mathcal{S}_2), p^*\}} - \epsilon\right)\left(1 - \frac{1}{\sqrt{e}}\right) \cdot \xi(\mathcal{S}^\circ)\right] \geq 1 - \delta,$$

where $\mathcal{S}_1$, $\mathcal{S}_2$, and p^* are derived from SurGreedyLLM.

Time Complexity. The time complexity of Algorithm 3 is dominated by the selection process with set function $\xi(\cdot)$. Specifically, there are at most $O(L^2)$ evaluations of correctness probability. Each evaluation invokes θ Monte Carlo simulations, and each Monte Carlo simulation conducts $O(L)$ evaluations. Thus, the overall time complexity is $O(\theta L^3) = O(\frac{L^3}{\epsilon^2 p^*} \ln(\frac{2L^2}{\delta})) = O(\frac{L^3}{\epsilon^2} \ln(\frac{L}{\delta}))$.

4.4 Extension to Probability Interval Estimates

Our algorithms as well as the approximation analysis assume the precise ground truth success probabilities of the models $\mathcal{L}$ are available. In practice, they are unavailable and must be estimated, e.g., using the historical query responses of these models as illustrated in Section 3.1. These estimates have an associated confidence interval. Specifically, given arbitrary sample sizes, confidence intervals can be derived by leveraging concentration inequalities [5]. We next show how our algorithms (and analysis) can be extended to work with confidence intervals associated with estimates of success probabilities. For clarity, denote the estimate of ground truth success probability p_l as $\hat{p}_l$. Let this estimate $\hat{p}_l$ have an associated confidence interval $[p_l^\perp, p_l^\top]$ at a confidence level of $1 - \delta_l$, where $\delta_l \in (0,1)$. In the following, we explore the approximation guarantee of ThriftLLM when these intervals are provided as inputs.

Let $\mathcal{P}_{\text{low}} = \{p_1^\perp, p_2^\perp, \cdots, p_L^\perp\}$, $\hat{\mathcal{P}} = \{\hat{p}_1, \hat{p}_2, \cdots, \hat{p}_L\}$, and $\mathcal{P}_{\text{up}} = \{p_1^\top, p_2^\top, \cdots, p_L^\top\}$ be the sets of lower bounds, estimated values, and upper bounds corresponding to model success probabilities, respectively. The corresponding correctness functions $\xi(\cdot)$, $\xi_l(\cdot)$, and $\xi_u(\cdot)$ are defined based on *ground-truth* probabilities, $\mathcal{P}_{\text{low}}$, and $\mathcal{P}_{\text{up}}$, respectively. Although the same observation space is shared by the four scenarios involving ground-truth probabilities, $\mathcal{P}_{\text{low}}$, $\hat{\mathcal{P}}$ and $\mathcal{P}_{\text{up}}$, the corresponding probability distributions of observations intrinsically differ. Run Algorithm 3 with each of the success probability sets $\mathcal{P}_{\text{low}}$, $\hat{\mathcal{P}}$, and $\mathcal{P}_{\text{up}}$ and let $\mathcal{S}_l^*$, $\mathcal{S}^*$, and $\mathcal{S}_u^*$ be the solution returned by the algorithm on these inputs respectively. Based on this, we can establish the following theorem.

Theorem 6. *Consider a set $\mathcal{L} = \{l_1, l_2, \cdots, l_L\}$ of LLMs and a random query q from class $\mathcal{Q}$. Suppose the success probability of model l on q is estimated as $\hat{p}_l$ with confidence interval $[p_l^\perp, p_l^\top]$ at a confidence level of $1 - \delta_l$ for $\delta_l \in (0,1)$. Given approximation parameters $\epsilon, \delta \in (0,1)$, we have*

$$\Pr\left[\frac{\xi(\mathcal{S}^*)}{\xi(\mathcal{S}^\circ)} \geq \frac{\xi_l(\mathcal{S}_l^*)}{\xi_u(\mathcal{S}_u^*)}\left(\frac{\max\{\xi_u(S_{u1}), \xi_u(S_{u2}), p_u^*\}}{\max\{\gamma_u(\mathcal{S}_{u2}), p_u^*\}} - \epsilon\right)\left(1 - \frac{1}{\sqrt{e}}\right)\right] \geq 1 - (\delta + L^2 \sum_{l=1}^{L} \delta_l),$$

where $\gamma_u(\cdot)$ is the surrogate set function, S_{u1} and S_{u2} are selected by SurGreedyLLM on $\mathcal{P}_{up}$, respectively.

To ensure this data-dependent approximation guarantee holds with high probability, a failure probability $\delta + L^2 \sum_{l=1}^{L} \delta_l \ll 1$ is necessary. To this end, the term $L^2 \sum_{l=1}^{L} \delta_l$ is supposed to be in the scale of δ, i.e., $L^2 \sum_{l=1}^{L} \delta_l = \Theta(\delta)$. In the following, we discuss how to ensure a small failure probability.

Diminishing failure probability. When substantial samples are available for query clusters, the failure probability (confidence level) in Theorem 6 can be significantly improved by enlarging the sample sizes. However, this may not be possible due to the lack of samples in real-world applications. In this scenario, we can calibrate the failure probability by repeatedly estimating the confidence intervals $[p_l^\perp, p_l^\top]$ with estimation $\hat{p}_l$ until a targeted failure probability (confidence level) is reached. Specifically, we sample a certain number of queries from each cluster and derive the confidence interval at a certain confidence level. We then repeat this procedure a sufficient number of times and return the median value among all estimates. By doing this, we can diminish the failure probability to a desired level. Formally, we establish the following Lemma.

Lemma 5. *Let $\delta_l \in (0,1)$ be the failure probability of confidence interval $[p_l^\perp, p_l^\top]$ with estimate $\hat{p}_l$ of success probability p_l derived by a sampling procedure $\mathcal{A}$, such that*

$$\Pr\left[p_l^\perp \leq p_l \leq p_l^\top\right] \geq 1 - \delta_l.$$

By independently repeating $\mathcal{A}$ a total of Λ_l times, and taking the interval with the corresponding estimates being the median among the Λ_l estimates, denoted as $[\bar{p}_l^\perp, \bar{p}_l^\top]$, we have

$$\Pr[\bar{p}_l^\perp \leq p_l \leq \bar{p}_l^\top] \geq 1 - \exp\left(-\frac{\Lambda_l(1-2\delta_l)^2}{2}\right). \tag{7}$$

Theoretically, we aim to limit probability $L^2 \sum_{l=0}^{L} \delta_l = \Theta(\delta)$ in Theorem 6. Lemma 5 proves that δ_l can be diminished to $\exp(-\frac{\Lambda_l(1-2\delta_l)^2}{2})$. In this regard, we can simply ensure $L^2 \exp(-\frac{\Lambda_l(1-2\delta_l)^2}{2}) \leq \frac{\delta}{L}$. Therefore, we have $\Lambda_l = \frac{6\log(L/\delta)}{(1-2\delta_l)^2}$.

5 RELATED WORK

LLM Ensemble. FrugalGPT [10] aims to reduce the utilization cost of large language models (LLMs) while improving the overall performance. In particular, it derives an LLM cascade from a candidate set tailored for queries under a budget constraint. However, its performance is suboptimal as only the response from the last executed model in the cascade is adopted, without exploiting previous responses. Furthermore, it generates one LLM cascade for the whole dataset, i.e., for all query classes, which can lead to inferior performance due to the inherent diversity of the datasets and query classes. LLM-Blender [27] is another recently proposed LLM ensemble approach, but it does not incorporate any budget constraint. Instead, it considers a set of existing LLMs from different mainstream providers. It first applies all models to the given query and selects the top-K responses using a ranking model called PairRanker. The K responses concatenated with the query are fed into a fine-tuned T5-like model, namely the GenFuser module, to generate the final response. In the development of LLMs, their performance grows gradually over time due to the scaling law and fine-tuning. LLM-Ensemble [17] learns aggregation weights for each LLM and forms an ensemble by weights for the final response. Similarly, LLM-Topla [49] selects a subset of LLMs optimized for diversity using a genetic algorithm. To predict the increasing performance and capture the convergence point along time, Xia et al. [53] develop a

time-increasing bandit algorithm TI-UCB. Specifically, TI-UCB identifies the optimal LLM among candidates regarding development trends with theoretical logarithmic regret upper bound. Different from TI-UCB on a single optimal LLM, C2MAB-V proposed by Dai et al. [13] seeks the optimal LLM combinations for various collaborative task types. It employs combinatorial multi-armed bandits with versatile reward models, aiming to balance cost and reward. Recently, Shekhar et al. [42] try to reduce the usage costs of LLM on document processing tasks. In particular, they first estimate the ability of each individual LLM by a BERT-based predictor. By taking these estimations as inputs, they solve the LLM selection as a linear programming (LP) optimization problem and propose the QC-Opt algorithm. Instead of selecting combinations of well-trained LLMs, Bansal et al. [2] propose to compose the internal representations of several LLMs by leveraging a cross-attention mechanism, enabling new capabilities. Recently, Octopus-v4 [12] has been proposed as an LLM router. It considers multiple LLMs with expertise in different domains and routes the queries to the one with the most matched topic. However, (i) it does not consider the budget constraint but only LLM expertise when routing, and (ii) it employs one single model instead of an LLM ensemble for enhanced performance. Among the above works, FrugalGPT [10] and LLM-Blender [27] have similar goals and are most relevant to our work. We experimentally compare with them in Section 6.

Entity Matching. Entity Matching (EM) aims to identify whether two records from possibly different tables refer to the same real-world entity. It is also known as record linkage or entity resolution [22]. According to [3], EM consists of five subtasks: data preprocessing, schema matching, blocking, record pair comparison, and classification. Magellan [30] serves as a representative end-to-end EM system, though it requires external human programming. An emerging and fruitful line of work [3, 7, 15, 36, 54, 59] proposes applying deep learning-based methods to improve the classification accuracy and automate the EM process. DeepMatcher [36] utilizes an RNN architecture to aggregate record attributes and perform comparisons based on the aggregated representations. DeepER [15] employs GloVe to generate word embeddings and trains a bidirectional LSTM-based EM model to obtain record embeddings. AutoEM [59] introduces a methodology to fine-tune pre-trained deep learning-based EM models using large-scale knowledge base data through transfer learning. Another line of research, such as EmbDI [7] and HierGAT [54], leverages graph structures to improve EM by learning proximity relationships between records. With the emergence of transformer-based language models like BERT [28] and RoBERTa [33], Ditto [32] fine-tunes these pre-trained models with EM corpora and introduces three optimization techniques to improve matching performance. As the state-of-the-art method, Peeters et al. [41] shows that LLMs with zero-shot learning outperform pre-trained language model-based methods, offering a more robust, general solution.

6 EXPERIMENTS

6.1 Experimental Setup

Datasets for text classification query. We conduct experiments on 5 datasets across various real-world applications, namely Overruling, AGNews, SciQ, Hellaswag, and Banking77, as summarized in Table 1. Specifically, Overruling [60] is a legal document dataset designed to determine if a given sentence is an overruling. In particular, an overruling sentence overrides the decision of a previous case as a precedent. AGNews [58] contains a corpus of news articles categorized into four classes: *World*, *Sports*, *Business*, and *Science/Technology*. Hellaswag [55] consists of unfinished sentences for commonsense inference. Specifically, given an incomplete sentence, models are required to select the most likely follow-up sentence from 4 candidates. SciQ [52] is a collection of multiple-choice questions from science exams, with a unique correct option. Finally, Banking77 [8] dataset consists of online banking queries from customer service interactions, with each query assigned a label corresponding to one of 77 fine-grained intents.

Datasets for entity matching query. For this task, we used five datasets with the same setup as in [41]. Dataset names and detailed statistics are presented in Table 2. On each dataset, the model is required to determine whether two real-world entity descriptions (records) refer to the same entity, answering either *yes* or *no*. Among

Table 1: Dataset details for text classification

Dataset	Overruling	AGNews	SciQ	Hellaswag[2]	Banking77
Domain	Law	News	Science	Commonsense	Banking
Sizes	2.1 K	7.6 K	12.7*K*	15 K	13 K
#Classes	2	4	4	4	77

Table 2: Dataset details for entity matching

Dataset	Training set		Test set	
	# Pos	# Neg	# Pos	# Neg
(WDC) - WDC Products	500	2,000	250	989
(A-B) - Abt-Buy	616	5,127	206	1,000
(W-A) - Walmart-Amazon	576	5,568	193	1,000
(A-G) - Amazon-Google	699	6,175	234	1,000
(D-S) - DBLP-Scholar	3,207	14,016	250	1,000

Table 3: Summary of commercial LLM APIs.

Company	LLM APIs	Size (B)	Cost/1M tokens (usd) Input	Output
OpenAI	GPT-4o-mini	N.A.	0.15	0.6
	GPT-4o	N.A	5.0	15.0
Google	Gemini-1.5 Flash	N.A.	0.075	0.3
	Gemini-1.5 Pro	N.A	3.5	10.5
	Gemini-1.0 Pro	N.A.	0.5	1.5
Microsoft	Phi-3-mini	3.8	0.13	0.52
	Phi-3.5-mini	3.8	0.13	0.52
	Phi-3-small	7	0.15	0.6
	Phi-3-medium	14	0.17	0.68
Meta	Llama-3 8B	8	0.055	0.055
	Llama-3 70B	70	0.35	0.4
Mistral AI	Mixtral-8x7B	46.7	0.24	0.24

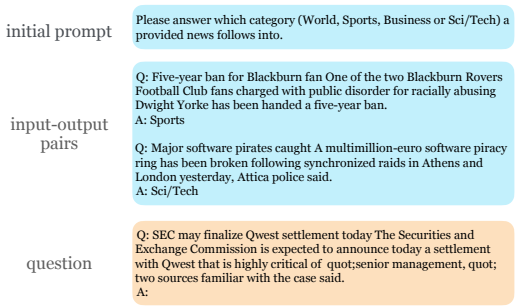


Figure 3: Prompt template for AGNews dataset.

these five datasets, DBLP-Scholar is a bibliographic entity dataset, while the remaining four are e-commerce datasets.

LLMs. We incorporate 12 commercial LLM APIs provided by five leading companies – OpenAI, Google, Microsoft, Meta, and Mistral AI. The details of these LLMs are summarized in Table 3. As shown, we select state-of-the-art LLMs from five companies as candidates. Additionally, we present the costs per 1 million input and 1 million output tokens, which vary from $0.055 to $15.0. Typically, larger models incur higher costs. To reduce the internal model randomness, we set the lowest temperature for all LLM candidates, thereby ensuring more deterministic responses.

Baselines. For *text classification*, we compare ThriftLLM with GreedyLLM and four related models, namely FrugalGPT [10], LLM-Blender [27], LLM-Topla [49], and LLM-Ensemble [17]. FrugalGPT, LLM-Ensemble, LLM-Topla, and LLM-Blender are described in detail in Section 5. Since LLM-Ensemble didn't address cost constraint in their original work, we apply a straightforward approach by greedily selecting the top-K weighted LLMs until the budget is met. For LLM-Blender, we utilize the latest models for PairRanker and GenFuser, following the original parameter settings recommended by the authors. For *entity matching*, we include the SOTA methods RoBERTa [33] finetuned by [41] and Ditto [32].

Parameter settings. In the experiments, we split the datasets for text classification into 80%/20% as historical and test sets, which form two disjoint query sets. As with datasets for entity matching, we follow the train/test splits as those in [41]. For ThriftLLM, we fix parameters $\epsilon = 0.1$ and $\delta = 0.01$. According to the actual query costs, we set a series of budgets $B = \{1.0, 5.0, 10, 50, 100\} \times 10^{-5}$ (USD) such that only subsets of LLMs in Table 3 are feasible given those budget constraints. Empirically, for a single query, when $B = 1.0 \times 10^{-5}$ USD, no more than 2–3 models are typically selected; whereas for $B = 1.0 \times 10^{-3}$ USD, up to 9–10 models are chosen.

Running environment. Our experiments are conducted on a Linux machine with an NVIDIA RTX A5000 (24GB memory), Intel Xeon(R) CPU (2.80GHz), and 500GB RAM.

Prompt Engineering. We design two-shot prompting templates for text-classification datasets to ensure models generate outputs in the desired format. For AGNews, we adopt the prompt template from FrugalGPT [10] but limit input-output examples to two, as shown in Figure 3. The blue text blocks contain the prompt and examples, while the orange block contains the target question. This structure is applied across all datasets. For other datasets in text classification, we randomly select two training records as input-output pairs. For datasets in entity matching, we follow the procedure outlined in [41] and use a zero-shot prompt with two templates: *domain-complex* for DBLP-Scholar and Amazon-Google, and *general-complex* for the others.

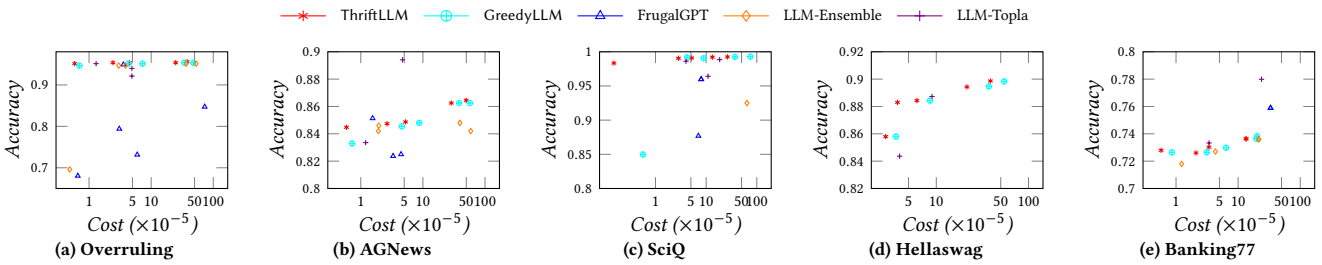
6.2 Performance on Text Classification Query

The tested methods are evaluated in terms of accuracy scores against LLM usage costs. Figure 4 presents the results of the accuracy versus costs of the 3 tested methods except for LLM-Blender on the five datasets. In particular, FrugalGPT encounters the out-of-memory issue on our machine (24GB GPU memory) on dataset Hellaswag, so its performance is not reported in Figure 4d. Moreover, FrugalGPT enforces budget constraints based on the *expected* training cost and does not respect the budget constraint strictly in a per-query manner, unlike ThriftLLM. For a fair comparison, we modify the budget constraint of FrugalGPT on testing queries to align with this per-query approach in our experiments. LLM-Blender utilizes all 12 LLM candidates for response collection, subsequently aggregating the most prominent responses to formulate the final solution. Given that LLM-Blender is not budget-aware, it is not appropriate to compare it with budget-constrained methods across different budget scenarios. As such, we report the performance of LLM-Blender and compare it with ThriftLLM separately in Table 4.

Figure 4 demonstrates the accuracy scores with the corresponding utilized cost of each method on the 5 datasets for text classification queries. As shown, ThriftLLM consistently outperforms all other baseline models with either superior accuracy at the lowest costs or the highest accuracy with lower costs on all datasets. In particular, it achieves the highest accuracy on 4 out of the 5 tested datasets, except on Banking77. ThriftLLM may select several weaker models instead of stronger ones on Banking77. This selection discrepancy arises because the success probabilities of these weaker models are overestimated when queries are from a substantial number of distinct classes. Compared with GreedyLLM, ThriftLLM acquires comparable accuracy scores but consumes notably lower costs. This observation demonstrates the effectiveness of adaptive selection in ThriftLLM on cost saving without sacrificing the performance. On datasets AGNews, Hellaswag, and Banking77, where the accuracy scores do not approach 1, ThriftLLM exhibits an ability to enhance accuracy further as costs increase. This indicates that ThriftLLM effectively harnesses the capabilities of the LLM ensemble and scales efficiently with increased budget allocation. For small budgets, LLM-Ensemble suffers significant performance degradation because the single top-weighted model exceeds the budget. LLM-Topla performs worse than ThriftLLM except on AGNews and Banking77. The performance gains stem from LLM-Topla being fine-tuned individually on each dataset, which incurs substantially higher computational cost.

Overall, ThriftLLM demonstrates a steadily strong performance, outperforming the baselines. The analysis reveals a general trend where ThriftLLM provides higher accuracy at lower cost levels, indicating its efficiency in utilizing computational resources.

Comparison with LLM-Blender. In Table 4, we compare the best accuracy scores of ThriftLLM across five different budget settings with the accuracy of LLM-Blender, which uses all model outputs as candidates for response selection. Despite this, it can be seen that



(a) Overruling (b) AGNews (c) SciQ (d) Hellaswag (e) Banking77

Figure 4: Accuracy vs cost for text classification query.

Table 4: Accuracy (%) of ThriftLLM and LLM-Blender.

Dataset	Overruling	AGNews	SciQ	Hellaswag	Banking77
ThriftLLM	95.60	86.45	99.25	89.94	73.82
LLM-Blender	89.35	83.02	90.06	53.18	52.08

on all datasets ThriftLLM clearly outperforms LLM-Blender by a significant margin. Distinct from the response aggregation mechanism in ThriftLLM, LLM-Blender relies on the GenFuser component, a generative model fine-tuned on the T5-like architecture, to generate the final outputs by fusing collected candidate responses with additional query interpretations, which, however, leads to suboptimal quality. These results reveal the superior performance of ThriftLLM over LLM-Blender in text classification even with smaller budgets.

6.3 Performance on Entity Matching Query

Figure 5 displays the F1 scores against the utilized costs of the three tested models. RoBERTa and Ditto, both based on the BERT [28] architecture, are fine-tuned tailored for each tested dataset. Consistent with the experiment settings in [41], we incorporate their reported results for RoBERTa and Ditto to ensure comparability. However, note that the corresponding utilization costs are not disclosed in [41]. To address this gap, we estimate the average cost per query by leveraging the reported fine-tuning time and the corresponding AWS pricing for computation time, given that their experiments were conducted on a p3.8xlarge AWS EC2 machine with 4 V100 GPUs (1 GPU per run).

As shown in Figure 5 for entity matching queries, ThriftLLM persistently dominates the baselines, yielding higher F1 scores but incurring lower costs on the four e-commerce datasets and acquiring the highest F1-score on DBLP-Scholar. In particular, ThriftLLM boosts the F1 scores with a notable improvement of 3.51%, 4.39%, 5.84%, 6.42%, and 0.30% on the 5 datasets respectively. Meanwhile, the observed performance pattern, where F1 scores increase with increasing budgets, signifies the strength of ThriftLLM to maximize the efficiency of allocated budgets thereby enhancing overall performance. This observation demonstrates that ThriftLLM aggregates less expensive LLMs in an effective manner, yielding superior performance at reduced costs.

In general, ThriftLLM achieves higher accuracy as the budget increases. When the budget is low, around $1 \sim 5 \times 10^{-5}$ USD, ThriftLLM tends to select $3 \sim 5$ weaker but inexpensive models that, when combined, offer improved ensemble performance. As the budget rises to $50 \sim 100 \times 10^{-5}$ USD, ThriftLLM typically selects $1 \sim 2$ stronger yet costly models (e.g., GPT-4o and Gemini-1.5 Pro), complemented by $5 \sim 7$ cheaper ones, to form a more effective ensemble, which enables flexible budget-adaptive selection.

Table 5: Accuracy (%) across confidence intervals on AGNews.

α	0	0.02	0.04	0.08	0.1
Acc. of $\mathcal{P}_{\text{low}}$	84.80	84.93	84.80	84.80	84.74
Acc. of $\mathcal{P}_{\text{up}}$	84.80	84.80	84.87	84.73	84.80

Table 6: Accuracy (%) of ThriftLLM vs single LLMs.

Dataset	Overruling	AGNews	SciQ	Hellaswag	Banking77
ThriftLLM	**95.60**	86.45	**99.25**	89.86	73.82
GPT-4o	94.68	86.71	99.17	**93.29**	**75.05**
Gemini-1.5 Pro	95.14	**89.01**	96.61	90.96	25.91
Phi-3-medium	95.14	84.21	98.50	88.73	59.84
Llama-3 70B	94.68	81.31	98.86	86.53	69.66
Mixtral-8x7B	94.90	79.34	96.29	N.A.	N.A.

6.4 Ablation Study

Confidence interval on approximation guarantees. Let α represent the length of the confidence interval in Section 4.4, i.e., $\alpha = p_l^{\top} - p_l^{\perp}$ for $l \in [L]$. Specifically, given the current estimated probability $\hat{p}_l$, we set $p_l^{\perp} = \hat{p}_l - \frac{\alpha}{2}$ and $p_l^{\top} = \min\{\hat{p}_l + \frac{\alpha}{2}, 1.0\}$. By feeding the resultant probability sets $\mathcal{P}_{low} = \{p_1^{\perp}, p_2^{\perp}, \cdots, p_L^{\perp}\}$ and $\mathcal{P}_{up} = \{p_1^{\top}, p_2^{\top}, \cdots, p_L^{\top}\}$ to ThriftLLM respectively, we record the accuracy scores of the selected LLMs.

We vary $\alpha = \{0, 0.02, 0.04, 0.08, 0.1\}$ and conduct this experiment on dataset AGNews with the budget $B = 1 \times 10^{-5}$. The results are reported in Table 5. The accuracy score of 84.80% with $\alpha = 0$ acts as the base case. Compared with this case, accuracy scores with $\alpha > 0$ are either the same or approximately 84.80% with slight variations incurred by the inherent randomness of accuracy estimation in model selection. This observation reveals that ThriftLLM is robust to the estimation errors in success probabilities.

ThriftLLM vs Single LLM. To further validate the advantage of the LLM ensemble over an individual LLM, we compare the accuracy scores obtained by ThriftLLM with single models in Table 3 on the 5 tested datasets for text classification queries. For a convincing comparison, we select the most powerful and most expensive LLMs provided by each company, including GPT-4o from OpenAI,

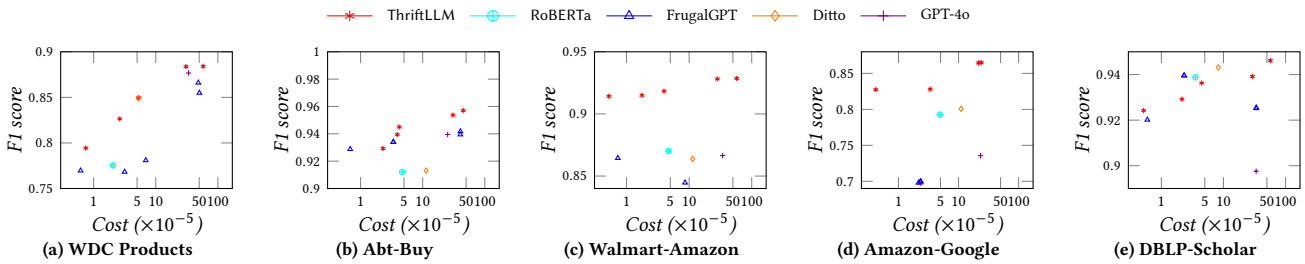


Figure 5: F1 score vs cost for entity matching query.

Table 7: Accuracy (%) across historical data on Overruling.

Budget	20%	40%	60%	80%	Original
1.0×10^{-5}	95.37	94.91	96.06	94.90	95.14
5.0×10^{-5}	95.37	95.37	95.60	95.13	95.37
10.0×10^{-5}	95.37	95.37	95.37	95.60	95.37
50.0×10^{-5}	95.60	95.37	95.37	95.13	95.37
100.0×10^{-5}	95.60	95.37	95.37	95.60	95.60

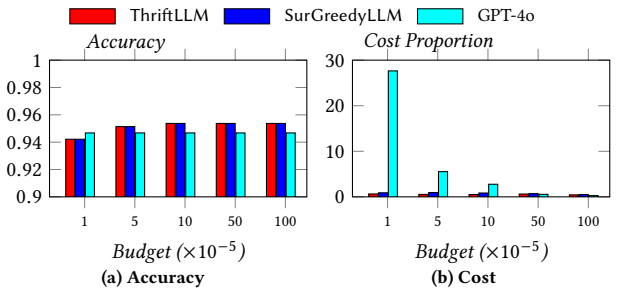


Figure 6: Accuracy vs cost on Overruling.

Gemini-1.5 Pro from Google, Phi-3-medium from Microsoft, Llama-3 70B from Meta, and Mixtral-8x7B from Mistral AI. The results are summarized in Table 6. For clarity, we highlight the highest accuracy score in bold and underline the second-highest score for each dataset. As displayed, ThriftLLM achieves the best on 2 out of 5 datasets. On the other 3 datasets, ThriftLLM either achieves or closely approaches the second-highest accuracy scores with negligible gaps. This evidence not only underscores the superior performance of ThriftLLM as an ensemble model across diverse topics but also suggests that individual powerful models do not consistently offer advantages across all domains.

Adaptive selection. We have proved (see Proposition 4) that the subset of LLMs selected by ThriftLLM makes the same prediction as that selected by SurGreedyLLM, while utilizing lower budgets. To verify this point and quantify the saved costs, we evaluate ThriftLLM and SurGreedyLLM on dataset Overruling by following the budget B setting. For comparison, we also include one strong single model GPT-4o as a baseline. The results are displayed in Figure 6. As shown in Figure 6a, ThriftLLM and SurGreedyLLM achieve exactly the same accuracy scores, consistent with our result in Proposition 4. Figure 6b presents the comparison between their cost proportions relative to the given budgets. It is worth noting that ThriftLLM achieves a saving of $\sim$ 10%-40% of the allowed budget compared to SurGreedyLLM. Furthermore, as the budget decreases, the cost savings achieved by ThriftLLM compared to SurGreedyLLM become more noticeable. Both ThriftLLM and SurGreedyLLM outperform GPT-4o when the budget is at least 5×10^{-5} USD per query. Overall, GPT-4o requires up to $\sim$30$\times$ higher cost to achieve only a marginal improvement over ThriftLLM and SurGreedyLLM, as shown in Figure 6b.

Sensitivity to size of historical data. In text classification, 80% of the dataset is used as historical data for success probability estimation. To evaluate the sensitivity of ThriftLLM to the size of historical data, we select {20%, 40%, 60%, 80%} of the original historical data on Overruling respectively for probability estimation, and then evaluate the performance of ThriftLLM on test queries by varying the budget $B = \{1.0, 5.0, 10, 50, 100\} \times 10^{-5}$ (see Table 7). The performance of ThriftLLM is stable and robust relative to the proportion of available historical data. The stability is further enhanced with increased budget allocations. These results imply that ThriftLLM consistently performs well across a wide range of sizes of available historical data.

7 CONCLUSION

We investigate the problem of finding an LLM ensemble under budget constraints for optimal query performance, with a focus on classification queries. We formalize this problem as the Optimal Ensemble Selection problem. To solve this problem, we design a new aggregation scheme for combining individual LLM responses and devise a notion of correctness probability to measure the aggregation quality. We prove that correctness probability is non-decreasing and non-submodular. Despite this, we develop ThriftLLM, a surrogate greedy algorithm that offers an instance dependent approximation guarantee. We evaluate ThriftLLM on several real-world datasets on text classification and entity matching queries. Our experiments show that it achieves state-of-the-art performance while utilizing a relatively small budget compared to the baselines tested. Extensions of ThriftLLM to regression and generation tasks are intriguing directions for future work.

ACKNOWLEDGMENTS

This research was supported by the Ministry of Education, Singapore, under its MOE AcRF TIER 3 Grant (MOE-MOET32022-0001). Lakshmanan's research was supported by an NSERC grant (RGPIN-2020-05408) and an NSERC Alliance Grant (ALLRP 599949 – 24).

A APPENDIX: PROOFS

Proof Sketch of Lemma 1. Part (i) is trivial. For part (ii), consider a random query $q \in Q$ with ground-truth class C_q, any LLM set $\mathcal{S} \subset \mathcal{L}$, and model $l \in \mathcal{L} \setminus \mathcal{S}$. Let $\mathcal{S}' := \mathcal{S} \cup \{l\}$. We can prove that $\xi(\mathcal{S}') \geq \xi(\mathcal{S})$. Part (ii) will follow from this. □

Proof of Lemma 2. We construct a counterexample to demonstrate that the function $\xi(\mathcal{S})$ is not submodular. Consider sets $\mathcal{S} = \{l_1\}$, $\mathcal{T} = \{l_1, l_2\}$, and a LLM l_3. W.l.o.g., we assume their success probabilities follow the partial ranking of $p_1 > p_2, p_1 > p_3$, and $\frac{p_2(K-1)}{1-p_2}\frac{p_3(K-1)}{1-p_3} > \frac{p_1(K-1)}{1-p_1}$, i.e., $\frac{p_2 p_3 (K-1)}{(1-p_2)(1-p_3)} > \frac{p_1}{1-p_1}$. As $\frac{p}{1-p} \in (0, \infty)$ for $p \in (0,1)$ and $\frac{p_1}{1-p_1} > \frac{p_2}{1-p_2}$ and $\frac{p_1}{1-p_1} > \frac{p_3}{1-p_3}$ hold simultaneously, such p_1, p_2, p_3 always exist when fixing K. If set function $\xi(\cdot)$ is submodular, it should satisfy

$$\xi(\mathcal{S} \cup \{l_3\}) - \xi(\mathcal{S}) \geq \xi(\mathcal{T} \cup \{l_3\}) - \xi(\mathcal{T}). \tag{8}$$

According to Proposition 2, we have $\xi(\mathcal{S}) = \xi(\mathcal{S} \cup \{l_3\}) = \xi(\mathbf{T}) = p_1$. For $\xi(\mathcal{T} \cup \{l_3\})$, since $p_1 > p_2, p_1 > p_3$, and $\frac{p_2(K-1)}{1-p_2}\frac{p_3(K-1)}{1-p_3} > \frac{p_1(K-1)}{1-p_1}$ hold, the prediction accuracy $\xi(\mathcal{T} \cup \{l_3\})$ is the total probability of two cases: (i) l_1 makes the correct prediction while $\{l_2, l_3\}$ fail to concur on the same incorrect prediction, and (ii) l_1 makes the incorrect prediction while l_2 and l_3 both predict correctly. Therefore, $\xi(\mathcal{T} \cup \{l_3\})$ is calculated as $\xi(\mathcal{T} \cup \{l_3\}) = p_1 - p_1(1-p_2)\frac{1-p_3}{K-1} + (1-p_1)p_2 p_3$, where the two terms in the sum correspond to the two cases. It can be shown from this that $\xi(\mathcal{T} \cup \{l_3\}) > p_1 = \xi(\mathcal{T})$, violating Equation (8). □

Proof of Theorem 3. Khuller et al. [29] prove that the modified greedy strategy yields a solution with a $(1-\frac{1}{\sqrt{e}})$-approximation guarantee, i.e., $\max\{\gamma(\mathcal{S}_2), p^*\} \geq (1-\frac{1}{\sqrt{e}})\gamma(\mathcal{S}_\gamma^\circ)$ where $\mathcal{S}_\gamma^\circ$ is the optimal solution for the budgeted submodular maximization with set function $\gamma(\cdot)$. Consequently, we have

$$\frac{\xi(\mathcal{S}^*)}{\xi(\mathcal{S}^\circ)} \geq \frac{\max\{\xi(S_1), \xi(S_2), p^*\}}{\gamma(\mathcal{S}^\circ)} \geq \frac{\max\{\xi(S_1), \xi(S_2), p^*\}}{\gamma(\mathcal{S}_\gamma^\circ)} = \frac{\max\{\xi(S_1), \xi(S_2), p^*\}}{\max\{\gamma(\mathcal{S}_2), p^*\}} \frac{\max\{\gamma(\mathcal{S}_2), p^*\}}{\gamma(\mathcal{S}_\gamma^\circ)} \geq \frac{\max\{\xi(S_1), \xi(S_2), p^*\}}{\max\{\gamma(\mathcal{S}_2), p^*\}}(1-\frac{1}{\sqrt{e}}),$$

which completes the proof. □

Proof of Proposition 4. Let C_* be the class with $H_1(\phi_\mathcal{S})$, which implies $h(C_* \mid \phi_\mathcal{S}) = \prod_{l_i \in \mathcal{S}(C_*)} \frac{p_i(K-1)}{1-p_i}$. The potential belief $F(\mathcal{T}^*) = \prod_{l_i \in \mathcal{T}^*} \frac{p_i(K-1)}{1-p_i}$ is the highest possible belief that the remaining LLMs in $\mathcal{T}^*$ can contribute to any class. If $F(\mathcal{T}^*)H_2(\phi_\mathcal{S}) \leq H_1(\phi_\mathcal{S})$ holds, we have

$$F(\mathcal{T}^*)H_K(\phi_\mathcal{S}) \leq \cdots \leq F(\mathcal{T}^*)H_3(\phi_\mathcal{S}) \leq F(\mathcal{T}^*)H_2(\phi_\mathcal{S}) \leq H_1(\phi_\mathcal{S}).$$

This inequality implies that the remaining models in $\mathcal{T}^*$ are not able to contribute the belief to any class except C_* so as to achieve a belief higher than $H_1(\phi_\mathcal{S})$. Therefore, Proposition 4 holds. □

Proof of Theorem 5. By Lemma 4, the prediction accuracy $\xi(\mathcal{S})$ of each inspected $\mathcal{S} \subseteq \mathcal{L}$ is estimated within error $\frac{\epsilon p^*}{2}$ with at least $1 - \frac{\delta}{L^2}$ probability. Let $\mathcal{S}^*$ be the set returned from SurGreedyLLM with θ Monte Carlo simulations of $\xi(\cdot)$. Therefore, for $\mathcal{S}^* = \arg\max\{p^*, \tilde{\xi}(\mathcal{S}_1), \tilde{\xi}(\mathcal{S}_2)\}$, it holds that $|\xi(\mathcal{S}^*) - \tilde{\xi}(\mathcal{S}^*)| \leq \frac{\epsilon p^*}{2}$ with high probability. Since the set function $\gamma(\cdot)$ can be exactly computed in linear time, $\max\{\gamma(\mathcal{S}_2), p^*\} \geq (1-\frac{1}{\sqrt{e}})\gamma(\mathcal{S}_\gamma^\circ)$ holds without involving estimation errors. As a consequence, we have

$$\frac{\xi(\mathcal{S}^*)}{\xi(\mathcal{S}^\circ)} \geq \frac{\tilde{\xi}(\mathcal{S}^*) - \epsilon p^*/2}{\xi(\mathcal{S}^\circ)} \geq \frac{\max\{\xi(S_1), \xi(S_2), p^*\} - \epsilon p^*}{\gamma(\mathcal{S}^\circ)} \geq \frac{\max\{\xi(S_1), \xi(S_2), p^*\} - \epsilon p^*}{\gamma(\mathcal{S}_\gamma^\circ)} = \frac{\max\{\xi(S_1), \xi(S_2), p^*\} - \epsilon p^*}{\max\{\gamma(\mathcal{S}_2), p^*\}} \frac{\max\{\gamma(\mathcal{S}_2), p^*\}}{\gamma(\mathcal{S}_\gamma^\circ)} \geq \frac{\max\{\xi(S_1), \xi(S_2), p^*\} - \epsilon p^*}{\max\{\gamma(\mathcal{S}_2), p^*\}} \frac{\sqrt{e}-1}{\sqrt{e}} \geq (\frac{\max\{\xi(S_1), \xi(S_2), p^*\}}{\max\{\gamma(\mathcal{S}_2), p^*\}} - \epsilon)(1-\frac{1}{\sqrt{e}}).$$

Considering that at most L^2 possible subsets are checked in GreedyLLM, the failure probability is bounded by union bound $\frac{\delta}{L^2} \cdot L^2 = \delta$. Therefore, it holds that

$$\Pr\left[\xi(\mathcal{S}^*) \geq (\frac{\max\{\xi(S_1), \xi(S_2), p^*\}}{\max\{\gamma(\mathcal{S}_2), p^*\}} - \epsilon)(1-\frac{1}{\sqrt{e}}) \cdot \xi(\mathcal{S}^\circ)\right] \geq 1 - \delta,$$

which completes the proof. □

Proof of Theorem 6. By taking $\mathcal{P}_{\text{low}}$, $\hat{\mathcal{P}}$, and $\mathcal{P}_{\text{up}}$ as inputs to ThriftLLM, let $\xi_l(\mathcal{S}_l^\circ)$, $\xi(\mathcal{S}^\circ)$, and $\xi_u(\mathcal{S}_u^\circ)$ be the accuracy scores of the corresponding optimal sets respectively, and $\xi_l(\mathcal{S}_l^*)$, $\xi(\mathcal{S}^*)$, and $\xi_u(\mathcal{S}_u^*)$ be the accuracy scores of the selected sets for $\mathcal{P}_{\text{low}}$, $\hat{\mathcal{P}}$, and $\mathcal{P}_{\text{up}}$ respectively. According to Theorem 5, we have

$$\Pr\left[\frac{\xi(\mathcal{S}^*)}{\xi(\mathcal{S}^\circ)} \geq (\frac{\max\{\xi(S_1), \xi(S_2), p^*\}}{\max\{\gamma(\mathcal{S}_2), p^*\}} - \epsilon)(1-\frac{1}{\sqrt{e}})\right] \geq 1 - \delta.$$

$$\Pr\left[\frac{\xi_u(\mathcal{S}_u^*)}{\xi_u(\mathcal{S}_u^\circ)} \geq (\frac{\max\{\xi_u(S_{u1}), \xi_u(S_{u2}), p_u^*\}}{\max\{\gamma_u(\mathcal{S}_{u2}), p_u^*\}} - \epsilon)(1-\frac{1}{\sqrt{e}})\right] \geq 1 - \delta,$$

where $\gamma_u(\cdot)$ is the surrogate set function, S_{u1} and S_{u2} are selected by SurGreedyLLM on $\mathcal{P}_{\text{up}}$, respectively. Therefore, we have

$$\frac{\xi(\mathcal{S}^*)}{\xi(\mathcal{S}^\circ)} \geq \frac{\xi_l(\mathcal{S}_l^*)}{\xi_u(\mathcal{S}_u^\circ)} \geq \frac{\xi_l(\mathcal{S}_l^*)}{\xi_u(\mathcal{S}_u^*)}(\frac{\max\{\xi_u(S_{u1}), \xi_u(S_{u2}), p_u^*\}}{\max\{\gamma_u(\mathcal{S}_{u2}), p_u^*\}} - \epsilon)(1-\frac{1}{\sqrt{e}}).$$

Meanwhile, as proved in Theorem 5, there are at most L^2 possible subsets for accuracy estimation. Therefore, each model is involved in estimation at most L^2 times. Therefore, the resulting failure probability is at most $\delta + L^2 \sum_{l=1}^{L} \delta_l$. Consequently, we have

$$\Pr\left[\frac{\xi(\mathcal{S}^*)}{\xi(\mathcal{S}^\circ)} \geq \frac{\xi_l(\mathcal{S}_l^*)}{\xi_u(\mathcal{S}_u^*)}(\frac{\max\{\xi_u(S_{u1}), \xi_u(S_{u2}), p_u^*\}}{\max\{\gamma_u(\mathcal{S}_{u2}), p_u^*\}} - \epsilon)(1-\frac{1}{\sqrt{e}})\right] \geq 1 - (\delta + L^2 \sum_{l=1}^{L} \delta_l).$$

□

Proof of Lemma 5. Let $X_1, \cdots, X_{\Lambda_l} \in \{0,1\}$ be random variables such that $X_i = 1$ if $p_l^{(i)\perp} \leq p_l \leq p_l^{(i)\top}$ holds where $[p_l^{(i)\perp}, p_l^{(i)\top}]$ is the confidence interval obtained in the i-th repetition; otherwise $X_i = 0$. Let $X = \frac{1}{\Lambda_l}\sum_{i=1}^{\Lambda_l} X_i$ be the average. Therefore, we have $\mathbb{E}[X] \geq 1 - \delta_l$ as $X_i = 1$ holds with at least $1 - \delta_l$ probability. Let the value centered at the interval $[p_l^{(i)\perp}, p_l^{(i)\top}]$ be the corresponding estimation. After algorithm $\mathcal{A}$ is repeated Λ_l times, let $[\bar{p}_l^\perp, \bar{p}_l^\top]$ be the confidence interval whose estimation is the median value among obtained estimations. The true probability p_l does not belong to $[\bar{p}_l^\perp, \bar{p}_l^\top]$ if and only if at least half of the estimations fail, i.e., $X_i = 0$. In this regard, we have

$$Pr[p_l \notin [\bar{p}_l^\perp, \bar{p}_l^\top]] \leq \Pr[X \leq \tfrac{1}{2}] \leq \Pr[X \leq \mathbb{E}[X] - \tfrac{1-2\delta_l}{2}]$$
$$\leq \exp(-2\Lambda_l(\tfrac{1-2\delta_l}{2})^2) = \exp(-\tfrac{\Lambda_l(1-2\delta_l)^2}{2}),$$

where the second inequality is due to the fact that $\mathbb{E}[X] - \frac{1-2\delta_l}{2} \geq \frac{1}{2}$ as $\mathbb{E}[X] \geq 1 - \delta_l$, and last one is due to Hoeffding's inequality [25]. □

REFERENCES

[1] Anthropic. 2024. Introducing the next generation of Claude. https://www.anthropic.com/news/claude-3-family Accessed on April 26, 2024.

[2] Rachit Bansal, Bidisha Samanta, Siddharth Dalmia, Nitish Gupta, Sriram Ganapathy, Abhishek Bapna, Prateek Jain, and Partha Talukdar. 2024. LLM Augmented LLMs: Expanding Capabilities through Composition. In *ICLR*.

[3] Nils Barlaug and Jon Atle Gulla. 2021. Neural networks for entity matching: A survey. *TKDD* 15, 3 (2021), 1–37.

[4] Andrew An Bian, Joachim M. Buhmann, Andreas Krause, and Sebastian Tschiatschek. 2017. Guarantees for Greedy Maximization of Non-submodular Functions with Applications. In *ICML*, Vol. 70. 498–507.

[5] Stéphane Boucheron, Gábor Lugosi, and Olivier Bousquet. 2003. Concentration Inequalities. In *Advanced Lectures on Machine Learning*, Vol. 3176. 208–240.

[6] Niv Buchbinder, Moran Feldman, Joseph Naor, and Roy Schwartz. 2012. A Tight Linear Time (1/2)-Approximation for Unconstrained Submodular Maximization. In *FOCS*. 649–658.

[7] Riccardo Cappuzzo, Paolo Papotti, and Saravanan Thirumuruganathan. 2020. Creating embeddings of heterogeneous relational datasets for data integration tasks. In *SIGMOD*. 1335–1349.

[8] Iñigo Casanueva, Tadas Temcinas, Daniela Gerz, Matthew Henderson, and Ivan Vulic. 2020. Efficient Intent Detection with Dual Sentence Encoders. In *Proceedings of the 2nd Workshop on NLP for ConvAI - ACL 2020*.

[9] Lihu Chen and Gaël Varoquaux. 2024. What is the Role of Small Models in the LLM Era: A Survey. *arXiv preprint arXiv:2409.06857* (2024).

[10] Lingjiao Chen, Matei Zaharia, and James Zou. 2023. FrugalGPT: How to Use Large Language Models While Reducing Cost and Improving Performance. *CoRR* abs/2305.05176 (2023).

[11] Wei Chen and Zhiyuan Li. 2024. Octopus v2: On-device language model for super agent. *CoRR* abs/2404.01744 (2024).

[12] Wei Chen and Zhiyuan Li. 2024. Octopus v4: Graph of language models. *CoRR* abs/2404.19296 (2024).

[13] Xiangxiang Dai, Jin Li, Xutong Liu, Anqi Yu, and John C. S. Lui. 2024. Cost-Effective Online Multi-LLM Selection with Versatile Reward Models. *CoRR* abs/2405.16587 (2024).

[14] Dujian Ding, Ankur Mallick, Chi Wang, Robert Sim, Subhabrata Mukherjee, Victor Rühle, Laks V. S. Lakshmanan, and Ahmed Hassan Awadallah. 2024. Hybrid LLM: Cost-Efficient and Quality-Aware Query Routing. In *ICLR*.

[15] Muhammad Ebraheem, Saravanan Thirumuruganathan, Shafiq Joty, Mourad Ouzzani, and Nan Tang. 2018. Distributed Representations of Tuples for Entity Resolution. *VLDB* 11, 11 (2018), 1454–1467.

[16] Ju Fan, Zihui Gu, Songyue Zhang, Yuxin Zhang, Zui Chen, Lei Cao, Guoliang Li, Samuel Madden, Xiaoyong Du, and Nan Tang. 2024. Combining Small Language Models and Large Language Models for Zero-Shot NL2SQL. *VLDB* 17, 11 (2024), 2750–2763.

[17] Chenhao Fang, Xiaohan Li, Zezhong Fan, Jianpeng Xu, Kaushiki Nag, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. 2024. Llm-ensemble: Optimal large language model ensemble method for e-commerce product attribute value extraction. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2910–2914.

[18] Xue-Yong Fu, Md. Tahmid Rahman Laskar, Elena Khasanova, Cheng Chen, and Shashi Bhushan TN. 2024. Tiny Titans: Can Smaller Large Language Models Punch Above Their Weight in the Real World for Meeting Summarization?. In *NAACL*. 387–394.

[19] Junhao Gan and Yufei Tao. 2015. DBSCAN Revisited: Mis-Claim, Un-Fixability, and Approximation. In *SIGMOD*. 519–530.

[20] Dawei Gao, Haibin Wang, Yaliang Li, Xiuyu Sun, Yichen Qian, Bolin Ding, and Jingren Zhou. 2024. Text-to-SQL Empowered by Large Language Models: A Benchmark Evaluation. *VLDB* 17, 5 (2024), 1132–1145.

[21] Rohan Anil Gemini Team, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, and Anja Hauth et al. 2023. Gemini: A Family of Highly Capable Multimodal Models. *CoRR* abs/2312.11805 (2023).

[22] Lise Getoor and Ashwin Machanavajjhala. 2012. Entity resolution: theory, practice & open challenges. *VLDB* 5, 12 (2012), 2018–2019.

[23] help.openai.com. 2024. How much does GPT-4 cost? Retrieved Oct 17, 2024 from https://help.openai.com/en/articles/7127956-how-much-does-gpt-4-cost

[24] Dylan Hillier, Leon Guertler, Cheston Tan, Palaash Agrawal, Chen Ruirui, and Bobby Cheng. 2024. Super Tiny Language Models. *CoRR* abs/2405.14159 (2024).

[25] Wassily Hoeffding. 1963. Probability Inequalities for Sums of Bounded Random Variables. *J. Amer. Statist. Assoc.* (1963), 13–30.

[26] Keke Huang, Jing Tang, Xiaokui Xiao, Aixin Sun, and Andrew Lim. 2020. Efficient Approximation Algorithms for Adaptive Target Profit Maximization. In *ICDE*. 649–660.

[27] Dongfu Jiang, Xiang Ren, and Bill Yuchen Lin. 2023. LLM-Blender: Ensembling Large Language Models with Pairwise Ranking and Generative Fusion. In *ACL*. 14165–14178.

[28] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, Vol. 1.

[29] Samir Khuller, Anna Moss, and Joseph Naor. 1999. The Budgeted Maximum Coverage Problem. *Inf. Process. Lett.* 70, 1 (1999), 39–45.

[30] Pradap Konda, Sanjib Das, Paul Suganthan G. C., AnHai Doan, Adel Ardalan, Jeffrey R. Ballard, Han Li, Fatemah Panahi, Haojun Zhang, Jeff Naughton, Shishir Prasad, Ganesh Krishnan, Rohit Deep, and Vijay Raghavendra. 2016. Magellan: toward building entity matching management systems. *VLDB* 9, 12 (2016), 1197––1208.

[31] Deren Lei, Yaxi Li, Mengya Hu, Mingyu Wang, Vincent Yun, Emily Ching, and Eslam Kamal. 2023. Chain of Natural Language Inference for Reducing Large Language Model Ungrounded Hallucinations. *CoRR* abs/2310.03951 (2023).

[32] Yuliang Li, Jinfeng Li, Yoshihiko Suhara, AnHai Doan, and Wang-Chiew Tan. 2020. Deep entity matching with pre-trained language models. *VLDB* 14, 1 (2020), 50–60.

[33] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692* (2019).

[34] livechatai.com. 2024. Gemini Pro API Pricing Calculator. Retrieved Oct 17, 2024 from https://livechatai.com/gemini-pro-api-pricing-calculator

[35] Silvano Martello and Paolo Toth. 1987. Algorithms for knapsack problems. *North-Holland Mathematics Studies* 132 (1987), 213–257.

[36] Sidharth Mudgal, Han Li, Theodoros Rekatsinas, AnHai Doan, Youngchoon Park, Ganesh Krishnan, Rohit Deep, Esteban Arcaute, and Vijay Raghavendra. 2018. Deep learning for entity matching: A design space exploration. In *SIGMOD*. 19–34.

[37] OpenAI. 2023. GPT-4 Technical Report. *CoRR* abs/2303.08774 (2023).

[38] OpenAI. 2024. OpenAI Embeddings API. https://api.openai.com/v1/embeddings.

[39] Diarmuid O'Reilly-Morgan, Elias Tragos, Erika Duriakova, Honghui Du, Neil Hurley, and Aonghus Lawlor. 2025. Entity Matching with Large Language Models as Weak and Strong Labellers. In *New Trends in Database and Information Systems*. 58–67.

[40] Marcel Parciak, Brecht Vandevoort, Frank Neven, Liesbet M. Peeters, and Stijn Vansummeren. 2024. Schema Matching with Large Language Models: an Experimental Study. *VLDB 2024 Workshop: Tabular Data Analysis Workshop (TaDA)* (2024).

[41] Ralph Peeters, Aaron Steiner, and Christian Bizer. 2025. Entity Matching using Large Language Models. In *EDBT*. 529–541.

[42] Shivanshu Shekhar, Tanishq Dubey, Koyel Mukherjee, Apoorv Saxena, Atharv Tyagi, and Nishanth Kotla. 2024. Towards Optimizing the Costs of LLM Usage. *CoRR* abs/2402.01742 (2024).

[43] Yishuo Shi and Xiaoyan Lai. 2024. Approximation algorithm of maximizing non-monotone non-submodular functions under knapsack constraint. *Theor. Comput. Sci.* 990 (2024), 114409.

[44] Dan Su, Yan Xu, Genta Indra Winata, Peng Xu, Hyeondey Kim, Zihan Liu, and Pascale Fung. 2019. Generalizing Question Answering System with Pre-trained Language Model Fine-tuning. In *EMNLP*. 203–211.

[45] Xiaofei Sun, Xiaoya Li, Jiwei Li, Fei Wu, Shangwei Guo, Tianwei Zhang, and Guoyin Wang. 2023. Text Classification via Large Language Models. In *EMNLP*. 8990–9005.

[46] Yushi Sun, Xin Hao, Kai Sun, Yifan Xu, Xiao Yang, Xin Luna Dong, Nan Tang, and Lei Chen. 2024. Are Large Language Models a Good Replacement of Taxonomies? *VLDB* 17, 11 (2024), 2919–2932.

[47] Yehui Tang, Kai Han, Fangcheng Liu, Yunsheng Ni, Yuchuan Tian, Zheyuan Bai, Yi-Qi Hu, Sichao Liu, Shangling Jui, and Yunhe Wang. 2024. Rethinking Optimization and Architecture for Tiny Language Models. In *ICML*.

[48] Youze Tang, Xiaokui Xiao, and Yanchen Shi. 2014. Influence maximization: near-optimal time complexity meets practical efficiency. In *SIGMOD*. 75–86.

[49] Selim Furkan Tekin, Fatih Ilhan, Tiansheng Huang, Sihao Hu, and Ling Liu. 2024. LLM-TOPLA: Efficient LLM Ensemble by Maximising Diversity. In *Findings of the Association for Computational Linguistics: EMNLP 2024*. Association for Computational Linguistics, 11951–11966.

[50] Immanuel Trummer. 2024. Generating Succinct Descriptions of Database Schemata for Cost-Efficient Prompting of Large Language Models. *VLDB* 17, 11 (2024), 3511–3523.

[51] Mengzhao Wang, Haotian Wu, Xiangyu Ke, Yunjun Gao, Xiaoliang Xu, and Lu Chen. 2024. An Interactive Multi-modal Query Answering System with Retrieval-Augmented Large Language Models. *VLDB* 17, 12 (2024), 4333–4336.

[52] Johannes Welbl, Nelson F. Liu, and Matt Gardner. 2017. Crowdsourcing Multiple Choice Science Questions. In *Workshop on Noisy User-generated Text, NUT@EMNLP*. 94–106.

[53] Yu Xia, Fang Kong, Tong Yu, Liya Guo, Ryan A. Rossi, Sungchul Kim, and Shuai Li. 2024. Which LLM to Play? Convergence-Aware Online Model Selection with Time-Increasing Bandits. In *WWW*. 4059–4070.

[54] Dezhong Yao, Yuhong Gu, Gao Cong, Hai Jin, and Xinqiao Lv. 2022. Entity resolution with hierarchical graph attention networks. In *SIGMOD*. 429–442.

[55] Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a Machine Really Finish Your Sentence?. In *ACL*. 4791–4800.
[56] Chao Zhang, Yuren Mao, Yijiang Fan, Yu Mi, Yunjun Gao, Lu Chen, Dongfang Lou, and Jinshu Lin. 2024. FinSQL: Model-Agnostic LLMs-based Text-to-SQL Framework for Financial Analysis. In *SIGMOD*. 93–105.
[57] Tianyi Zhang, Faisal Ladhak, Esin Durmus, Percy Liang, Kathleen R. McKeown, and Tatsunori B. Hashimoto. 2024. Benchmarking Large Language Models for News Summarization. *ACL* 12 (2024), 39–57.
[58] Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level Convolutional Networks for Text Classification. In *NIPS*. 649–657.
[59] Chen Zhao and Yeye He. 2019. Auto-em: End-to-end fuzzy entity-matching using pre-trained deep models and transfer learning. In *WWW*. 2413–2424.
[60] Lucia Zheng, Neel Guha, Brandon R. Anderson, Peter Henderson, and Daniel E. Ho. 2021. When does pretraining help?: assessing self-supervised learning for law and the CaseHOLD dataset of 53, 000+ legal holdings. In *ICAIL*. 159–168.
[61] Jun-Peng Zhu, Peng Cai, Kai Xu, Li Li, Yishen Sun, Shuai Zhou, Haihuang Su, Liu Tang, and Qi Liu. 2024. AutoTQA: Towards Autonomous Tabular Question Answering through Multi-Agent Large Language Models. *VLDB* 17, 12 (2024), 3920–3933.