Model Performance-Guided Evaluation Data Selection for Effective Prompt Optimization

Ximing Dong[1], Shaowei Wang[2*], Dayi Lin[1], Ahmed E. Hassan[3]
[1]Centre for Software Excellence, Huawei, Canada
[2] Department of Computer Science, University of Manitoba, Canada
[3]School of Computing, Queen's University, Canada
{ximing.dong,dayi.lin}@huawei.com, shaowei.wang@umanitoba.ca, ahmed@cs.queensu.ca

Abstract

Optimizing Large Language Model (LLM) performance requires well-crafted prompts, but manual prompt engineering is labor-intensive and often ineffective. Automated prompt optimization techniques address this challenge but the majority of them rely on randomly selected evaluation subsets, which fail to represent the full dataset, leading to unreliable evaluations and suboptimal prompts. Existing coreset selection methods, designed for LLM benchmarking, are unsuitable for prompt optimization due to challenges in clustering similar samples, high data collection costs, and the unavailability of performance data for new or private datasets. To overcome these issues, we propose IPOMP, an **I**terative evaluation data selection for effective **P**rompt **O**ptimization using real-time **M**odel **P**erformance. IPOMP is a two-stage approach that selects representative and diverse samples using semantic clustering and boundary analysis, followed by iterative refinement with real-time model performance data to replace redundant samples. Evaluations on two datasets BIG-bench and LIAR, and two models GPT-3.5 and GPT-4o-mini, show that IPOMP improves effectiveness by at least 1.6% to 3.1%, and stability by at least 50% to 55.5% compared with the best baseline across the studied datasets and models, with minimal computational overhead below 1%. Furthermore, the results demonstrate that our real-time performance-guided refinement approach can be universally applied to enhance existing coreset selection methods.

1 Introduction

Given a task, drafting an effective prompt is a key part of optimizing the Large Language Model's (LLM) performance (Kojima et al., 2022; Pryzant et al., 2023; Wei et al., 2022). Minor changes in prompt can lead to performance gains or losses, necessitating prompt engineering for effective LLM utilization (Liu et al., 2023). To avoid the process of manually creating prompts, recent work aims to automate the process of generating natural language prompts that are also interpretable (Zhou et al., 2022b; Zhang et al., 2023; Guo et al., 2023; Yang et al., 2024; Zhang et al., 2022; Deng et al., 2022).

All the proposed approaches require optimizing prompts by evaluating them over an evaluation dataset. However, using the entire training dataset is impractical and cost-prohibitive (Pacchiardi et al., 2024; Albalak et al., 2024). Consequently, most approaches randomly select a small subset of samples from the entire training data to evaluate new prompts (Zhou et al., 2022b; Zhang et al., 2023; Guo et al., 2023; Yang et al., 2024; Pryzant et al., 2023). However, random selection often fails to produce representative samples of the entire training dataset, leading to unreliable evaluation results (Zadrozny, 2004) and under-optimized prompts. No existing approaches have been proposed to select representative samples for evaluating prompts for prompt optimization.

Various coreset selection approaches have been developed for benchmarking machine learning models, with the core idea being to select representative samples that effectively represent the entire dataset based on different criteria such as semantics (Sener and Savarese, 2017; Har-Peled and Mazumdar, 2004), model performance indicators such as confidence scores (Pacchiardi et al., 2024; Vivek et al., 2023; Polo et al., 2024), and training errors (Paul et al., 2021). However, these approaches are not well-suited for prompt optimization. In semantics-based approaches, samples for certain tasks (e.g., Navigation task from BIG-bench (bench authors, 2023)) tend to be highly similar, making it challenging to cluster them effectively solely based on their semantics. On the other hand, approaches that leverage model performance information rely on evaluation results from previously tested mod-

* Corresponding author.

Findings of the Association for Computational Linguistics: ACL 2025, pages 2844–2859

els to predict the performance of new models (Pacchiardi et al., 2024; Vivek et al., 2023; Zhou et al., 2023, 2022a). However, such approaches have notable limitations. Firstly, model performance data for training samples is not always available, particularly for new or proprietary datasets. Even if it is possible, collecting the model performance information prior to the prompt optimization process is expensive. Secondly, using past model performance to estimate the capabilities of current LLMs often results in sub-optimal predictions, as model behaviors may vary significantly (which is evidenced by our results in Section 5.1).

To address the limitations of existing coreset selection methods and tailor them for prompt optimization, we propose a two-stage approach called IPOMP that leverages both semantic and model performance information. In the first stage, we identify informative samples by clustering the entire training dataset based on semantic similarity and selecting representative samples from each cluster. Additionally, to enhance diversity, we incorporate boundary cases by selecting the most distant sample pairs in the semantic space. In the second stage, we iteratively refine the evaluation samples by incorporating their real-time model performance during the optimization process. Specifically, we identify redundant samples based on their performance across the generated prompts and replace them with contrasting samples.

We evaluated IPOMP on the BIG-Bench and LIAR datasets, comparing its performance against several SOTA baseline methods. IPOMP outperformed all baselines, achieving effectiveness improvements in terms of Accuracy ranging from 1.6% to 3.1% and significantly enhancing stability by at least 50% in terms of standard deviation, while with a computational overhead of less than 1%. Furthermore, our evaluation results demonstrate that our real-time model performance-guided refinement approach in the second stage can be universally adapted to existing coreset selection approaches to enhance their effectiveness and stability.

2 Background and related work

2.1 Prompt Optimization

Prompt optimization refers to the systematic process of designing, refining, and evaluating prompts to improve the performance of large language models (LLMs) on specific tasks (Zhou et al., 2022b). Let $\mathcal{L}$ denote a large language model, T represent a given task, and $D^{\text{evaluation}} = \{x_i, y_i\}_{i=1}^N$ be the evaluation dataset, where x_i are inputs and y_i are the corresponding desired outputs for T. A prompt P is a sequence of tokens that guides $\mathcal{L}$ to generate outputs $\hat{y}_i = \mathcal{L}(x_i, P)$.

The objective of prompt optimization is to find a prompt P^* that maximizes the task performance over $D^{\text{evaluation}}$. This can be expressed as:

$$P^* = \arg\max_P \mathcal{M}\left(\{\mathcal{L}(x_i, P) y_i\}_{i=1}^N\right),$$

where $\mathcal{M}$ is a performance metric that quantifies the alignment between the model-generated outputs $\hat{y}_i = \mathcal{L}(x_i, P)$ and the ground-truth outputs y_i, such as Accuracy, F1-score, BLEU score, or task-specific measures.

The optimization typically involves iterative refinement of prompt P using various strategies, which typically can be categorized into two families: *non-directional* and *directional*. Non-directional approaches sample or generate new inputs randomly and do not explicitly aim to reduce error on a train set over the optimization iteration based on feedback, such as random search (Zhou et al., 2022b; Zhang et al., 2023) and evolutionary algorithm (Guo et al., 2023; Yang et al., 2024; Fernando et al.). For instance, APE generates semantically similar candidate prompts for a task based on their performance on a training subset and iteratively selects the best prompt(Zhou et al., 2022b). In *Directional* family, the generation of new prompts is guided by the error measure on evaluation data, such as using gradient (Pryzant et al., 2023; Juneja et al., 2024) and reinforcement learning (Zhang et al., 2022; Deng et al., 2022; Yao et al., 2023). For instance, APO uses minibatches of data to form natural language gradients that criticize the current prompt (Pryzant et al., 2023) . The gradients are then propagated into the prompt by editing the prompt in the opposite semantic direction of the gradient.

Most existing prompt optimization approaches either utilize the entire training dataset or randomly sample a subset, which can make the evaluation process overly expensive or suboptimal. To address this challenge, we propose a novel evaluation data selection approach specifically designed for prompt optimization.

2.2 Coreset selection for benchmarking machine learning models

We are the first to propose evaluation data selection in the context of prompt optimization. The most related field to our work is coreset selection. Coreset selection aims to find the most informative subset $D^{\text{Core}} \subset D^{\text{Training}}$ with the constraint $|D^{\text{Core}}| \ll |D^{\text{Training}}|$, so that the model trained on D^{Core} has close generalization performance to the model trained on the whole training set D^{Training}.

Numerous approaches have been developed for coreset selection for evaluating machine learning models in recent years. Geometry-based methods assume that semantically similar data points share properties (Chen et al., 2012; Sener and Savarese, 2017; Sinha et al., 2020; Agarwal et al., 2020). However, solely relying on semantics while overlooking the model performance could lead to suboptimal performance in identifying the representative samples, typically for tasks where the samples are naturally semantically close to each other. To improve accuracy, performance-based approaches consider factors such as confidence (Coleman et al., 2019; Margatina et al., 2021; Lin et al., 2023; Kim et al., 2020), error (Paul et al., 2021; Toneva et al., 2018; Liu et al., 2021). For instance, approaches based on confidence prioritize uncertain samples, assuming they have a greater impact on model performance. Error-based approaches assume that training samples are more important if they contribute more to the error or loss when training models. Decision boundary-based approaches focus on samples near the decision boundary, as they are harder to classify and valuable for coreset selection (Ducoffe and Precioso, 2018; Margatina et al., 2021; Chai et al., 2023). More recently, methods have leveraged evaluation results from previously tested LLMs to predict the performance of new models (Pacchiardi et al., 2024; Vivek et al., 2023; Zhou et al., 2023, 2022a).

We propose a novel two-stage data selection approach for prompt optimization that leverages both semantic and real-time model performance features. Unlike existing model performance-based methods that require a preliminary stage to gather performance data or rely on prior model results to predict new outcomes (Vivek et al., 2023; Pacchiardi et al., 2024; Zhou et al., 2022a), our approach dynamically collects performance data during optimization in real-time, ensuring greater accuracy and cost-efficiency. Additionally, it can be seamlessly integrated into any prompt optimization method involving iterative refinement.

3 Methodology

We propose a two-stage approach that leverages both semantic and model performance features to select evaluation data for effective prompt optimization: 1) Diverse sample selection; and 2) Real-time model performance-guided iterative refinement. In stage 1, we cluster training samples based on semantics and select representative samples from each cluster, while incorporating boundary cases by selecting the most distant pairs in the semantic space. In stage 2, we iteratively refine the selected samples by analyzing real-time model performance during optimization, removing redundant samples, and replacing them with contrasting ones. For simplicity, we use the terms “coreset selection” and “data selection” interchangeably in the following text.

3.1 Stage 1: Diverse sample selection

Algorithm 1 Diverse sample selection

Input: Training set D^{training}, number of selected samples N , number of clustering groups k, portion of samples from semantic clustering α

Output: $D^{evaluation}$ of size N

Select αN samples based on semantic clustering
$S_{\text{clustering}} \leftarrow \{\}$
$clusters \leftarrow \text{KMeans}(D^{\text{training}}, k)$
$S_{\text{clustering}} \leftarrow \text{sampleProportionly}(\alpha N, clusters)$
$D^{\text{training}}.\text{remove}(S_{\text{clustering}})$
Select $(1-\alpha)N$ boundary samples
$n, S_{\text{boundary}} \leftarrow 0, \{\}$
while $n \leq \alpha N$ **do**
    $d_1, d_2 \leftarrow \text{getLeastSimilarPair}(D^{\text{training}})$
    if $d_1 \notin S_{\text{clustering}}$ **then**
        $S_{\text{boundary}}.\text{add}(d_1, d_2)$
        $n \leftarrow n + 2$
    end if
    $D^{\text{training}}.\text{remove}(d_1, d_2)$
end while
Return $D^{\text{evaluation}} \leftarrow S_{\text{clustering}} + S_{\text{boundary}}$

In this stage, we aim to select a small subset from the entire training set that comprehensively represents all training samples. One common strategy is to first cluster samples of the training set based on their selected properties, such as semantic (Sener and Savarese, 2017; Har-Peled and Mazumdar, 2004). Then it samples representatives from each resultant cluster to form a reduced set. However, such clustering-based approaches probably would miss boundary cases (Huang et al., 2024). Therefore, in this stage, we combine semantic clustering and boundary selection methods to select a small yet comprehensive subset of samples from the training set.

Given the training set D^{training} which contains M training samples $\{D_1, D_2, \ldots, D_M\}$, our algorithm outputs a subset $D^{\text{evaluation}}$ with a size of N, where $N \ll M$. We demonstrate the algorithm in Algorithm 1, which consists of two steps: 1) Selecting informative samples using semantic clustering. We first embed each sample in D^{training} into latent space. We utilize Sentence-Bert (Reimers, 2019) to encode semantic representations. We then use K-means to cluster samples into k clusters (Line 3). Note that K-means is selected due to its effectiveness and efficiency. We do not use methods to determine the value of k since they typically require additional time. In addition, throughout experiments, we find that the value of k does not impact our data selection approach significantly (see more results in Appendix A.5). Lastly, we randomly select samples proportionally from each cluster based on their size and totally select αN samples (i.e., $S_{clustering}$), where α is the portion of selected samples from semantic clustering (Line 4). 2) Identifying boundary cases. Inspired by prior study (Huang et al., 2024), we select boundary cases by finding samples that are least similar to each other. To do so, similar to step 1, we embed the samples into semantic latent space and find the pairs of samples having the furthest distance, iteratively (Lines 8 - 14). Note that we only include the samples that are not included in $S_{\text{clustering}}$. Calculating the distance among all samples is expensive, with a time complexity of $O(dN^2)$, where d is the dimension of embeddings and N is the size of dataset. To improve the efficiency, we first detect the boundary points in the latent space by following previous study (Angiulli and Pizzuti, 2002), and then find the furthest pairs among those boundary points. Finally, we combine the clustering samples $S_{\text{clustering}}$ and boundary samples S_{boundary} together as the final subset of samples.

Algorithm 2 Real-time model performance-guided iterative refinement.

Input: Selected samples from stage 1 $D^{\text{evaluation}}$, Replace rate β, Training set D^{training}, Correlation threshold CT, Prompt optimization approach $\mathcal{PO}$, LLM $\mathcal{L}$, number of iterations I
Output: Refined samples $D^{\text{evaluation}}_{\text{refined}}$, best prompts $bestPrompt$

$i, S_i \leftarrow 0, D^{\text{evaluation}}$
while $i < I$ **do**
  $candP \leftarrow \mathcal{PO}$.updatePrompts$(i, candP)$
  $MP_{\text{runtime}} \leftarrow$ recordPerf$(S_i, \mathcal{L}, candP)$
  #Identify redundant samples in S_i based on their model performance
  $clusters \leftarrow$ Clustering$(MP_{\text{runtime}}, CT)$
  $S_{redundant} \leftarrow$ sampleRedundant$(clusters, \beta)$
  # Find least samples to replace
  for $e_i \in S_{redundant}$ **do**
    $d \leftarrow$ leastSimSample$(e_i, D^{\text{training}})$
    $S_{redundant}$.replace(e_i, d)
  end for
end while
Return $D^{\text{evaluation}}_{\text{refined}} \leftarrow S_i$, $bestPrompt \leftarrow identifyBest(candP)$

3.2 Stage 2: Real-time model performance-guided iterative refinement

We obtain a reduced set $D^{\text{evaluation}}$ based on their semantics after applying Algorithm 1. However, solely relying on semantics while overlooking the model performance could lead to sub-optimal performance, typically for tasks where the samples are naturally semantically close to each other. To address this, we design a novel algorithm called Real-time model performance-guided iterative refinement, which updates $D^{\text{evaluation}}$ iteratively by replacing redundant samples with contrasting ones based on their model performance. This process leverages real-time model performance observed during prompt optimization, eliminating the need for pre-collected performance data from existing models or preliminary evaluations.

Our approach is inspired by a key observation that a significant portion of samples exhibit high correlations in their model performance across prompts during prompt optimization. Figure 1 presents a heatmap where each cell represents the correlation between two samples based on their real-time model performance (i.e., logits in this case). As we can see, a substantial portion of samples (20% in this case) exhibit a correlation greater than 0.9 with others. This indicates redundancy among these samples and they can be replaced with alternative samples from the training set to enhance the diversity of $D^{\text{evaluation}}$.

We demonstrate our algorithm in Algorithm 2. Given a prompt optimization technique $\mathcal{PO}$, for each iteration, our algorithm first identifies the redundant samples in the $D^{\text{evaluation}}$ (Lines 3 - 7) and replaces a certain portion of them with the opposite (i.e., the most dissimilar) ones retrieved from the training set. To be specific, for each iteration, we record the performance of each example's performance across candidate prompts on the LLM $\mathcal{L}$ (line 4). The performance matrix $MP_{runtime} \in \mathbb{R}^{|S_i| \times (|output| \times |candP|)}$ where $|S_i|$ is the number of samples in S_i, $|output|$ is the size of output labels, and $|candP|$ is the number of candidate prompts $candP$ generated by prompt optimization approach $\mathcal{PO}$ in each iteration. We use logits as the proxy of the model's confidence to construct the performance matrix. For instance, if the output labels of a task are True/False, the matrix includes two dimensions to represent the performance: $\{Logit(True), Logit(False)\}$. If the output is True, the performance is $\{Logit(True), 0\}$. In cases where the output is neither True nor False, the performance is set to $\{0, 0\}$. To identify the redundant samples, we use a hierarchical cluster to build clusters and group samples that share high correlations in the same cluster (Line 6) by following prior studies (Wang et al., 2018; Rajbahadur et al., 2017). In our case, we set the threshold CT to 0.9 (i.e., highly correlated). We then randomly select a portion (i.e., β) of samples $S_{redundant}$ from each cluster with highly correlated samples and replace them. For replacement, we iteratively select samples from D^{training} which have the lowest semantic similarity for examples in $S_{redundant}$ (Line 9 - 12). We assume that the pair with the lowest semantic similarity is more likely to yield answers, which leads to different model performances. For efficient search, we use Hierarchical Navigable Small World (HNSW) (Malkov and Yashunin, 2018) to perform an approximate search by inverting its similarity function to calculate the dissimilarity between the query examples and examples in the training data. HNSW is efficient in high-dimension space.

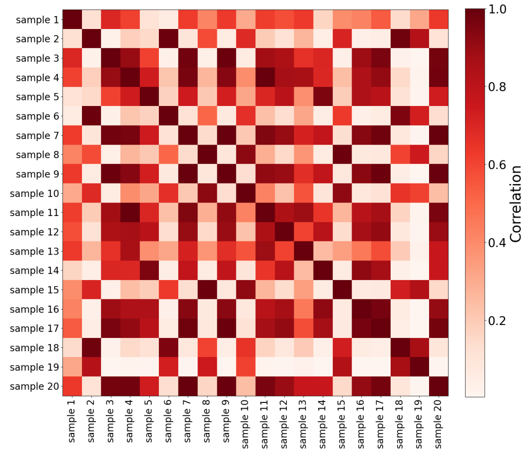

Figure 1: Correlation among the samples selected in stage 1 based on their real-time performance across candidate prompts during the initial iteration of APE. Each cell represents the correlation between a pair of samples.

4 Experimental Setting

4.1 Datasets

In this paper, we evaluated our approach on two datasets **BIG-bench** dataset (bench authors, 2023) and **LIAR** (Wang, 2017). We selected the following five tasks from Big-bench. **Presuppositions as Natural Language Inference (NLI)**, where to reason whether a presupposition is embedded in a given statement and output entailment, neutral, or contradiction. **Navigation**: Given a sequence of navigational instructions, the objective is to ascertain if an agent would return to the original point of departure. **Implicatures**: This task requires models to determine whether a response given by one speaker to another constitutes an affirmative or negative reply. **Metaphor Understanding**: This task presents a model with a metaphoric sentence and requires it to identify whether a subsequent sentence accurately interprets the initial metaphor. **Sports Understanding**: This task requires models to assess whether a synthetically constructed sentence related to a sport is plausible or not. We also evaluated **LIAR** (Wang, 2017), a widely used public dataset for **Fake News Detection**.

We measure the **Accuracy** of those classification tasks to evaluate the effectiveness of the studied coreset selection approaches. In addition, we use the **standard deviation** (SD) to measure the stability of the approaches.

4.2 Prompt optimization approaches and base LLMs

As discussed in Section 2, prompt optimization can primarily be categorized into two families, directional and non-directional. We selected two state-of-the-art approaches APO (Pryzant et al., 2023) and EVOPROMPT (Guo et al., 2023) from non-directional family, and one of the most commonly used approaches APE (Zhou et al., 2022b) from directional family. We use their default setting in our experiments and choose GPT-3.5 and GPT-4o-mini.

4.3 Baselines

To evaluate the effectiveness of our approach, we compare our proposed approach with various baselines. **Random.** Randomly sample examples from the training set. **Clustering.** Selecting examples proportionally from the clusters that are constructed based on samples' semantics as discussed in Section 3.1. **Boundary.** We select boundary cases as discussed in Section 3.1 by following prior study (Huang et al., 2024). Our work is the first to address evaluation data selection for prompt optimization, and no existing state-of-the-art (SOTA) approaches are available. Therefore, we benchmark our approach against two SOTA coreset selection methods designed for LLM evaluation. **Anchor-Point** (Vivek et al., 2023). This method clusters examples based on the model's confidence in the correct class and selects representative examples, called "anchor point" as the coreset. To adapt it for prompt optimization, we collect the model's confidence scores by running the training dataset through a set of 10 prompts generated via prompt optimization in a preliminary stage. **Prediction-based** (Pacchiardi et al., 2024). This method predicts an instance's performance on an LLM by training a generic assessor on existing LLM performance data. We adapt this approach by training the assessor on our dataset, BIG-bench, using GPT-3.5, following (Pacchiardi et al., 2024). The trained assessor predicts performance on a set of initial prompts, and examples are clustered based on these predictions, similar to Anchor-Point. See more details in Appendix A.2. Note that we do not consider the entire training data as a baseline, since it is too expensive and infeasible in practice.

4.4 Implementation Details

In our experiments, the size of $D^{\text{evaluation}}$ is set to 20, and the number of clustering groups K is set to five by default unless otherwise specified. For IPOMP, we set α to 0.5, meaning that half of the samples in $D^{\text{evaluation}}$ come from the boundary method, while the remaining half comes from the clustering method. The correlation threshold CT and the replace rate β are set to 0.9 and 0.5, respectively. We run each task of each method five times and report the average performance.

5 Results

5.1 Effectiveness and stability

All coreset selection approaches enhance the effectiveness and stability of prompt optimization techniques compared to Random selection. Among these, IPOMP demonstrates superior performance. Table 1 presents the effectiveness and stability of all studied coreset selection approaches, across different prompt optimization techniques. Comparing Random with all other data sampling approaches, including IPOMP, we observe a significantly superior performance across all prompt optimization techniques, which indicates that selecting representative samples is important for prompt optimization. IPOMP improves the best baseline (Anchor-Point) by at least 1.6% to 3.1% across the studied datasets and models. Typically for APE, the improvement gained from IPOMP is larger than other prompt optimization techniques, which is probably attributed to its nature, where in each iteration, the prompt is optimized based on the evaluation (e.g., gradient) from the last iteration, the evaluation data's quality is typically important for the success of the prompt optimization. On the other hand, we summarize the standard deviation (SD) across all datasets for each selected baseline. We find that IPOMP exhibits greater stability than other baselines, achieving the lowest standard deviation across prompt optimization techniques, with an improvement of at least 50%. This is typically attributed to our real-time model performance-guided iterative refinement strategy (i.e., stage 2 of IPOMP), which dynamically refines the evaluation data during runtime and guarantees the stability of our approach. See more ablation analysis in Section 5.2.

Boundary and Clustering share similar effectiveness across all studied prompt optimization techniques. However, compared with IPOMP, it

Table 1: Comparison of the effectiveness (Accuracy) and stability (SD) of the studied prompt optimization approaches with different evaluation data selection approaches.

	GPT-3.5 - BIG-bench						**GPT-4o-mini - BIG-bench**					
	Random	**Boundary**	**Clustering**	**Anchor-Point**	**Prediction-based**	**IPOMP**	**Random**	**Boundary**	**Clustering**	**Anchor-Point**	**Prediction-based**	**IPOMP**
EVOPROMPT	0.743±0.028	0.757±0.022	0.759±0.031	0.774±0.028	0.758±0.025	**0.776**↑**±0.017**↓	0.694 ± 0.047	0.715 ± 0.042	0.706 ± 0.037	0.756 ± 0.026	0.709 ± 0.047	**0.758**↑**±0.011**↓
APO	0.691±0.040	0.718±0.052	0.723±0.025	0.750±0.020	0.743±0.042	**0.753**↑**±0.009**↓	0.703 ±0.038	0.72± 0.043	0.701 ± 0.022	0.743± 0.032	0.690 ±0.043	**0.780**↑**±0.012**↓
APE	0.722±0.037	0.708±0.045	0.684±0.032	0.727±0.035	0.707±0.048	**0.742**↑**±0.010**↓	0.717 ± 0.040	0.734 ± 0.030	0.770 ± 0.030	0.770 ± 0.023	0.716 ± 0.042	**0.794**↑**±0.012**↓
Average	0.719±0.035	0.727±0.040	0.725±0.029	0.745±0.028	0.725±0.038	**0.757**↑**±0.012**↓	0.704± 0.041	0.723 ± 0.038	0.725 ± 0.029	0.756 ± 0.027	0.705 ± 0.044	**0.778**↑**±0.011**↓
	GPT-3.5 - LIAR						**GPT-4o-mini - LIAR**					
	Random	**Boundary**	**Clustering**	**Anchor-Point**	**Prediction-based**	**IPOMP**	**Random**	**Boundary**	**Clustering**	**Anchor-Point**	**Prediction-based**	**IPOMP**
EVOPROMPT	0.753 ± 0.042	0.798 ± 0.037	0.772 ± 0.035	0.810 ± 0.030	0.734 ± 0.043	**0.818**↑**±0.015**↓	0.756 ± 0.045	0.806 ± 0.041	0.782 ± 0.037	0.816 ± 0.026	0.754 ± 0.043	**0.838**↑**±0.012**↓
APO	0.732 ± 0.039	0.792 ±0.032	0.731 ± 0.031	0.794 ± 0.028	0.754 ± 0.038	**0.812**↑**±0.014**↓	0.746 ± 0.059	0.792 ± 0.061	0.776 ± 0.037	0.806 ± 0.022	0.754 ± 0.048	**0.836**↑**±0.013**↓
APE	0.743 ± 0.043	0.721 ± 0.035	0.753 ± 0.042	0.801 ± 0.023	0.748 ± 0.037	**0.832**↑**±0.011**↓	0.746 ± 0.04	0.792 ± 0.039	0.808 ± 0.042	0.8 ± 0.025	0.742 ± 0.04	**0.826**↑**±0.011**↓
Average	0.742 ± 0.041	0.770 ± 0.035	0.752 ± 0.036	0.801 ± 0.027	0.746 ± 0.039	**0.820**↑**±0.012**↓	0.748 ± 0.048	0.797 ± 0.047	0.788 ± 0.038	0.807 ± 0.024	0.75 ± 0.043	**0.833**↑**±0.012**↓

demonstrates lower effectiveness and suffers lower stability, which indicates that relying solely on semantics to obtain samples is insufficient, and incorporating model performance offers a promising approach to identifying representative examples. On the other hand, Prediction-based requires adaptation to new datasets and exhibits lower effectiveness compared to IPOMP, making it limited compatibility in the context of prompt optimization.

Anchor-Point consistently ranks as the second-best approach across all prompt optimization methods. Notably, approaches utilizing real-time model performance data (IPOMP and Anchor-Point) outperform those relying solely on semantics or prior model data, as performance feedback offers more precise insights for distinguishing samples. IPOMP surpasses Anchor-Point by combining semantic and real-time performance data, meanwhile achieving better efficiency. Unlike Anchor-Point, which requires a costly preliminary stage to collect model confidence, IPOMP gathers performance data in real-time during the optimization process.

5.2 Ablation Analysis

IPOMP$_{Stage1}$ vs. IPOMP To evaluate the contribution of model performance-guided iterative refinement (i.e., IPOMP$_{Stage2}$), we conduct an ablation analysis. We compare IPOMP with its variant in which sage 2 is removed (i.e., IPOMP$_{Stage1}$). As shown in Table 2, on average, without stage 2, the effectiveness of IPOMP drops 2.4%. In addition, the performance becomes unstable without stage 2. Specifically, the standard deviation increases by a factor of 2.83, indicating that stage 2 significantly enhances stability. Actually stage 2 indeed reduce the redundant samples. For instance, in APO, after the first round of refinement of stage 2, the redundancy of examples (correlation > 0.9) are significantly reduced from 19% to 10% (see Appendix A.8).

IPOMP$_{Random}$ vs. IPOMP To evaluate the importance of the diverse sample selection of IPOMP (i.e., IPOMP$_{Stage1}$), we construct a variant, namely IPOMP$_{Random}$, where we replace stage 1 with random sampling and keep the rest of IPOMP unchanged. As shown in Table 2, the accuracy of IPOMP$_{Random}$ is substantially 2% lower than IPOMP, illustrating the necessity of diverse data selection at the beginning. In summary, both stages of IPOMP make significant contributions to the effectiveness and stability of IPOMP.

Baseline vs. Baseline+IPOMP$_{Stage2}$ Besides enhancing IPOMP, our real-time model performance-guided iterative refinement can serve as a versatile plugin alongside any data sampling approach to refine evaluation data during runtime. To assess its effectiveness, we apply it to all selected baselines. Figure 2 illustrates the performance of these baselines before and after incorporating IPOMP$_{Stage2}$. The performance of all baselines improves after applying our real-time refinement component (i.e., IPOMP$_{Stage2}$), except for Anchor-Point in some cases. For instance, on average, IPOMP$_{Stage2}$ improves the performance of Random, Boundary, Clustering, Anchor-Point, and Prediction-based baselines by 2.3%, 1.1%, 1.5%, 0.3% and 1.6% when using GPT-3.5 on BIG-bench dataset, respectively. More importantly, incorporating IPOMP$_{Stage2}$ into the original baselines improves the stability significantly. For instance, IPOMP$_{Stage2}$ reduces the standard deviation by 18.8%, 60.0%, 6.9%, 10.8%, and 16.8% for Random, Boundary, Semantic, Anchor-Point, and Prediction-based when using GPT-3.5 on BIG-bench, respectively. In summary, IPOMP$_{Stage2}$ not only boosts the performance of baselines but also enhances their stability. Our results demonstrate that **IPOMP$_{Stage2}$ is a practical and adaptable enhancement for various data selection methods, effectively refining evaluation data by leveraging real-time model performance insights.**

Table 2: Comparison of the effectiveness and stability of the studied prompt optimization approaches with IPOMP and its variants.

	GPT-3.5 - BIG-bench			GPT-4o-mini - BIG-bench		
	IPOMP$_{Stage1}$	IPOMP$_{Random}$	IPOMP	IPOMP$_{Stage1}$	IPOMP$_{Random}$	IPOMP
EVOPROMPT	0.745±0.029	0.751±0.021	**0.776±0.017**	0.737±0.014	0.754±0.012	**0.758±0.011**
APO	0.730±0.031	0.738±0.015	**0.753±0.009**	0.739±0.012	0.757±0.012	**0.780±0.012**
APE	0.724±0.042	0.723±0.027	**0.742±0.010**	0.754±0.012	0.705±0.013	**0.794±0.012**
Average	0.733±0.034	0.737±0.021	**0.757±0.012**	0.743±0.012	0.738±0.012	**0.778±0.011**
	GPT-3.5 - LIAR			GPT-4o-mini - LIAR		
	IPOMP$_{Stage1}$	IPOMP$_{Random}$	IPOMP	IPOMP$_{Stage1}$	IPOMP$_{Random}$	IPOMP
EVOPROMPT	0.802±0.019	0.807±0.017	**0.818±0.015**	0.820±0.015	0.809±0.014	**0.838±0.011**
APO	0.801±0.021	0.812±0.018	**0.812±0.014**	0.797±0.022	0.824±0.015	**0.836±0.011**
APE	0.812±0.018	0.829±0.013	**0.832±0.011**	0.801±0.014	0.826±0.015	**0.827±0.013**
Average	0.805±0.020	0.816±0.016	**0.820±0.013**	0.806±0.017	0.820±0.015	**0.833±0.012**

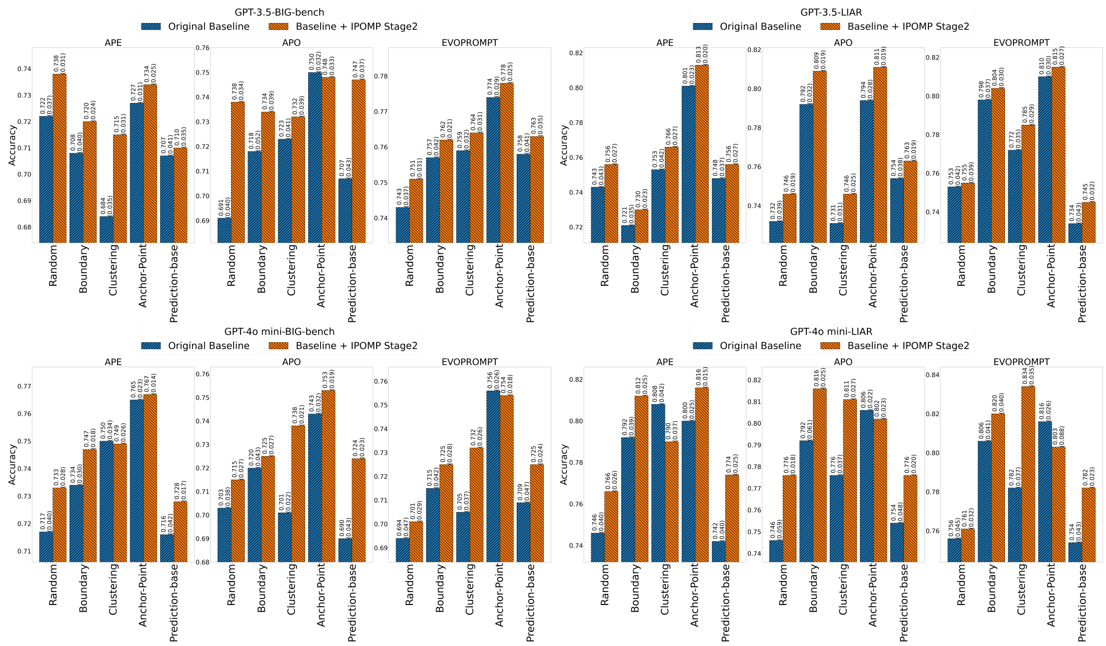

Figure 2: Comparison of the effectiveness of original baselines and Baseline+IPOMP$_{Stage2}$. SD is indicated in parentheses.

Table 3: Average execution time (in seconds) of the studied prompt optimization after applying IPOMP (including time required for each stage) and other baselines on BIG-bench when using GPT-3.5. Note that Random can be considered as the **pure execution time** of prompt optimization techniques as the execution time of random sampling is negligible. For Anchor-Point, we also present the time for the preliminary stage to collect the model confidence information.

	IPOMP$_{Stage1}$	IPOMP$_{Stage2}$	IPOMP	Random	Boundary	Clustering	Anchor-Point	Prediction-based
APO	0.45	2.74	470.86	469.74	470.34	470.83	469.74+200.36	480.57
APE	0.37	2.31	120.23	109.84	110.67	111.12	113.26+205.32	123.31
EVOPROMPT	0.51	3.43	613.75	609.35	611.53	608.38	609.31+207.85	622.61
Average	0.45	2.83	401.61	396.31	397.51	396.61	397.43+204.51	405.50

5.3 Impact of sample size

In this section, we further evaluate the impact of sample size across the prompt optimization techniques. Specifically, we conduct experiments with sample sizes of 5, 10, 15, 20, and 30 on GPT-3.5 with BIG-bench, as shown in Figure 3. Note that we observe the same trend on GPT-4o-mini with LIAR. Overall, the performance improves as the sample size increases from 5 to 20, then stabilizes or slightly declines beyond 20, suggesting that 20 samples may represent the optimal balance between cost and effectiveness for prompt optimization. Across all sample sizes, **IPOMP consistently outperforms other baselines across all prompt optimization techniques**, even with as few as 5 samples. Anchor-Point generally ranks

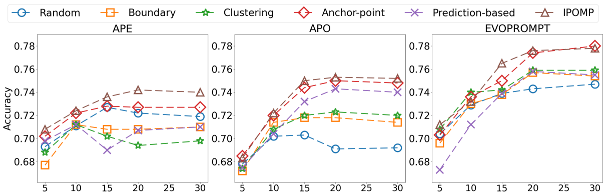

Figure 3: The impact of different sample sizes selected by the studied selection approaches on BIG-bench when using GPT-3.5.

Table 4: The cost analysis of the studied prompt optimization approaches and warm-up stage for Anchor-Point. The actual cost (in USD) is in parentheses.

	Prompt Optimization	Preliminary stage for Anchor-Point
APO	637 (0.20)	6,124 (0.95)
APE	231 (0.06)	2,277 (0.163)
EVOPROMPT	2,245 (2.75)	22,349 (6.18)
Average	1,037 (1.00)	10,250 (2.43)

second, reinforcing our earlier finding (Section 5.1) that approaches leveraging model performance data (IPOMP and Anchor-Point) tend to outperform those relying solely on semantic or historical performance data. In addition, IPOMP is more stable than other approaches across different sample sizes. See more details in Appendix A.6.

5.4 Overhead and cost analysis

The overhead of IPOMP is less than 1%, making it comparable to or better than other baselines. Table 3 presents the average execution time of three prompt optimization techniques, after applying coreset selection approaches on BIG-bench using GPT-3.5. IPOMP has a comparable overhead to other baselines. The overhead of IPOMP primarily comes from stage 2, which iteratively identifies and replaces redundant samples based on model performance (2.83 seconds on average). Stage 1 has a similar overhead as Boundary and Clustering (0.45 seconds on average). Anchor-Point has the highest overhead, which requires an additional preliminary stage to evaluate training samples on prompts, resulting in a significantly higher overhead of 51% and an execution time of approximately 200 seconds. Note that we observe a similar overhead when evaluated on LIAR using GPT-4o-mini.

For Clustering, Boundary, Prediction-based, and IPOMP, since the cost for those approaches only comes from LLM's inference for prompt optimization. For the approach itself, no additional cost is required. The calculation related to clustering and prediction can be done locally by deploying small models. The cost for prompt optimization is presented in Table 4. Similar to overhead, Anchor-Point requires evaluating the entire model performance during the preliminary stage, therefore additional cost is required.

6 Conclusion

We introduced IPOMP, a two-stage approach that enhances coreset selection for prompt optimization by leveraging both semantic and model performance information. Our method first selects representative and diverse samples based on semantic clustering and boundary analysis, followed by an iterative refinement process that integrates real-time model performance information to replace redundant samples with more informative ones. Evaluation on the BIG-bench dataset demonstrated that IPOMP consistently outperforms existing baselines with minimal computational overhead of less than 1%. Furthermore, the real-time performance-guided refinement approach in IPOMP is universally applicable to other coreset selection methods, enhancing their overall effectiveness and stability.

7 Limitations

One limitation is in our experiment we only use GPT-3.5 as the base model. Our findings may not generalize to other large language models with different architectures, training data, or capabilities. We encourage future research to evaluate our approach using a diverse set of base models to assess its applicability across different LLMs. We selected three prompt optimization techniques, from different families. We encourage future research to evaluate on more optimization techniques.

References

Sharat Agarwal, Himanshu Arora, Saket Anand, and Chetan Arora. 2020. Contextual diversity for active learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVI 16*, pages 137–153. Springer.

Alon Albalak, Yanai Elazar, Sang Michael Xie, Shayne Longpre, Nathan Lambert, Xinyi Wang, Niklas Muennighoff, Bairu Hou, Liangming Pan, Haewon Jeong, et al. 2024. A survey on data selection for language models. *arXiv preprint arXiv:2402.16827*.

Fabrizio Angiulli and Clara Pizzuti. 2002. Fast outlier detection in high dimensional spaces. In *European* *conference on principles of data mining and knowledge discovery*, pages 15–27. Springer.

BIG bench authors. 2023. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*.

Chengliang Chai, Jiayi Wang, Nan Tang, Ye Yuan, Jiabin Liu, Yuhao Deng, and Guoren Wang. 2023. Efficient coreset selection with cluster-based methods. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 167–178.

Yutian Chen, Max Welling, and Alex Smola. 2012. Super-samples from kernel herding. *arXiv preprint arXiv:1203.3472*.

Cody Coleman, Christopher Yeh, Stephen Mussmann, Baharan Mirzasoleiman, Peter Bailis, Percy Liang, Jure Leskovec, and Matei Zaharia. 2019. Selection via proxy: Efficient data selection for deep learning. *arXiv preprint arXiv:1906.11829*.

Mingkai Deng, Jianyu Wang, Cheng-Ping Hsieh, Yihan Wang, Han Guo, Tianmin Shu, Meng Song, Eric P Xing, and Zhiting Hu. 2022. Rlprompt: Optimizing discrete text prompts with reinforcement learning. *arXiv preprint arXiv:2205.12548*.

Melanie Ducoffe and Frederic Precioso. 2018. Adversarial active learning for deep networks: a margin based approach. *arXiv preprint arXiv:1802.09841*.

Chrisantha Fernando, D Banarse, Henryk Michalewski, Simon Osindero, and Tim Rocktäschel. Promptbreeder: Self-referential self-improvement via prompt evolution (2023). *arXiv preprint arXiv:2309.16797*.

Qingyan Guo, Rui Wang, Junliang Guo, Bei Li, Kaitao Song, Xu Tan, Guoqing Liu, Jiang Bian, and Yujiu Yang. 2023. Connecting large language models with evolutionary algorithms yields powerful prompt optimizers. *arXiv preprint arXiv:2309.08532*.

Sariel Har-Peled and Soham Mazumdar. 2004. On coresets for k-means and k-median clustering. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 291–300.

Yihao Huang, Chong Wang, Xiaojun Jia, Qing Guo, Felix Juefei-Xu, Jian Zhang, Geguang Pu, and Yang Liu. 2024. Semantic-guided prompt organization for universal goal hijacking against llms. *arXiv preprint arXiv:2405.14189*.

Gurusha Juneja, Nagarajan Natarajan, Hua Li, Jian Jiao, and Amit Sharma. 2024. Task facet learning: A structured approach to prompt optimization. *arXiv preprint arXiv:2406.10504*.

Seong Tae Kim, Farrukh Mushtaq, and Nassir Navab. 2020. Confident coreset for active learning in medical image analysis. *arXiv preprint arXiv:2004.02200*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Yong Lin, Chen Liu, Chenlu Ye, Qing Lian, Yuan Yao, and Tong Zhang. 2023. Optimal sample selection through uncertainty estimation and its application in deep learning. *arXiv preprint arXiv:2309.02476*.

Evan Z Liu, Behzad Haghgoo, Annie S Chen, Aditi Raghunathan, Pang Wei Koh, Shiori Sagawa, Percy Liang, and Chelsea Finn. 2021. Just train twice: Improving group robustness without training group information. In *International Conference on Machine Learning*, pages 6781–6792. PMLR.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35.

Yu A Malkov and Dmitry A Yashunin. 2018. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence*, 42(4):824–836.

Katerina Margatina, Giorgos Vernikos, Loïc Barrault, and Nikolaos Aletras. 2021. Active learning by acquiring contrastive examples. *arXiv preprint arXiv:2109.03764*.

Lorenzo Pacchiardi, Lucy G Cheke, and José Hernández-Orallo. 2024. 100 instances is all you need: predicting the success of a new llm on unseen data by testing on a few instances. *arXiv preprint arXiv:2409.03563*.

Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. 2021. Deep learning on a data diet: Finding important examples early in training. *Advances in neural information processing systems*, 34:20596–20607.

Felipe Maia Polo, Lucas Weber, Leshem Choshen, Yuekai Sun, Gongjun Xu, and Mikhail Yurochkin. 2024. tinybenchmarks: evaluating llms with fewer examples. *arXiv preprint arXiv:2402.14992*.

Reid Pryzant, Dan Iter, Jerry Li, Yin Tat Lee, Chenguang Zhu, and Michael Zeng. 2023. Automatic prompt optimization with" gradient descent" and beam search. *arXiv preprint arXiv:2305.03495*.

Gopi Krishnan Rajbahadur, Shaowei Wang, Yasutaka Kamei, and Ahmed E Hassan. 2017. The impact of using regression models to build defect classifiers. In *2017 IEEE/ACM 14th International Conference on Mining Software Repositories (MSR)*, pages 135–145. IEEE.

N Reimers. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Ozan Sener and Silvio Savarese. 2017. Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489*.

Samarth Sinha, Han Zhang, Anirudh Goyal, Yoshua Bengio, Hugo Larochelle, and Augustus Odena. 2020. Small-gan: Speeding up gan training using core-sets. In *International Conference on Machine Learning*, pages 9005–9015. PMLR.

Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J Gordon. 2018. An empirical study of example forgetting during deep neural network learning. *arXiv preprint arXiv:1812.05159*.

Rajan Vivek, Kawin Ethayarajh, Diyi Yang, and Douwe Kiela. 2023. Anchor points: Benchmarking models with much fewer examples. *arXiv preprint arXiv:2309.08638*.

Shaowei Wang, Tse-Hsun Chen, and Ahmed E Hassan. 2018. Understanding the factors for fast answers in technical q&a websites: An empirical study of four stack exchange websites. *Empirical Software Engineering*, 23:1552–1593.

William Yang Wang. 2017. " liar, liar pants on fire": A new benchmark dataset for fake news detection. *arXiv preprint arXiv:1705.00648*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. 2024. Large language models as optimizers. *Preprint*, arXiv:2309.03409.

Weiran Yao, Shelby Heinecke, Juan Carlos Niebles, Zhiwei Liu, Yihao Feng, Le Xue, Rithesh Murthy, Zeyuan Chen, Jianguo Zhang, Devansh Arpit, et al. 2023. Retroformer: Retrospective large language agents with policy gradient optimization. *arXiv preprint arXiv:2308.02151*.

Bianca Zadrozny. 2004. Learning and evaluating classifiers under sample selection bias. In *Proceedings of the twenty-first international conference on Machine learning*, page 114.

Tianjun Zhang, Xuezhi Wang, Denny Zhou, Dale Schuurmans, and Joseph E Gonzalez. 2022. Tempera: Test-time prompting via reinforcement learning. *arXiv preprint arXiv:2211.11890*.

Zhihan Zhang, Shuohang Wang, Wenhao Yu, Yichong Xu, Dan Iter, Qingkai Zeng, Yang Liu, Chenguang Zhu, and Meng Jiang. 2023. Auto-instruct: Automatic instruction generation and ranking for blackbox language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9850–9867, Singapore. Association for Computational Linguistics.

Lexin Zhou, Fernando Martínez-Plumed, José Hernández-Orallo, Cèsar Ferri, and Wout Schellaert. 2022a. Reject before you run: Small assessors anticipate big language models. In *EBeM@ IJCAI*.

Lexin Zhou, Pablo A Moreno-Casares, Fernando Martínez-Plumed, John Burden, Ryan Burnell, Lucy Cheke, Cèsar Ferri, Alexandru Marcoci, Behzad Mehrbakhsh, Yael Moros-Daval, et al. 2023. Predictable artificial intelligence. *arXiv preprint arXiv:2310.06167*.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2022b. Large language models are human-level prompt engineers. *arXiv preprint arXiv:2211.01910*.

A Appendix

A.1 Experimental setup

We use the API provided by OpenAI to access GPT-3.5-turbo as the base model. The temperature is set to 0 for inference. All experiments are done in Python 3.10. All experiments were conducted on a machine equipped with a GPU of 24GB, a 24-core CPU, and 24 GB of RAM.

A.2 Detailed implementation of baselines

Anchor-Point (Vivek et al., 2023). To adopt this approach in the context of prompt optimization, we first collect the model's confidence scores on the examples. Specifically, we run the training dataset through an initial set of prompts and then cluster the examples based on their confidence scores across these prompts, following (Vivek et al., 2023). During this stage, we generate 10 prompts using the prompt optimization technique as the initial set. Evaluating confidence on more prompts typically can lead to better clustering results, while inference is expensive. To balance the quality and cost, we select 10 prompts. Also, evaluating the confidence of the entire training dataset is infeasible and expensive. To expedite the process and save inference costs, we first select 200 examples from the entire training data using the same approach of stage 1, and then apply Anchor-Point to select the final evaluation data. **Prediction-based (Pacchiardi et al., 2024).** This approach predicts the performance of an instance on an LLM by training a generic assessor based on the performance of each sample in the training set on existing LLMs. We adapt the generic assessor using our dataset, BIG-bench, on GPT-3.5, following the approach outlined by (Pacchiardi et al., 2024). We then use the trained assessor to predict the performance of each example in the training dataset on a set of initial prompts and subsequently cluster those examples based on their predicted performance, similar to the Anchor-Point.

A.3 Dataset statistics

The sizes of the training and testing datasets used in our experiments are presented in Table 5.

A.4 Ablation analysis on boundary cases selection

In Stage 1: Diverse sample selection, we select boundary cases to diversify our samples. To understand the impact of the boundary case, we conducted an ablation analysis on the boundary case and the results are shown in Table 6. Note that we extended our evaluation with one more data LIAR and one more model GPT-4o mini, as suggested by the reviewer. As the results show, boundary case selection makes significant contribution to IPOMP.

Table 5: The size of Training dataset and testing in our experiments

	Training Dataset	Testing dataset
Navigation	800	200
Implicatures	392	100
Metaphor Understanding	544	136
Sports Understanding	788	198
Natural Language Inference	588	147
Fake News Detection	10240	1267

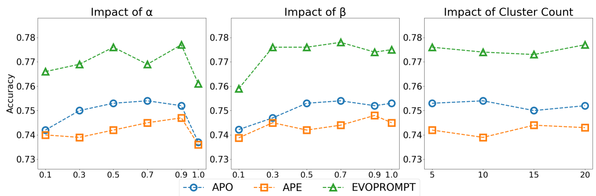

Figure 4: The impact of different values of hyper-parameters α , β and cluster size K on the effectiveness of APE, APO, and EVOPROMPT.

A.5 Results of impact of hyper-parameters

We examine the impact of key hyper-parameters in IPOMP: α which controls the proportion of samples selected based on semantic clustering in stage 1, and K which controls the cluster size in stage 1, and β which determines the proportion of redundant samples to be replaced in stage 2. Figure 4 presents the results of BIG-bench when using GPT-3.5.

As α increases from 0.1 to 0.9, the accuracy of IPOMP consistently improves across all three prompt optimization techniques. However, when α exceeds 0.9, the performance of APE, APE, and EVOPROMPT degrades significantly. This observation suggests that while incorporating a small portion of boundary cases can enhance the diversity of the evaluation data and improve the performance of IPOMP, relying too heavily on boundary cases can negatively impact the overall effectiveness (i.e., $\alpha < 0.9$). In contrast, selecting samples purely based on semantic clustering results in suboptimal performance (i.e., $\alpha = 1$).

In terms of β, as it increases from 0.1 to 0.7, the performance of the three prompt optimization techniques improves gradually, particularly noticeable

Table 6: Comparison of IPOMP and IPOMP without boundary cases.

	GPT-4o-mini- Big-Bench		GPT-4o-mini-LIAR		GPT-3.5-LIAR		GPT-3.5-Big-Bench	
	IPOMP	w/o Boundary	IPOMP	w/o Boundary	IPOMP	w/o Boundary	IPOMP	w/o Boundary
EVOPROMPT	**0.758±0.011**	0.732±0.026	**0.838±0.011**	0.834±0.035	**0.818±0.015**	0.785±0.029	**0.776±0.017**	0.759±0.042
APO	**0.780±0.012**	0.738±0.021	**0.836±0.011**	0.811±0.027	**0.812±0.014**	0.746±0.025	**0.753±0.009**	0.723±0.039
APE	**0.794±0.012**	0.749±0.026	**0.827±0.013**	0.790±0.037	**0.832±0.011**	0.766±0.027	**0.742±0.010**	0.715±0.031
Average	**0.778±0.011**	0.740±0.024	**0.833±0.012**	0.812±0.033	**0.820±0.013**	0.765±0.027	**0.757±0.012**	0.732±0.037

Table 7: Comparison of the effectiveness (in terms of accuracy) and stability (in terms of deviation) of the studied prompt optimization approaches with different evaluation data sampling approaches on BIG-bench when using GPT-3.5.

	Size	Random	Boundary	Clustering	Anchor-Point	Prediction-based	IPOMP
APO	5	0.678±0.013	0.672±0.043	0.674±0.021	**0.685↑±0.029**	0.675±0.039	0.683±0.017↓
	10	0.702±0.031	0.714±0.029	0.708±0.019	0.720±0.015	0.704±0.032	**0.722↑±0.010**↓
	15	0.703±0.023	0.718±0.042	0.720±0.022	0.744±0.020	0.732±0.038	**0.750↑±0.008**↓
	20	0.691±0.040	0.718±0.052	0.723±0.025	0.750±0.020	0.743±0.042	**0.753↑±0.009**↓
	30	0.692±0.023	0.714±0.038	0.720±0.028	0.748±0.028	0.740±0.038	**0.752↑±0.011**↓
APE	5	0.693±0.022	0.677±0.024	0.688±0.023	0.702±0.021	0.698±0.029	**0.708↑±0.019**↓
	10	0.711±0.026	0.712±0.024	0.712±0.020	0.722±0.026	0.712±0.030	**0.724↑±0.014**↓
	15	0.727±0.041	0.708±0.042	0.702±0.032	0.728±0.032	0.690±0.042	**0.736↑±0.013**↓
	20	0.722±0.037	0.708±0.045	0.684±0.032	0.727±0.035	0.707±0.048	**0.742↑±0.010**↓
	30	0.719±0.035	0.710±0.040	0.698±0.034	0.727±0.033	0.710±0.032	**0.740↑±0.012**↓
EVOPROMPT	5	0.704 ±0.034	0.696±0.027	0.708±0.018	0.703±0.027	0.673±0.034	**0.712↑±0.020**↓
	10	0.729±0.029	0.730±0.026	**0.740↑±0.032**	0.736±0.018	0.712±0.020	0.732±0.017↓
	15	0.739±0.019	0.738±0.029	0.742±0.019	0.750±0.019	0.739±0.029	**0.765↑±0.011**↓
	20	0.743±0.028	0.757±0.022	0.759±0.031	0.774±0.028	0.758±0.025	**0.776↑±0.017**↓
	30	0.747±0.032	0.754±0.032	0.759±0.034	**0.780↑±0.036**	0.755±0.030	0.778±0.018↓

in the range from 0.1 to 0.3. For APO and EVOPROMPT, their performance still gets improved until β reaches 0.7 and gets degraded after 0.7. For APE, its performance degrades from 0.3 to 0.5, and improves after 0.5. This enhancement is attributed to the replacement of redundant examples during each iteration, which effectively increases the diversity of the evaluation samples based on the feedback from the model in real-time.

We also investigate the impact of the cluster size K. As Figure 4 shows, the size of clusters does not impact the effectiveness of IPOMP significantly.

A.6 Detailed results of impact of sample size

The accuracy and standard deviation of prompt optimization techniques over different sizes of samples selected by the studied approaches are presented in Table 7.

A.7 Impact of replacement strategy in Stage 2

. To verify our hypothesis that selecting dissimilar examples yields significantly lower performance correlation compared to both random and similar selection, we conducted an ablation study to compare three sample replacement strategies:

- **Dissimilar**: Replacing redundant samples with the most dissimilar samples from the training dataset (our proposed method).

Table 8: Comparison of five-number summary of correlation among the samples selected using different strategies.

	Min	Q1	Median	Q3	Max
Similar	0.623	0.938	0.960	0.978	0.999
Random	0.002	0.418	0.634	0.882	0.943
Dissimilar	0.002	0.109	0.400	0.480	0.681

- **Random**: Replacing redundant samples with randomly selected samples from the training dataset.
- **Similar**: Replacing redundant samples with the most similar samples from the training dataset.

We measured the correlation between the performance of the original samples and replaced samples using the above replacement strategies. The results shown in Table 8 confirm our hypothesis. This validates the importance of our replacement strategy in influencing model behavior.

A.8 Case study

Figure 5 presents the correlation among samples before and after applying real-time model performance-guided refinement in APO. As we can observe, after the refinement, the redundancy of examples (correlation > 0.9) are significantly

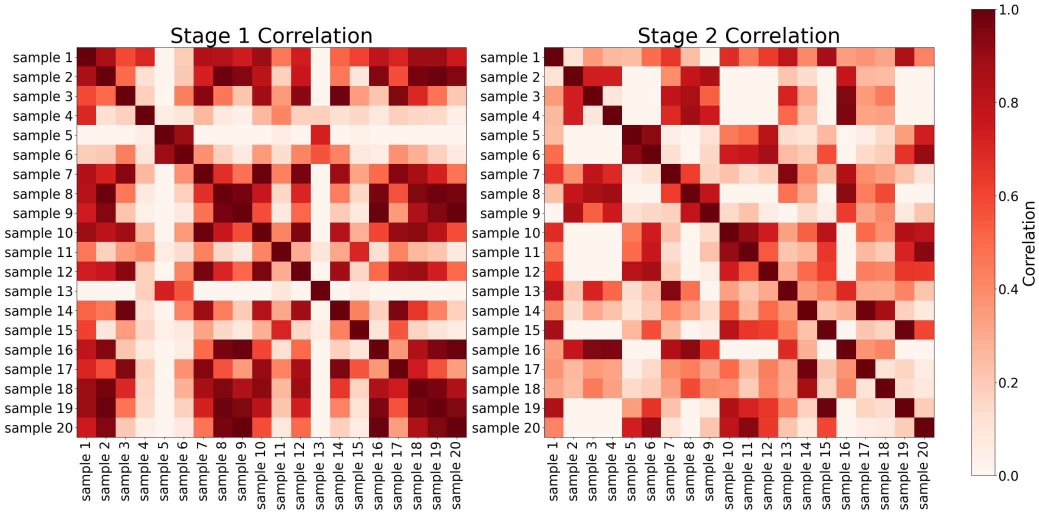

Figure 5: Correlation among samples before and after the first round of real-time model performance-guided refinement in APO on the Implicatures dataset. Each cell represents the correlation between a pair of samples.

reduced from 19% to 10%.

We present the selected examples in stage 1 and after the first round of refinement in stage 2 below.

Selected Examples by stage 1 of IPOMP

- Example 1: Speaker 1: *'This is a costume?'* Speaker 2: *'Aaaiyyyy... worked on it all night long!'*
- Example 2: Speaker 1: *'Do you love me?'* Speaker 2: *'My love for you is as deep as the ocean.'*
- Example 3: Speaker 1: *'Did you sleep well last night?'* Speaker 2: *'Last night, I slept like a log.'*
- Example 4: Speaker 1: *'Do you think that Dr. Luby will organize a theatre trip to New York this year?'* Speaker 2: *'I have already signed up for it.'*
- Example 5: Speaker 1: *'Did you report Private Barnes to your superiors?'* Speaker 2: *'I remember thinking very highly of Private Barnes, and not wanting to see his record tarnished by a formal charge.'*
- Example 6: Speaker 1: *'I bought the wrong math book. Here is the receipt. Can I get my money back?'* Speaker 2: *'Not after ten days. But you can exchange something for it.'*
- Example 7: Speaker 1: *'Does it bother you that your husband goes away on long business trips?'* Speaker 2: *'Absence makes the heart grow fonder.'*
- Example 8: Speaker 1: *'Did you order the code red?'* Speaker 2: *'You're goddamn right.'*
- Example 9: Speaker 1: *'My client is taking me to a really fancy restaurant tonight. So I am wearing this new cologne. I got a sample of it from a magazine. Can you smell it?'* Speaker 2: *'From across the room. But it is not exactly subtle. Is it?'*
- Example 10: Speaker 1: *'Are you a Dodgers fan?'* Speaker 2: *'I don't like baseball.'*
- Example 11: Speaker 1: *'Is everyone comfortable?'* Speaker 2: *'Everyone is on pins and needles.'*
- Example 12: Speaker 1: *'I feel horrible. Debbie was furious that I lost her notes. Do you think I should apologize to her again?'* Speaker 2: *'If I were you, I would cool off for some days before I talk to her again.'*
- Example 13: Speaker 1: *'Should I decide now?'* Speaker 2: *'Why don't you go home and sleep on it?'*
- Example 14: Speaker 1: *'Did I do it well?'* Speaker 2: *'You were as brave as a lion.'*
- Example 15: Speaker 1: *'Are you coming with me to the exhibition today?'* Speaker 2: *'I made plans with Susan to go to the exhibition tomorrow afternoon.'*
- Example 16: Speaker 1: *'Does it rain here nowadays?'* Speaker 2: *'It's been raining for 40 days and 40 nights.'*
- Example 17: Speaker 1: *'Are you planning to buy a house?'* Speaker 2: *'I really want a place to call my own.'*
- Example 18: Speaker 1: *'Is it a good product?'* Speaker 2: *'They had put a lot of thought into making it.'*
- Example 19: Speaker 1: *'Is that book about lullabies?'* Speaker 2: *'It is about symphonies.'*
- Example 20: Speaker 1: *'But aren't you afraid?'* Speaker 2: *'Ma'am, sharks never attack anybody.'*

Selected Examples after first round of refinment at stage 2.

Here are selected examples from Stage 2 of our tool's processing:

- Example 1: Speaker 1: *'This is a costume?'* Speaker 2: *'Aaaiyyyy... worked on it all night long!'*
- Example 2: Speaker 1: *'Do you love me?'* Speaker 2: *'My love for you is as deep as the ocean.'*
- Example 3: Speaker 1: *'Did you sleep well last night?'* Speaker 2: *'Last night, I slept like a log.'*
- Example 4: Speaker 1: *'My client is taking me to a really fancy restaurant tonight. So I am wearing this new cologne. I got a sample of it from a magazine. Can you smell it?'* Speaker 2: *'From across the room. But it is not exactly subtle. Is it?'*
- Example 5: Speaker 1: *'I bought the wrong math book. Here is the receipt. Can I get my money back?'* Speaker 2: *'Not after ten days. But you can exchange something for it.'*
- Example 6: Speaker 1: *'I feel horrible. Debbie was furious that I lost her notes. Do you think I should apologize to her again?'* Speaker 2: *'If I were you, I would cool off for some days before I talk to her again.'*
- Example 7: Speaker 1: *'Should I decide now?'* Speaker 2: *'Why don't you go home and sleep on it?'*
- Example 8: Speaker 1: *'Are you planning to buy a house?'* Speaker 2: *'I really want a place to call my own.'*
- Example 9: Speaker 1: *'Is that book about lullabies?'* Speaker 2: *'It is about symphonies.'*
- Example 10: Speaker 1: *'Are you angry at me?'* Speaker 2: *'To err is human, to forgive divine.'*
- Example 11: Speaker 1: *'Does it rain here nowadays?'* Speaker 2: *'It's been raining for 40 days and 40 nights.'*
- Example 12: Speaker 1: *'That cake looks delicious. Aren't you going to have some with me?'* Speaker 2: *'I am watching my calorie intake.'*
- Example 13: Speaker 1: *'Does she know how to play the piano?'* Speaker 2: *'Now it is like second nature to her.'*
- Example 14: Speaker 1: *'Do you have these journals?'* Speaker 2: *'I can have them flown here from Geneva in an hour.'*
- Example 15: Speaker 1: *'Do you have any agricultural background?'* Speaker 2: *'I used to work in an office.'*
- Example 16: Speaker 1: *'Do you have field hands that help you?'* Speaker 2: *'We work the land alone.'*
- Example 17: Speaker 1: *'I feel horrible. Debbie was furious that I lost her notes. Do you think I should apologize to her again?'* Speaker 2: *'If I were you, I would cool off for some days before I talk to her again.'*
- Example 18: Speaker 1: *'You have it, then?'* Speaker 2: *'I had to slit a few throats to get it.'*
- Example 19: Speaker 1: *'Were you hiding from me?'* Speaker 2: *'I didn't want to scare you.'*
- Example 20: Speaker 1: *'Are you coming with me to the exhibition today?'* Speaker 2: *'I made plans with Susan to go to the exhibition tomorrow afternoon.'*