Rethinking the Evaluation of In-Context Learning for LLMs

Guoxin Yu, **Lemao Liu***, **Mo Yu**, **Yue Yu***, **Xiang Ao***

Key Lab of Intelligent Information Processing of Chinese Academy of Sciences (CAS), Institute of Computing Technology, CAS, Beijing 100190, China.
Peng Cheng Laboratory, Shenzhen, China.
University of Chinese Academy of Sciences, Beijing 100049, China.
Wechat AI, Tencent.
Key Lab of AI Safety, Chinese Academy of Sciences, Beijing, 100094, China.
{yuguoxin20g,aoxiang}@ict.ac.cn,{yuy}@pcl.ac.cn
{redmondliu}@tencent.com, {moyumyu}@global.tencent.com

Abstract

In-context learning (ICL) has demonstrated excellent performance across various downstream NLP tasks, especially when synergized with powerful large language models (LLMs). Existing studies evaluate ICL methods primarily based on downstream task performance. This evaluation protocol overlooks the significant cost associated with the demonstration configuration process, i.e., tuning the demonstration as the ICL prompt. However, in this work, we point out that the evaluation protocol leads to unfair comparisons and potentially biased evaluation, because we surprisingly find the correlation between the configuration costs and task performance. Then we call for a two-dimensional evaluation paradigm that considers both of these aspects, facilitating a fairer comparison. Finally, based on our empirical finding that the optimized demonstration on one language model generalizes across language models of different sizes, we introduce a simple yet efficient strategy that can be applied to any ICL method as a plugin, yielding a better trade-off between the two dimensions according to the proposed evaluation paradigm.

1 Introduction

Recent years have witnessed a great success on In-context learning (ICL) (Brown et al., 2020) for a range of downstream natural language processing (NLP) tasks, such as text classification, sentiment analysis, and question answering (Liu et al., 2022; Lu et al., 2022; Yang et al., 2023b; Wang et al., 2023a). Its main idea is to configurate a demonstration (i.e., selecting examples and organizing their order), and then integrate it with an input to formulate a prompt for a specific task, which is subsequently received by a language model to produce the corresponding answer. Many ICL methods have been proposed (Liu et al., 2022; Gonen et al., 2023; Honovich et al., 2023; Yang et al., 2023a; Rubin et al., 2022; Iter et al., 2023; Wang et al., 2024) and they mainly differ in demonstration configuration, which plays a critical role in ICL.

Notable ICL methods in task performance typically consume a significant cost to configure their demonstrations, and such cost may even hinder their deployment in real-world applications. Despite the important role of the demonstration configuration cost, all prior studies surprisingly overlook it in their evaluations. More importantly, our pilot study reveals that the demonstration configuration cost of an ICL method is correlated with its performance: increasing the cost usually leads to gains in performance. As a result, this finding indicates that the standard evaluation based on task performance induces unfair comparisons and biased evaluation results when evaluating different ICL methods (§3).

In this paper, we thereby call for a two-dimensional evaluation paradigm considering task performance and demonstration configuration cost, which provides a more comprehensive perspective to compare diverse ICL methods. Under the new evaluation paradigm, we employ two variants to re-evaluate existing ICL methods for fair comparison, in terms of task performance as well as configuration cost (§4). Moreover, drawing inspiration from the new evaluation results, we propose a simple strategy to optimize the ICL methods towards zero configuration cost: it resorts to a very small

*Corresponding authors.

language model for demonstration configuration even if conducting inference on a super-large language model. The strategy is effective because of the empirically verified property on the demonstration transferability that *the optimized demonstration on one language model is transferable to another language model*. Our strategy is applicable to any demonstration configuration methods and we apply it to several strong ICL methods for both open-source (e.g., OPT) and closed-source (e.g., ChatGPT) large language models (LLMs). Our experiments show that our strategy indeed achieves better trade-offs between the two dimensions considered in the proposed evaluation paradigm (§5). Our contributions could be summarized as follows:

- We for the first time point out that the standard evaluation protocol suffers from an issue of unfairness due to the overlooked demonstration configuration cost.
- To ensure a fair comparison, we call for a two-dimensional evaluation paradigm to evaluate ICL methods.
- We propose a simple yet effective strategy to configurate demonstration for various ICL methods as a plugin that achieves better trade-offs between task performance and demonstration configuration cost.

2 Preliminary

2.1 In-Context Learning

The advent of LLMs has propelled ICL methods to the forefront of NLP paradigms (Dakhel et al., 2024; Minaee et al., 2024), which generally select examples from a training set (Dong et al., 2022) and combine them with a target input to construct a prompt. A pre-trained language model subsequently processes the prompt to yield the requisite response for a designated downstream task. In this paper, we simply decompose the ICL process into example selection and ordering and use text classification tasks to illustrate the formulation. For a given text x, the objective of ICL methods for text classification is to predict the label $\hat{y}$ of x with k demonstration examples C as follows:

$$\begin{aligned} &P(y_j|x) = f_{\mathcal{LM}}(x, C), \\ &C = \{\langle x_1, y_1\rangle, \ldots, \langle x_k, y_k\rangle\}, \\ &\hat{y} = \text{argmax}_{y_j} P(y_j|x) \end{aligned} \tag{1}$$

where C comprises k input-output pairs. Each $\langle x_i, y_i\rangle$ will be structured into $\mathcal{T}(\langle x_i, y_i\rangle)$ using a manually crafted template $\mathcal{T}$. $f_{\mathcal{LM}}$ denotes a language model that maps the input text to label distributions.

2.2 Demonstration Configuration

The configuration of demonstrations, encompassing the selection and ordering of training examples, is pivotal for achieving optimal performance in ICL (Liu et al., 2022). Selecting relevant and diverse examples is essential as it aids the language model in comprehending the task's scope and intricacies. The sequence in which these examples are also presented plays a significant role (Minaee et al., 2024), particularly when the selected demonstration examples are not robust enough.

This paper primarily focuses on **demonstration configuration** methods, rather than on the design of templates or ICL paradigms. Generally, demonstration configuration includes (demonstration) example selection and example ordering.

Example Selection Example selection aims to select effective input-output pairs from a training set $\mathcal{D}$ for the input target text x according to a retrieval metric $\text{sim}(x, d_i)$. It measures the relevance of example d_i to the target input x for the downstream task.

Example Ordering Lu et al. (2022) claimed the ICL performance is sensitive to the ordering of selected examples. Given the selected k examples, $k!$ permutations could be generated. Each permutation p_i is subsequently fed into LMs to yield a designated metric $e(p_i)$, which determines the optimal order finally.

Configuration Methods In this paper, we focus on the following representative demonstration configuration methods, ensuring the comprehensiveness and rationality of the empirical study.

For **example selection**, we employ three distinct methods, namely *Random*, *TopK*, and *DataModel*. *Random* randomly selects examples from the training set. *TopK* selects the top K examples based on similarities calculated using the Sentence Transformer (Reimers and Gurevych, 2019). *DataModel* (Chang and Jia, 2023) pre-trained a datamodel (Ilyas et al., 2022) according to the output logits of LLMs to approximate the effectiveness of an example.

We select three methods for **example ordering**. *GlobalE* (Lu et al., 2022) validates an example permutation through a global entropy, aiming to mitigate the issue of highly imbalanced predictions. *LocalE* (Lu et al., 2022) ranks the example permutation with a local entropy to avoid model overconfidence. *MDL* (Wu et al., 2022) uses the Minimum Description Length principle (Grünwald, 2007) to find the permutation with the minimum codelength for compressing testing samples.

Besides, we obtain two **hybrid methods** by combining *TopK* with either *LocalE* or *MDL*, inspired by Wu et al. (2022). Please refer to Appendix A for more details of these methods.

3 Pilot Study on Demonstration Configuration Cost

3.1 Demonstration Configuration Cost

Demonstration configuration plays an important role in ICL. Existing demonstration configuration methods have attempted to design strategies for configurating (selecting and ordering) demonstration examples, which lies in the computation of $\mathrm{sim}(x, d_i)$ and $e(p_i)$ mentioned in §2.2. This computation process relies on the output probabilities from LLMs towards the downstream tasks after receiving a demonstration and input text, generally necessitating repeated calls of LLMs. As a result, it is natural to define the demonstration configuration cost for a demonstration configuration method in terms of calling a targeted LLM as follows.

Definition of demonstration configuration cost Formally, given a targeted LLM system $\mathcal{S}$ and a demonstration configuration method M, the demonstration configuration cost (or DC Cost for brevity) is measured by *the total number of calling a targeted LLM system* $\mathcal{S}$ when optimizing a demonstration by using the configuration method M to select examples and order the selected examples. As an illustration, GlobalE selects k training examples and generates $k!$ possible permutations. Each permutation was evaluated against $|D_{val}|$ validation instances, resulting in a substantial DC cost of $k! \times |D_{val}|$ validations calling $\mathcal{S}$.

Importance of DC cost Although prior studies do not take into account the DC cost, it actually plays an important role in configurating ICL for LLMs. On one hand, the DC cost of a configuration method directly determines its configuration efficiency. Such efficiency may be critical in some situations. For example, if the DC cost for a method is very high, it may be impractical to be applied to GPT-4 due to the considerable computing overhead and financial cost. On the other hand, more importantly, there exists a relationship between DC cost and the downstream task performance, which leads to a critical issue in the standard evaluation paradigm for in-context learning. In the next subsection, we will empirically conduct a pilot study to investigate the relationship between DC cost and task performance for each configuration method.

3.2 Experimental Results

Setting We executed all the methods with GPT2-xl (1.5B) (Radford et al., 2019) and OPT-2.7B (Zhang et al., 2022a), and assessed them on SST2, SST5, Trec, and AgNews datasets. We run existing methods with altering their DC costs and illustrate the results in Figure 1.

Higher DC cost yields higher accuracy As Figure 1 shows, we delve deeper into the correlation between performance and DC cost across two datasets for each method. In each sub-figure, we observe a consistent uptrend in performance alongside escalating DC costs. Specifically, a lower DC cost means demonstration selection within a limited demonstration pool or validation of candidate demonstrations on insufficient data, thus leading to sub-optimal demonstration selection and inferior performance. Conversely, increasing the DC cost is more likely to find a demonstration that is relatively suitable for a specific downstream task.

On the bias of the standard evaluation Different ICL methods show varying correlations with the DC cost. The results in Figure 1 reveal that MDL exhibits greater potential with increasing DC cost, particularly pronounced on SST2. In contrast, GlobalE displays a more gradual variation, yet consistently drivers superior performance overall, which may potentially stem from access to its more comprehensive permutation search, albeit with less validation data available. Besides, it is noted that DataModel appears to showcase a steeper slope. We conjecture this is due to the fact that DataModel requires training a datamodel for demonstration configuration by consuming DC cost. The larger variation of DC cost displayed in the x-axis greatly influences the efficacy of the datamodel training and consequently affects the accuracy.

Moreover, an inferior method might surpass another method when subjected to an increased DC

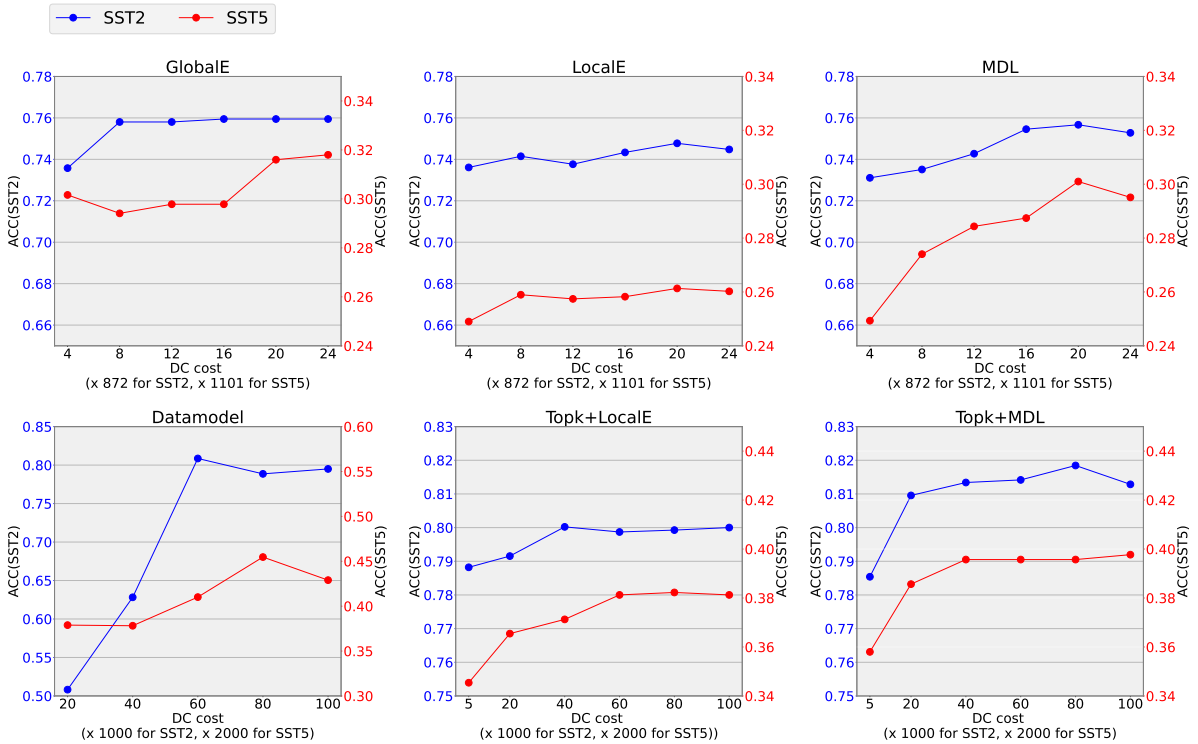

Figure 1: The relationships between accuracy and DC costs. The accuracy on SST2 and SST5 are reported by double y-axes. We manipulate the number of candidate permutations for each validation data to alter the DC cost for each method. We adjust the scales of y-axis to clearly show the relationships between performance and DC cost in distinctive methods. More details can be found in Appendix B.

cost. For example, TopK+MDL with 100×1000 DC cost could outperform DataModel on SST2. Conversely, MDL with lower DC cost may be inferior to GlobalE. These observations further demonstrated the biased comparison under the standard evaluation without consideration of the DC cost.

4 A Call for Two-dimensional Evaluation

As analyzed in §3.2, the higher DC cost might result in improved accuracy and thus it would be unfair to compare different ICL methods solely in terms of accuracy as in standard evaluation. Therefore, to ensure fair comparison and unbiased evaluation, we call for a two-dimensional evaluation for ICL methods, which takes into account both accuracy and DC cost. In this section, we employ two different implementations of the two-dimension evaluation to re-evaluate several ICL methods.

4.1 Bi-objective Evaluation

The bi-objective evaluation depicts both accuracy and the DC cost when comparing different ICL configuration methods. In addition to ensuring a fair comparison, mentioning the DC cost is meaningful and helpful for users to choose suitable ICL methods for their downstream applications. For example, if the accuracy is the top priority for a user, then the ICL method with the best accuracy is preferred; on the contrary, if the DC cost is the top priority, then the ICL method with the lowest DC cost is a better choice for the user.

The main results in terms of bi-objective evaluation are reported in Table 1, in which several intriguing observations emerge from the experimental results. On one hand, according to the accuracy as the standard evaluation criterion following prior studies (Liu et al., 2022), DataModel performs the best among the non-hybrid methods, followed by GlobalE, MDL, and LocalE, and Random performs the worst. On the other hand, in terms of DC cost [1], DataModel has the largest cost followed by GlobalE. Except for Random, LocalE has the lowest DC cost.

4.2 Evaluation by Controlling DC cost

Under the proposed two-dimensional paradigm, we re-evaluate these ICL methods by controlling DC cost. Specifically, we maintain a consistent DC

[1]The DC cost of the same method across different datasets is inconsistent because the initial validation data is different.

Models	SST2	SST5	Trec	AgNews	Avg.
GPT2-xl (1.5B)					
Random	$0.6911_{\pm 0.0080}(0.00)$	$0.2177_{\pm 0.0063}(0.00)$	$0.3632_{\pm 0.0122}(0.00)$	$0.5153_{\pm 0.0021}(0.00)$	0.4468
TopK	$0.7660_{\pm 0.0000}(0.00)$	$0.3349_{\pm 0.0000}(0.00)$	$0.8080_{\pm 0.0000}(0.00)$	$0.9067_{\pm 0.0000}(0.00)$	0.7039
DataModel	$0.7910_{\pm 0.0312}(10.0)$	$0.3852_{\pm 0.0496}(20.0)$	$0.5618_{\pm 0.0391}(30.0)$	$0.7878_{\pm 0.0117}(20.0)$	0.6315
LocalE	$0.7477_{\pm 0.0082}(0.36)$	$0.2573_{\pm 0.0104}(0.44)$	$0.3640_{\pm 0.0072}(0.10)$	$0.5107_{\pm 0.0012}(1.52)$	0.4699
MDL	$0.7566_{\pm 0.0029}(1.82)$	$0.3090_{\pm 0.0042}(2.21)$	$0.3680_{\pm 0.0211}(0.50)$	$0.5333_{\pm 0.0032}(3.80)$	0.4917
GlobalE	$0.7595_{\pm 0.1472}(2.09)$	$0.3160_{\pm 0.0616}(2.64)$	$0.4516_{\pm 0.0749}(1.27)$	$0.5667_{\pm 0.1353}(2.40)$	0.5235
TopK+LocalE	$0.7911_{\pm 0.0089}(0.36)$	$0.3405_{\pm 0.0086}(0.44)$	$0.8268_{\pm 0.0127}(0.10)$	$0.8743_{\pm 0.0040}(1.52)$	0.7082
TopK+MDL	$0.8164_{\pm 0.0063}(1.82)$	$0.3883_{\pm 0.0061}(2.21)$	$0.8336_{\pm 0.0162}(0.50)$	$0.8993_{\pm 0.0017}(3.80)$	0.7344
OPT-2.7B					
Random	$0.9187_{\pm 0.0022}(0.00)$	$0.3169_{\pm 0.0122}(0.00)$	$0.4004_{\pm 0.0047}(0.00)$	$0.5300_{\pm 0.0585}(0.00)$	0.5415
TopK	$0.9330_{\pm 0.0000}(0.00)$	$0.4170_{\pm 0.0000}(0.00)$	$0.8080_{\pm 0.0000}(0.00)$	$0.8900_{\pm 0.0000}(0.00)$	0.7620
DataModel	$0.9420_{\pm 0.0207}(10.0)$	$0.4852_{\pm 0.0203}(20.0)$	$0.6652_{\pm 0.0344}(30.0)$	$0.8302_{\pm 0.0118}(20.0)$	0.7307
LocalE	$0.9174_{\pm 0.0044}(0.36)$	$0.3290_{\pm 0.0062}(0.44)$	$0.3967_{\pm 0.0041}(0.10)$	$0.5399_{\pm 0.0042}(1.52)$	0.5458
MDL	$0.9350_{\pm 0.0039}(1.82)$	$0.4031_{\pm 0.0046}(2.21)$	$0.4140_{\pm 0.0295}(0.50)$	$0.5520_{\pm 0.0060}(3.80)$	0.5760
GlobalE	$0.9263_{\pm 0.0362}(2.09)$	$0.4085_{\pm 0.0405}(2.64)$	$0.5136_{\pm 0.1091}(1.27)$	$0.6350_{\pm 0.1809}(2.40)$	0.6208
TopK+LocalE	$0.9393_{\pm 0.0024}(0.36)$	$0.4204_{\pm 0.0015}(0.44)$	$0.8044_{\pm 0.0109}(0.10)$	$0.9059_{\pm 0.0007}(1.52)$	0.7675
TopK+MDL	$0.9456_{\pm 0.0008}(1.82)$	$0.4809_{\pm 0.0014}(2.21)$	$0.8208_{\pm 0.0146}(0.50)$	$0.8740_{\pm 0.0089}(3.80)$	0.7803

Table 1: Bi-objective Evaluation for different configuration methods. The reported values represent the average accuracy and standard deviation on the original test sets across five seed runs. The DC cost is provided in parentheses ($\times 10000$). "Avg." signifies the average accuracy across the four evaluated tasks. The results are categorized into example selection methods, example ordering methods, and hybrid methods. More details are shown in Appendix B.

cost for methods across the same dataset and utilize either GPT2-xl or OPT-2.7B for demonstration configuration. The re-evaluation results, presented in Table 2, reveal several new phenomena. It is noted that the size of test sets in Table 2 is different from that in Table 1 for DC cost consistency.

Different comparison results from Table 1 could be found in Table 2. Notably, the inferior MDL in bi-objective evaluation yields better results than GlobalE with controlled DC cost, among the example ordering methods. We analyze that GlobalE performs better in Table 1 due to its higher DC cost. When the DC cost decreases, MDL makes it easier to find good examples for a given test input because GlobalE configurates the same demonstration for all the test instances while MDL selects a unique demonstration for each test instance.

DataModel and TopK consistently outperform the example ordering methods. A closer examination reveals that randomly selected examples in LocalE, MDL, and GlobalE lead to their inferiority, highlighting the critical importance of effective demonstration selection over ordering. This point is further confirmed by the performance improvement when MDL and LocalE are paired with TopK. Moreover, DataModel performs better than TopK across SST2 and SST5 but lags in Trec and Agnews. This can be attributed to Datamodel's preference for selecting label-balanced examples, whereas TopK selects examples based on semantic similarity. So TopK is more effective for tasks where semantically similar instances share similar labels.

4.3 Further Analysis

We further analyze how to trade-off between the example selection and ordering with a fixed DC cost. Comparing Table 1 and Table 2 reveals that the advantage of hybrid methods over example selection methods diminishes as the DC cost is controlled lower. This suggests that when DC costs are lower, the performance boost from considering ordering might decrease, inspiring us to prioritize example selection when DC cost is limited.

Based on the observation that example selection might outweigh ordering in §4.2, we investigate how to improve performance with a certain example selection method and fixed DC costs by adjusting the diversity of candidate demonstration. In Figure 3, we fix the DC cost by fixing the number of candidate sequences for validation and examine the effect of the diversity of candidate demonstration.

Models	SST2	SST5	Trec	AgNews	Avg.
GPT2-xl (1.5B)					
Random	0.6883 ± 0.0208	0.2087 ± 0.0211	0.3273 ± 0.0274	0.5500 ± 0.0081	0.4436
TopK	0.7333 ± 0.0000	0.3166 ± 0.0000	0.6933 ± 0.0000	0.8128 ± 0.0000	0.6390
Datamodel	0.7374 ± 0.0399	0.3571 ± 0.0299	0.5136 ± 0.0170	0.8060 ± 0.0339	0.6035
LocalE	0.7100 ± 0.0088	0.2460 ± 0.0375	0.3144 ± 0.0226	0.5500 ± 0.0360	0.4551
MDL	0.7211 ± 0.0092	0.2800 ± 0.0351	0.3646 ± 0.0280	0.5875 ± 0.0259	0.4883
GlobalE	0.7193 ± 0.1681	0.2760 ± 0.0580	0.3426 ± 0.0580	0.5600 ± 0.1168	0.4745
TopK+LocalE	0.7856 ± 0.0160	0.3078 ± 0.0126	0.7244 ± 0.0050	0.8133 ± 0.0252	0.6578
TopK+MDL	0.7840 ± 0.0059	0.3773 ± 0.0193	0.7300 ± 0.0092	0.8760 ± 0.0089	0.6818
OPT-2.7B					
Random	0.9173 ± 0.0155	0.3013 ± 0.0206	0.3447 ± 0.0227	0.5920 ± 0.0585	0.5388
TopK	0.9366 ± 0.0000	0.3633 ± 0.0000	0.7033 ± 0.0000	0.8900 ± 0.0000	0.7233
Datamodel	0.9588 ± 0.0081	0.4528 ± 0.0065	0.7116 ± 0.0222	0.8264 ± 0.0159	0.7374
LocalE	0.9167 ± 0.0173	0.3867 ± 0.0067	0.3455 ± 0.0214	0.6440 ± 0.0208	0.5732
MDL	0.9533 ± 0.0125	0.3927 ± 0.0458	0.3533 ± 0.0007	0.6867 ± 0.0152	0.5965
GlobalE	0.9240 ± 0.0256	0.3747 ± 0.0846	0.3534 ± 0.1264	0.6900 ± 0.1219	0.5855
TopK+LocalE	0.9360 ± 0.0098	0.3867 ± 0.0033	0.7300 ± 0.0067	0.8600 ± 0.0100	0.7282
TopK+MDL	0.9547 ± 0.0062	0.4373 ± 0.0116	0.7428 ± 0.0156	0.8840 ± 0.0089	0.7547

Table 2: Evaluation results after controlling DC cost for different ICL demonstration configuration methods. We run methods on the sampled small test sets across five seeds (refer to Appendix B for details). Except for zero DC cost for Random and TopK methods, it is set to 7200 across SST2, SST5, and Trec, and 2400 on AgNews. The results are categorized into example selection methods, example ordering methods, and hybrid methods.

The findings indicate that for less reliable selection methods such as Random, enriching demonstrations with a wider array of examples can lead to improved performance. Conversely, for more dependable selection methods like TopK, choosing fewer examples is often adequate.

5 Towards Zero Configuration Cost

Since the DC cost is also important when configurating ICL for very large language models (especially commercial LLM APIs such as ChatGPT and GPT-4) as presented in previous sections, we seek better ICL configuration methods that attain better trade-offs between accuracy and DC cost according to our two-dimensional evaluation. To this end, we first justify a nice property about the transferability across different LLMs. Then, based on the property we propose a strategy to achieve zero DC cost and implement variant ICL methods under this strategy.

5.1 Transferability on Optimized Demonstration

Intuitively, a optimized demonstration depends on the targeted task itself and thus it may be independent of the inference LLMs. This intuition further motivates us to raise a hypothesis: *the optimized demonstration by a configuration method over an LLM is transferable across different LLMs*, which provides a basis to achieve the strong ICL method with zero DC cost.

To verify the above hypothesis, we choose three configuration methods, DataModel, TopK+LocalE, and TopK+MDL, which achieve good trade-offs between accuracy and DC cost as evaluated in §4. By using each method, we then respectively optimize a demonstration for each LLM from a model set with different sizes (i.e., GPT2-medium (355M), GPT2-xl (1.5B), OPT-2.7B and OPT-6.7B). Finally, we measure its accuracy on a test set for each optimized demonstration.

The results are shown in Figure 2. We can see that: 1) the diagonal values are almost the highest in each line, this is because the configuration model and inference model are the same model. 2) non-diagonal values are very close to the diagonal values, which indeed verifies our hypothesis. This finding demonstrates that the optimized demonstration is transferable across different LLMs. To visualize the transferability property across different models, we examine the selected demonstrations with DataModel configured at different parameter scales in Appendix D.

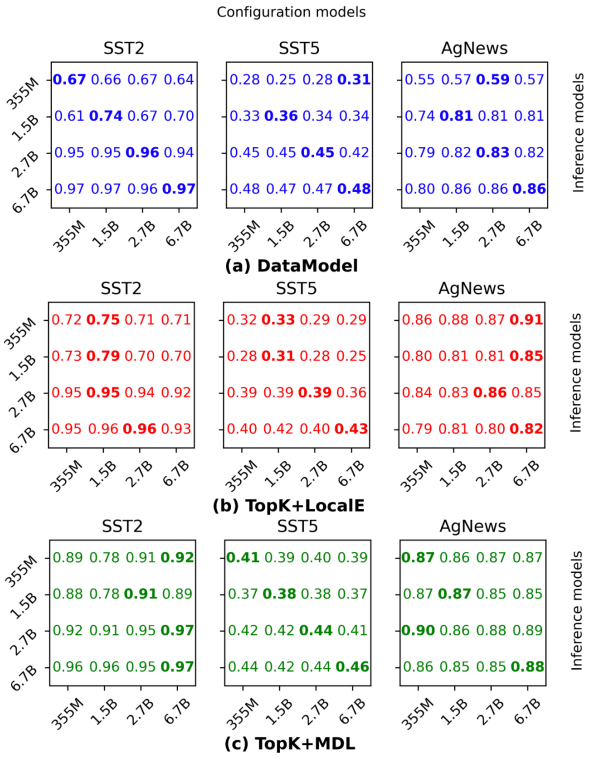

Figure 2: Transferability Property: the ordered demonstrations optimized by the configuration model (in the horizontal axis) are almost transferable across different inference models (in the vertical axis). The parameter scales of the models are GPT2-medium(355M), GPT2-XL(1.5B), OPT-2.7B, and OPT-6.7B.

Dataset	SST2	SST5	AgNews	DC Cost
Inference with OPT-6.7B				
DataModel	**0.971**	**0.485**	0.860	7200/2400
TopK+LocalE	0.926	0.429	0.824	7200/2400
TopK+MDL	0.971	0.458	**0.880**	7200/2400
DataModel(ours)	0.969	0.483	0.799	**0**
TopK+LocalE(ours)	0.950	0.395	0.790	**0**
TopK+MDL(ours)	0.957	0.435	0.864	**0**
DataModel+TopK+MDL	0.960	0.448	0.770	**0**
Inference with OPT-13B				
DataModel	**0.980**	**0.508**	**0.905**	7200/2400
TopK+LocalE	0.917	0.386	0.840	7200/2400
TopK+MDL	0.976	0.453	0.870	7200/2400
DataModel(ours)	0.963	0.496	0.833	**0**
TopK+LocalE(ours)	0.909	0.372	0.869	**0**
TopK+MDL(ours)	0.950	0.432	0.900	**0**
DataModel+TopK+MDL	0.970	0.460	0.850	**0**

Table 3: Comparison results with standard configuration methods. The configuration models employed in "ours" are GPT2-medium(355M), while the configuration models for baselines are the original inference model OPT-6.7B or OPT-13B.

5.2 Strategy to Reduce DC Cost

Thanks to the nice property of transferability on demonstration, it is straightforward to design a simple yet effective strategy to reduce the demonstration configuration cost. Specifically, unlike the standard ICL where the configuration model is exactly the same as the original inference model, *we instead employ another small language model for configuration and still run inference on the original model.* As long as the configuration model is small enough, its configuration cost is almost zero, i.e., the configuration process is very efficient and free of financial cost. This strategy is very general and can be applied on top of ICL methods for demonstration configuration if possible. [2]

In our experiments, we use GPT2-medium (355M) as the configuration model and apply our strategy to three notable methods, (TopK+LocalE, TopK+MDL, DataModel), which are denoted by TopK+LocalE(ours), TopK+MDL(ours) and DataMmodel(ours) to differentiate from their original versions. For a better trade-off between DC cost and accuracy, we further combine DataModel and Topk+MDL methods together and apply our strategy on top of it. Specifically, we first leverage DataModel to roughly select demonstrations and construct a pool, given its initial design intent to choose a demonstration subset with higher average accuracy (Chang and Jia, 2023). Subsequently, we employ Topk+MDL to select examples and arrange them to construct a demonstration for the downstream task. Finally, we implement the proposed strategy on top of it, leading to a new method denoted by "DataModel+Topk+MDL". We report the comparison results of these methods in Figure 3.

First, the results illustrate that our strategy effectively reduces the DC cost while achieving comparable accuracy to methods where the configuration model is the same as the inference model. In the scenarios where the DC cost is the top priority, our strategy is more efficient and could save 7200 or 2400 inferences on the large language models. Secondly, as the scale of the inference model increases, our strategy exhibits a reduced performance gap compared to the standard configuration methods, all while optimizing time efficiency. In certain instances, our methods even surpass some baseline approaches, highlighting their efficacy and their considerable potential when coupled with large language models. Third, the strategy demonstrates remarkable flexibility by seamlessly adapting to various language models and demonstration config-

[2]For the random method, this strategy is not applicable because there is no need to do so.

Dataset	SST2	SST5	Trec	AgNews
Inference with davinci-002				
Random	0.946	0.443	0.480	0.720
TopK+LocalE(ours)	0.960	**0.507**	0.763	0.860
TopK+MDL(ours)	0.957	0.493	**0.783**	**0.900**
DataModel+TopK+MDL	**0.963**	0.483	0.687	0.840
Inference with gpt-3.5-turbo				
Zero-shot	0.523	0.203	0.190	0.870
Random	0.963	0.426	0.603	0.890
TopK+LocalE(ours)	0.970	0.467	**0.830**	0.860
TopK+MDL(ours)	0.967	**0.510**	0.820	0.900
DataModel+TopK+MDL	**0.976**	0.483	0.793	**0.920**

Table 4: ICL with zero DC cost over closed-source davinci-002 and gpt-3.5-turbo.

uration methods.

To further substantiate the effectiveness of our strategy, we apply the selected examples derived from GPT2-medium(355M) on the top of closed-source LLMs, including davinci-002 and gpt-3.5-turbo. We didn't adapt TopK+LocalE and TopK+MDL to close-source LLMs due to their higher DC cost and financial cost deriving from the frequent calling of APIs. As delineated in Table 4, integrating different configuration methods with the zero DC cost strategy consistently outperforms zero-shot and random examples selection methods, thereby confirming its practical significance. On the one hand, the strategy with zero DC cost offers a more economical option in resource-limited scenarios, saving the expenses for querying LLMs hundreds or thousands of times in other methods. As the scale of LLMs increases or its associated costs rise, the potential of our strategy becomes even more pronounced. On the other hand, for tasks that are difficult for LLMs to understand, the proposed strategy holds promise in swiftly identifying optimal examples to augment LLM performance.

6 Related Work

Current research on ICL primarily concentrates on example selection, example ordering, demonstration formats (Dong et al., 2022), etc. This paper focuses on the evaluation of demonstration configuration and introduces a two-dimensional evaluation for a fair comparison of these methods.

Studies for demonstration example selection aim to retrieve examples of high quality for a given test instance. Existing methods mainly concentrate on the similarity (Wang et al., 2022a; Ye et al., 2023b) and diversity (Zhang et al., 2022c; Naik et al., 2023; Ma et al., 2024) of selected examples (Liu et al., 2022). Some of them select demonstration example according to off-the-shell metric, such as BM25 (Wang et al., 2022a), text vector similarity (Gao et al., 2021; Liu et al., 2022; Ye et al., 2023b; Wang et al., 2023b; Li and Qiu, 2023b), mutual information (Sorensen et al., 2022), determinantal point process (Wu et al., 2022; Yang et al., 2023a), perplexity (Gonen et al., 2023), and skill-based KNN (An et al., 2023). To better adapt to various downstream tasks, different loss functions are proposed to train more effective example retrievers tailored to specific downstream objectives, including contrastive loss (Rubin et al., 2022; Luo et al., 2023) and negative loss (Karpukhin et al., 2020; Li et al., 2023). Moreover, reinforcement learning techniques (Zhang et al., 2022b; Scarlatos and Lan, 2023) are also adopted for more adaptive training. Chang and Jia (2023) trains datamodels to select based on feedback from LLMs specific to certain tasks. Ye et al. (2023a) trains an example retriever and considers the interaction among them. Nguyen and Wong (2023); Li and Qiu (2023a); Li et al. (2024) distill representative examples from the corpra. Wang et al. (2024) treats LLMs as latent variables and pre-trains concept tokens for demonstration selection. Iter et al. (2023) selects demonstrations with the lowest cross-entropy difference with a test example. More recently, Chen et al. (2023) prompts large language models to generate demonstration examples instead of relying on retrieval methods.

Since Lu et al. (2022) demonstrates that the order of demonstration significantly influences the performance of ICL, numerous methods have emerged to determine the optimal order for different tasks. To name some, Wu et al. (2022) evaluate various permutations of demonstrations through the lens of information compression. (Liu et al., 2022) arrange examples based on their similarities with the input instance. Xu et al. (2024) rank selected demonstrations according to label distributions. Xiong et al. (2023) employs a PCA-based re-ranking method to reorder retrieved examples. Zhang et al. (2024) proposed Batch-ICL from the perspective of meta-gradient aggregation.

7 Conclusion

Although existing ICL methods have succeeded across various downstream NLP tasks, they overlook the evaluation of the demonstration configura-

tion cost, which induces an unfair comparison of ICL methods. In this paper, we first illustrate the positive co-relationship between the performance and the DC cost. Then we call for a two-dimension evaluation considering both of the aspects. Based on the new evaluation results, a simple yet effective strategy is proposed to reduce the DC cost. Experiments proved such a strategy achieves better trade-offs between the proposed two dimensions.

Limitations

This study explored the relationships between ICL performance and DC cost, calling for a two-dimensional evaluation paradigm. Nonetheless, several limitations persist. Firstly, our investigation relied on generic ICL templates, overlooking potentially more refined alternatives like Mot (Li and Qiu, 2023b) and self-instruct (Wang et al., 2022b). Secondly, we only conducted experiments on text classification tasks and did not consider more challenging tasks like text generation tasks, which serve as our future work. Thirdly, due to time constraints, we did not conduct experiments that consume greater DC costs in the bi-objective evaluation for comparison, such as methods that require querying LLMs tens of thousands of times.

Acknowledgements

The research work is supported by National Key RD Plan No. 2022YFC3303303, the Project of Youth Innovation Promotion Association CAS, Beijing Nova Program 20230484430, the Innovation Funding of ICT, CAS under Grant No. E461060. This study is also supported by grants from the Major Key Project of PCL (Grant Number: PCL2023A09).

References

Shengnan An, Bo Zhou, Zeqi Lin, Qiang Fu, Bei Chen, Nanning Zheng, Weizhu Chen, and Jian-Guang Lou. 2023. Skill-based few-shot selection for in-context learning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13472–13492, Singapore. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Ting-Yun Chang and Robin Jia. 2023. Data curation alone can stabilize in-context learning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8123–8144.

Wei-Lin Chen, Cheng-Kuang Wu, Yun-Nung Chen, and Hsin-Hsi Chen. 2023. Self-ICL: Zero-shot in-context learning with self-generated demonstrations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 15651–15662, Singapore. Association for Computational Linguistics.

Arghavan Moradi Dakhel, Amin Nikanjam, Foutse Khomh, Michel C Desmarais, and Hironori Washizaki. 2024. An overview on large language models. *Generative AI for Effective Software Development*, pages 3–21.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, and Zhifang Sui. 2022. A survey on in-context learning. *arXiv preprint arXiv:2301.00234*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830.

Hila Gonen, Srini Iyer, Terra Blevins, Noah A Smith, and Luke Zettlemoyer. 2023. Demystifying prompts in language models via perplexity estimation. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Peter D Grünwald. 2007. *The minimum description length principle*. MIT press.

Or Honovich, Uri Shaham, Samuel Bowman, and Omer Levy. 2023. Instruction induction: From few examples to natural language task descriptions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1935–1952.

Eduard Hovy, Laurie Gerber, Ulf Hermjakob, Chin-Yew Lin, and Deepak Ravichandran. 2001. Toward semantics-based answer pinpointing. In *Proceedings of the first international conference on Human language technology research*.

Andrew Ilyas, Sung Min Park, Logan Engstrom, Guillaume Leclerc, and Aleksander Madry. 2022. Datamodels: Predicting predictions from training data. In *Proceedings of the 39th International Conference on Machine Learning*.

Dan Iter, Reid Pryzant, Ruochen Xu, Shuohang Wang, Yang Liu, Yichong Xu, and Chenguang Zhu. 2023. In-context demonstration selection with cross entropy

difference. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 1150–1162, Singapore. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781.

Xiaonan Li, Kai Lv, Hang Yan, Tianyang Lin, Wei Zhu, Yuan Ni, Guotong Xie, Xiaoling Wang, and Xipeng Qiu. 2023. Unified demonstration retriever for in-context learning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4644–4668.

Xiaonan Li and Xipeng Qiu. 2023a. Finding support examples for in-context learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 6219–6235.

Xiaonan Li and Xipeng Qiu. 2023b. Mot: Pre-thinking and recalling enable chatgpt to self-improve with memory-of-thoughts. *arXiv preprint arXiv:2305.05181*.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Yichuan Li, Xiyao Ma, Sixing Lu, Kyumin Lee, Xiaohu Liu, and Chenlei Guo. 2024. Mend: Meta demonstration distillation for efficient and effective in-context learning. *The Twelfth International Conference on Learning Representations*.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2022. What makes good in-context examples for GPT-3? In *Proceedings of Deep Learning Inside Out (DeeLIO 2022): The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 100–114, Dublin, Ireland and Online. Association for Computational Linguistics.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8086–8098.

Man Luo, Xin Xu, Zhuyun Dai, Panupong Pasupat, Mehran Kazemi, Chitta Baral, Vaiva Imbrasaite, and Vincent Y Zhao. 2023. Dr. icl: Demonstration-retrieved in-context learning. *arXiv preprint arXiv:2305.14128*.

Huan Ma, Changqing Zhang, Yatao Bian, Lemao Liu, Zhirui Zhang, Peilin Zhao, Shu Zhang, Huazhu Fu, Qinghua Hu, and Bingzhe Wu. 2024. Fairness-guided few-shot prompting for large language models. *Advances in Neural Information Processing Systems*, 36.

Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. 2024. Large language models: A survey. *arXiv preprint arXiv:2402.06196*.

Ranjita Naik, Varun Chandrasekaran, Mert Yuksekgonul, Hamid Palangi, and Besmira Nushi. 2023. Diversity of thought improves reasoning abilities of large language models. *arXiv preprint arXiv:2310.07088*.

Tai Nguyen and Eric Wong. 2023. In-context example selection with influences. *arXiv preprint arXiv:2302.11042*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992.

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2022. Learning to retrieve prompts for in-context learning. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2655–2671.

Alexander Scarlatos and Andrew Lan. 2023. Reticl: Sequential retrieval of in-context examples with reinforcement learning. *arXiv preprint arXiv:2305.14502*.

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

Taylor Sorensen, Joshua Robinson, Christopher Michael Rytting, Alexander Glenn Shaw, Kyle Jeffrey Rogers, Alexia Pauline Delorey, Mahmoud Khalil, Nancy Fulda, and David Wingate. 2022. An information-theoretic approach to prompt engineering without ground truth labels. *arXiv preprint arXiv:2203.11364*.

Lean Wang, Lei Li, Damai Dai, Deli Chen, Hao Zhou, Fandong Meng, Jie Zhou, and Xu Sun. 2023a. Label words are anchors: An information flow perspective for understanding in-context learning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9840–9855.

Liang Wang, Nan Yang, and Furu Wei. 2023b. Learning to retrieve in-context examples for large language models. *arXiv preprint arXiv:2307.07164*.

Shuohang Wang, Yichong Xu, Yuwei Fang, Yang Liu, Siqi Sun, Ruochen Xu, Chenguang Zhu, and Michael Zeng. 2022a. Training data is more valuable than you think: A simple and effective method by retrieving from training data. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3170–3179.

Xinyi Wang, Wanrong Zhu, Michael Saxon, Mark Steyvers, and William Yang Wang. 2024. Large language models are latent variable models: Explaining and finding good demonstrations for in-context learning. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2022b. Self-instruct: Aligning language models with self-generated instructions. *arXiv preprint arXiv:2212.10560*.

Zhiyong Wu, Yaoxiang Wang, Jiacheng Ye, and Lingpeng Kong. 2022. Self-adaptive in-context learning: An information compression perspective for in-context example selection and ordering. *arXiv preprint arXiv:2212.10375*.

Jing Xiong, Zixuan Li, Chuanyang Zheng, Zhijiang Guo, Yichun Yin, Enze Xie, YANG Zhicheng, Qingxing Cao, Haiming Wang, Xiongwei Han, et al. 2023. Dq-lore: Dual queries with low rank approximation re-ranking for in-context learning. In *The Twelfth International Conference on Learning Representations*.

Zhichao Xu, Daniel Cohen, Bei Wang, and Vivek Srikumar. 2024. In-context example ordering guided by label distributions. *arXiv preprint arXiv:2402.11447*.

Zhao Yang, Yuanzhe Zhang, Dianbo Sui, Cao Liu, Jun Zhao, and Kang Liu. 2023a. Representative demonstration selection for in-context learning with two-stage determinantal point process. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5443–5456, Singapore. Association for Computational Linguistics.

Zhe Yang, Damai Dai, Peiyi Wang, and Zhifang Sui. 2023b. Not all demonstration examples are equally beneficial: Reweighting demonstration examples for in-context learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 13209–13221.

Jiacheng Ye, Zhiyong Wu, Jiangtao Feng, Tao Yu, and Lingpeng Kong. 2023a. Compositional exemplars for in-context learning. In *International Conference on Machine Learning*, pages 39818–39833. PMLR.

Xi Ye, Srinivasan Iyer, Asli Celikyilmaz, Veselin Stoyanov, Greg Durrett, and Ramakanth Pasunuru. 2023b. Complementary explanations for effective in-context learning. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 4469–4484.

Kaiyi Zhang, Ang Lv, Yuhan Chen, Hansen Ha, Tao Xu, and Rui Yan. 2024. Batch-icl: Effective, efficient, and order-agnostic in-context learning. *arXiv preprint arXiv:2401.06469*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022a. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.

Yiming Zhang, Shi Feng, and Chenhao Tan. 2022b. Active example selection for in-context learning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9134–9148.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. 2022c. Automatic chain of thought prompting in large language models. In *The Eleventh International Conference on Learning Representations*.

A Summarization of Demonstration Configuration Methods

In this section, we provide a detailed elaboration on the selected demonstration configuration methods and present a summary of these methods in Table 5. A more comprehensive description of these methods can be found in their initial papers.

Model	Selection	Ordering	Val. Data
Example Selection Methods			
Random	✗	✗	✗
TopK	✓	✗	✗
DataModel	✓	✗	✓
Example Ordering Methods			
GlobalE	✗	✓	✓
LocalE	✗	✓	✗
MDL	✗	✓	✗
Hybrid Methods			
TopK+LocalE	✓	✓	✗
TopK+MDL	✓	✓	✗

Table 5: The summarization of the selected configuration methods. Val. Data marked with ✗ means that the method validates a candidate demonstration on the test set without visible gold labels.

Example Selection Methods *Random* randomly selects examples from the training set $\mathcal{D}$. *TopK* selects the top K examples based on similarities calculated using the Sentence Transformer (Reimers and Gurevych, 2019).

$$\text{sim}(x, d_i) = f_{st}(x) \cdot f_{st}(d_i)^\intercal \tag{2}$$

where f_{st} represents the SentenceBERT (Reimers and Gurevych, 2019) function that maps texts into embeddings. *DataModel* pre-trained a datamodel (Ilyas et al., 2022) to select examples.

$$\text{sim}(x, d_i) = \sum_{x \in D_{val}} \sum_{j} 1\{\hat{w}_x(d_i, p_i) > 0\} \tag{3}$$

$\hat{w}$ is the weight from the pre-trained data model, which denotes the effectiveness of example d_i when it appears in the permutation p_i. It is noteworthy that a pre-trained datamodel can rank examples without visiting a specific test input.

Example Ordering Methods *GlobalE* aims to mitigate the issue of highly imbalanced predictions. It calculates a global entropy based on the prediction results on the downstream task:

$$\text{e}(p_i) = \sum_{v \in V} -\text{p}_i^v \log \text{p}_i^v, \tag{4}$$

p_i^v is the proportion of v-class predictions using p_i as prompt on a validation set. *LocalE* computes a local entropy to avoid model overconfidence.

$$\text{e}(p_{i,j}) = \frac{-\sum_j \sum_{v \in \mathcal{V}} \text{p}_{i,j}^v \log \text{p}_{i,j}^v}{|D_{val}|} \tag{5}$$

$\text{p}_{i,j}^v$ is the probability of class v on the j-th validation example using p_i as prompt. For the k-th test input x, *MDL* evaluates the codelength metric as follows:

$$\begin{aligned} \text{e}(p_{i,k}) &= -\sum_{v \in \mathcal{V}} q(v|\mathcal{V}) \log_2 \text{p}_{i,k}^v \\ &= -\sum_{v \in \mathcal{V}} \text{p}_{i,k}^v \log_2 \text{p}_{i,k}^v \end{aligned} \tag{6}$$

$\text{p}_{i,k}^v$ is the probability of class v when p_i serves as the prompt of the k-th test instance.

Hybrid Methods *TopK+LocalE* uses *TopK* and *LocalE* for example selection and ordering. *TopK+MDL* combines *TopK* and *MDL*.

B Experimental Settings

General settings In this paper, we employ four open-source language models, namely GPT2-medium, GPT2-xl (Radford et al., 2019), OPT-2.7B, OPT-6.7B (Zhang et al., 2022a), and two closed-source models for experiments (e.g., davinci-002, gpt3.5-turbo). We select SST2 (Socher et al., 2013), SST5 (Socher et al., 2013), Trec (Li and Roth, 2002; Hovy et al., 2001), and AgNews (Zhang et al., 2015) to evaluate all the demonstration configuration methods, representing 2-class, 5-class, 6-class, and 4-class classification tasks, respectively. If a dataset does not contain a validation set initially, we randomly select examples from the training set to construct it. For each of these datasets, we conducted few-shot experiments using 4, 5, 6, and 4 examples from the training set, respectively. All reported results represent average accuracy over 5 distinct seeds. The language models were executed on hardware configurations including Tesla V100-PCIE-32GB, NVIDIA GeForce RTX 4090, or NVIDIA A800 80GB PCIe, and access to the closed-source models was facilitated through APIs [3].

Templates for demonstration construction We utilize the first template for the sentiment classification task (SST2 and SST5) and the second template for the topic classification task (Trec and AgNews).

`(1) Review: <X> Sentiment: <Y>`

`(2) Article: <X> Answer: <Y>`

where `<X>` denotes the input text and `<Y>` is the corresponding label.

Settings for pilot study In our pilot study, we assess six methods across SST2 and SST5 datasets, as depicted in Figure 1. The demonstration examples are drawn from the original training set, and all reported results are evaluated on the respective original test sets for the four tasks. Notably, GlobalE and DataModel utilize the original validation set for demonstration validation, whereas LocalE, MDL, TopK+LocalE, and TopK+MDL directly employ the corresponding test set for validation without visible gold labels, as detailed in Table 5. Regarding the specific settings of sub-figures, the x-axes of GlobalE, LocalE, and MDL share the same scales, while DataModels, TopK/TopK+LocalE, and TopK+MDL are set to the same scales. We select examples based on the shots required for

[3] https://openai.com/api/

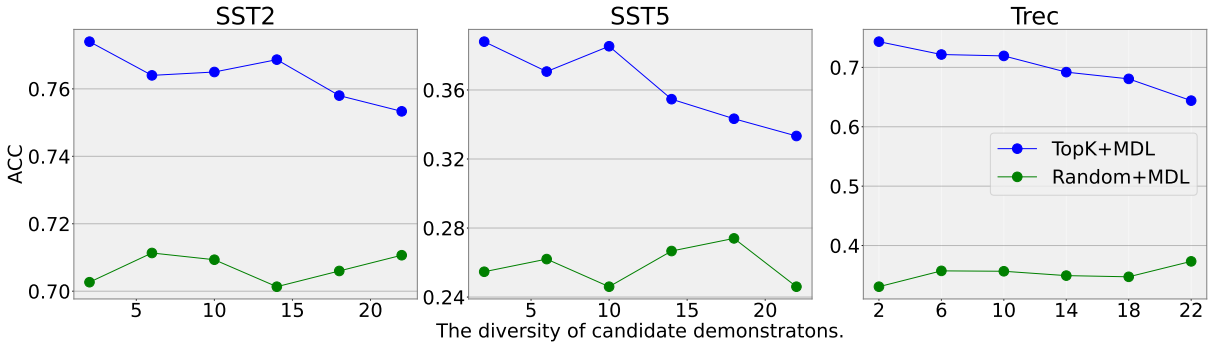

Figure 3: The relationships between performance and the diversity of candidate demonstrations. Diversity measures the probability of various examples within the demonstrations being ordered. For instance, when the x-axis is set to 5, we choose 5 examples using either the random or TopK method, and then sample 24 sequences from the full permutation of these 5 examples for validation. When the number of selected sequences remains constant, increasing the data on the x-axis leads to a greater variety of examples included in the selected sequence.

the corresponding tasks for Random, TopK, GlobalE, LocalE, and TopK+LocalE. For MDL and TopK+MDL, adhering to their initial settings, we select a pool containing 10 candidate examples and further construct demonstrations by randomly selecting examples and ordering from the pool according to the shots. For DataModel, we sample 100 and 200 data points from the original validation sets of SST2 and SST5, respectively. Next, for both tasks, we sample 4 or 5 examples from the training set 500 times and generate two random permutations of the selected examples each time to construct candidate demonstrations. These constructed demonstrations are then inputted into LLMs, resulting in output logits across the downstream task. Subsequently, the candidate demonstrations and their corresponding outputs are utilized for training a datamodel.

Settings for two-dimensional evaluation The experimental settings of results in Table 1 largely mirror those of the pilot study, with the exception of varying Dynamic Control (DC) costs. In Table 2, we standardize the DC costs across different methods by adjusting the number of example permutations and validation data. Since different methods employ varied example selection approaches and validate demonstrations on either the validation or test set, it's crucial to maintain consistency in the number of validation data. Inspired by existing studies (Chang and Jia, 2023), we randomly select 300 data from each task's training, validation, and test sets to obtain datasets used in Table 2, keeping their original label distribution as much as possible. Different from other tasks, the number of selected data on AgNews is set to 100 for more efficient inference. Due to inconsistencies in the test sets, comparing scores in identical positions between Table 2 and Table 1 lacks significance.

Settings for experiments with zero cost For more efficient experiments, the results in §5 are also evaluated on the sample small datasets. Particularly, for results in Table 4, we set the `temperature` of the davinci-002 and gpt-3.5-turbo to 0 and use a simple instruct for the Zero-shot method as follows:

SST2: `Classify the following text into categories: negative, positive.`

SST5: `Classify the following text into categories: terrible, bad, okay, good, great.`

Trec: `Classify the following text into categories: world, sports, business, technology.`

AgNews: `Classify the following text into categories: expression, entity, description, human, location, number.`

C The Effect of the Diversity of Candidate Demonstrations

This section demonstrated that a better trade-off could realized by controlling the diversity of candidate demonstrations. The experimental results are shown in igure 3.

D The Overlap of Demonstrations Selected by Different LMs

To better visualize the transferability property across different models, we examine the selected demonstrations with DataModel configured at dif-

Model	SST2	SST5	Trec	AgNews
OPT-6.7B	26%	14%	12%	54%
OPT-13B	18%	18%	14%	60%
OPT-6.7B&13B	14%	10%	2%	42%

Table 6: The overlap between the selected demonstrations utilizing DataModel configured with GPT2-medium and those employing OPT-6.7B or OPT-13B models.

ferent parameter scales. Table 6 shows that a small model might optimize the same demonstrations with a large model from the same example pool, which proves that small models are indeed capable of curating a high-quality demonstration. Table 7 and Table 8 show the overlapped examples selected by DataModel based on GPT2-medium, OPT-2.7B, and OPT-13B.

Text	Label
But seriously, folks, it doesn't work.	negative
Little more than a stylish exercise in revisionism whose point ... is no doubt true, but serves as a rather thin moral to such a knowing fable.	negative
should have gone straight to video.	negative
... hokey art house pretension.	negative
average B-movie with no aspirations to be anything more.	negative
Plunges you into a reality that is, more often than not, difficult and sad, and then, without sentimentalizing it or denying its brutality, transforms that reality into a lyrical and celebratory vision.	positive
intriguing look at the French film industry during the German occupation; its most delightful moments come when various characters express their quirky inner selves.	positive

Table 7: Overlapped examples selected by DataModel based on GPT2-medium, OPT-2.7B, and OPT-13B across SST2.

Text	Label
big whoop, nothing new to see, zero thrills, too many flashbacks and a choppy ending make for a bad film.	terrible
exploitative and largely devoid of the depth or sophistication that would make watching such a graphic treatment of the crimes bearable.	bad
The gags are often a stitch.	good
there's a lot to recommend read my lips.	good
he nonetheless appreciates the art and reveals a music scene that transcends culture and race.	great

Table 8: Overlapped examples selected by DataModel based on GPT2-medium, OPT-2.7B, and OPT-13B across SST5.