Prompt Optimization with EASE? Efficient Ordering-aware Automated Selection of Exemplars

Zhaoxuan Wu*§, Xiaoqiang Lin*†, Zhongxiang Dai‡, Wenyang Hu§†,
Yao Shu‖, See-Kiong Ng§†, Patrick Jaillet‡, Bryan Kian Hsiang Low†
Institute of Data Science, National University of Singapore, Republic of Singapore§
Dept. of Computer Science, National University of Singapore, Republic of Singapore†
LIDS and EECS, Massachusetts Institute of Technology, USA‡
Guangdong Lab of AI and Digital Economy (SZ)‖
{wu.zhaoxuan, xiaoqiang.lin, wenyang.hu}@u.nus.edu§
seekiong@nus.edu.sg§
lowkh@comp.nus.edu.sg†
{daizx,jaillet}@mit.edu‡
shuyao@gml.ac.cn‖

Abstract

Large language models (LLMs) have shown impressive capabilities in real-world applications. The capability of *in-context learning* (ICL) allows us to adapt an LLM to downstream tasks by including input-label exemplars in the prompt without model fine-tuning. However, the quality of these exemplars in the prompt greatly impacts performance, highlighting the need for an effective automated exemplar selection method. Recent studies have explored retrieval-based approaches to select exemplars tailored to individual test queries, which can be undesirable due to extra test-time computation and an increased risk of data exposure. Moreover, existing methods fail to adequately account for the impact of exemplar ordering on the performance. On the other hand, the impact of the *instruction*, another essential component in the prompt given to the LLM, is often overlooked in existing exemplar selection methods. To address these challenges, we propose a novel method named EASE, which leverages the hidden embedding from a pre-trained language model to represent ordered sets of exemplars and uses a neural bandit algorithm to optimize the sets of exemplars *while accounting for exemplar ordering*. Our EASE can efficiently find an ordered set of exemplars that *performs well for all test queries* from a given task, thereby eliminating test-time computation. Importantly, EASE can be readily extended to *jointly optimize both the exemplars and the instruction*. Through extensive empirical evaluations (including novel tasks), we demonstrate the superiority of EASE over existing methods, and reveal practical insights about the impact of exemplar selection on ICL, which may be of independent interest. Our code is available at https://github.com/ZhaoxuanWu/EASE-Prompt-Optimization.

1 Introduction

Large language models (LLMs) have recently drawn significant attention and have been widely deployed in various real-world scenarios [18, 20, 40] due to their strong capabilities. Of note, a particularly impressive capability of LLMs is *in-context learning* (ICL): LLMs can learn from a

---

* Equal contribution.

handful of input-label demonstrations (i.e., *data exemplars*) included in its prompt to perform a downstream task [2, 15]. ICL allows us to adapt an LLM to a downstream task without fine-tuning the model parameters [13, 21]. However, the ICL performance is heavily dependent on the data exemplars in the prompt [1, 15, 29]. Therefore, to maximize the ICL performance, it is of paramount importance to carefully select the set of exemplars. Unfortunately, exemplar selection for ICL is challenging because the mechanism of ICL is complicated and unknown, and this difficulty is further aggravated for black-box LLMs to which we only have API access (e.g., GPT-4 [24], Gemini Ultra [32]).

A large number of existing works on exemplar selection for ICL have considered *retrieval-based* methods [1, 15, 29]. Specifically, they aim to develop a retriever model such that for every test query, the retriever model retrieves the corresponding best set of exemplars tailored to this particular query [1, 38]. Therefore, at test time, the retriever needs to be deployed for *every test query*, which could incur significant additional computational overheads especially when the number of test queries is large [17]. This is likely to hinder their ease of adoption in practice because of the prolonged response time and the additional complexity from running the retriever for every query. Moreover, another drawback of these retrieval-based methods is that they may lead to increased privacy risks. Compared to using a fixed set of exemplars for all test queries submitted to the LLM provider (e.g., via an API), using a different set of exemplars for every test query may lead to greater data exposure, i.e., more data exemplars being exposed to the LLM provider. Privacy issues have become increasingly prominent in discussions surrounding LLMs [22], with evidence of user data being output by these models [36]. Therefore, retrieval-based methods are undesirable in scenarios where such privacy concerns are important considerations. Given these two important shortcomings of retrieval-based methods, it is imperative to develop exemplar selection methods that generalize to test queries and are capable of choosing a fixed set of exemplars that perform well for all test queries for a task.

Another major limitation of existing exemplar selection methods (including retrieval-based and other methods) is their inability to account for the impact of the *ordering of the exemplars* within a subset [16, 25, 41]. Specifically, to select a subset of k exemplars, previous works have either relied on simple heuristics (e.g., selecting the top-k exemplars based on the score from a retriever [17]) or used subset selection techniques which are able to account for the inter-relationships among the exemplars within a subset [3, 9, 11, 12, 38]. Due to the computational complexity caused by the combinatorial search space, these works based on subset selection have often employed an iterative greedy strategy to sequentially select the subset of exemplars. It is also non-trivial to extend these existing works to consider exemplar ordering. However, it has been repeatedly verified that the ordering of the exemplars has a significant impact on the performance of ICL [16, 25, 41]. To the best of our knowledge, the impact of the exemplar ordering during the selection process remains inadequately addressed by these previous works that relied on simple heuristics (e.g., random ordering) and approximations (e.g., greedy strategy).

In addition, when optimizing the set of exemplars to improve the performance of the LLM, existing methods have often ignored the impact of the *instruction*, which is another essential component in the prompt given to the LLM [14]. Specifically, previous works on exemplar selection often use a fixed pre-determined instruction or do not include any instruction at all in the prompt [3, 12, 15, 29]. Meanwhile, existing works on instruction optimization for LLMs typically include a fixed manually selected set of exemplars in the prompt [6, 10, 14, 37] or simply adopt the zero-shot setting (i.e., without using any exemplar) [4, 8]. However, these two lines of work have separately demonstrated that both the exemplars and the instruction have significant impacts on the performance of the LLM. Therefore, the existing works that optimize these two components separately are unable to account for their interactions, which may be crucial for further boosting the performance of LLMs. This leaves considerable untapped potential in enhancing the performance of the LLM through ICL, which can be achieved by *jointly optimizing the instruction and the exemplars*.

In this work, we propose a novel exemplar selection algorithm that addresses the above-mentioned challenges faced by existing works (with detailed discussions of related work in Sec. 2). We formulate exemplar selection as a black-box optimization problem, in which every input in the domain corresponds to a sequence of k exemplars and its corresponding output represents the ICL performance achieved by including this sequence in the prompt for the LLM. Given this formulation, for every sequence of k exemplars in the domain, we adopt the embedding from a powerful pre-trained language model as its continuous representation, and train a neural network (NN) to predict its ICL performance. Based on the trained NN, we use a neural bandit algorithm to find the optimal

exemplar sequence in a query-efficient manner. Our <u>E</u>fficient ordering-aware <u>A</u>utomated <u>S</u>election of <u>E</u>xemplars (EASE) algorithm offers several significant benefits:

- Our EASE can *find an efficacious sequence of k exemplars* (which performs well for all test queries from a task) in a *query-efficient manner* (i.e., it only needs to test a small number of exemplar sequences). This can mainly be attributed to our algorithmic design, which allows us to use neural bandits to balance the *exploration* of the space of exemplar sequences and the *exploitation* of the predicted performance from the NN in a principled way. Importantly, in contrast to retrieval-based methods, our EASE *requires no test-time computation* to select test query-specific exemplars.
- Our EASE naturally *takes into account the ordering of the exemplars* when maximizing the ICL performance. This is because, for the same subset of exemplars, a different ordering leads to a different pre-trained embedding, which allows the trained NN (that takes the embedding as input) to predict the performances of different orderings of these exemplars. We have made our EASE computationally feasible for large search spaces of exemplar sequences using a technique based on optimal transport (OT), which reduces the computational cost of EASE while preserving its strong performance by imposing an implicit preference towards exemplars that are more relevant to the task (Sec. 4.2).
- Our EASE is readily extended to *jointly optimize the exemplars and instruction* in the prompt. This is achieved by augmenting the domain of exemplar sequences with the instruction (Sec. 4.3).

These advantages of our EASE algorithm allow it to significantly boost the performance of ICL. To empirically validate this, we compare our EASE algorithm with a comprehensive suite of baselines in a variety of tasks. To begin with, we show that our EASE consistently outperforms previous baselines in large number of benchmark tasks (Sec. 5.1). Next, by using our EASE algorithm in a novel experiment, we unveil an interesting insight about ICL: The selection of exemplars is more important when the LLM has less knowledge about the task (Sec. 5.2). Based on this insight, we design a set of novel experiments in which the selection of exemplars has an important impact on the ICL performance, and use these experiments to further demonstrate the superiority of our EASE (Sec. 5.3). Furthermore, we showcase the ability of our EASE to jointly optimize the exemplars and the instruction to enhance the performance of the LLM even further (Sec. 5.4). We also included a retrieval-based extension of EASE to deal with large exemplar set sizes (Sec. 5.5). Of note, our novel experimental designs, as well as the insights from them, *may be of independent interest for future works on ICL and beyond*.

2   Related work

Retrieval-based methods. This approach utilizes trainable exemplar retrievers to select exemplars depending on the text sequence of each individual test sample [38, 1]. Liu et al. [15] first proposed a similarity-based (e.g., negative Euclidean distance or cosine similarity) retrieval strategy on the sentence embeddings of the exemplars for ICL. Gao et al. [7] further enhanced the above by emphasizing on exemplars with top-2 labels that the LLM is most uncertain about. Adapting retrievers to specific tasks, Rubin et al. [29] learned a retrieval model from evaluating individual exemplars by their 1-shot ICL performance on validation data. Improving on the limitation of the above methods that exemplars are considered in isolation (i.e., independently), Ye et al. [38] proposed to retrieve a set of exemplars jointly by examining their joint probability to capture inter-relationships. Likewise, Levy et al. [11] retrieved a set by selecting diverse exemplars that collectively cover all of the output structures. Gupta et al. [9] instead generalized the individual metrics to a set-level metric through a submodular function, which is well-suited for optimization via greedy algorithms. However, retrieval-based methods are characterized by varying exemplars for each test sample, which do not align with our practical setting of maintaining a fixed set of exemplars for the entire task. Also, another common drawback is that heuristics are typically required to order the exemplars in the retrieved set.

Selection of a fixed set of exemplars. Having a fixed set of exemplars offers practical and privacy-related advantages, such as the ease of implementation and reduced data exposure [17, 22]. Wang et al. [35] proposed to train a small LLM as a latent variable model using output token logits from independent exemplars, then forming a fixed exemplar set to directly transfer to target models. Similarly, Chang and Jia [3] developed a data model for each validation sample and introduced an aggregate metric to select subsets. Li and Qiu [12] proposed to filter and search for best exemplars using an informativeness metric derived from LLM output logits on classification tasks. However, these works are restricted to classification tasks. The closest to our work is that of Zhang et al. [39], which used reinforcement learning to actively select exemplars. In a similar setting to ours, Nguyen

and Wong [23] used influence to select the most influential exemplars and order them arbitrarily to form the subset. However, both methods perform worse than our algorithm in our experiments.

Instruction optimization. Another related direction for enhancing the performance of LLMs is instruction optimization. Focusing on instructions within the prompt, evolutionary algorithms and zeroth-order optimization algorithms are gaining popularity in refining the prompts for black-box LLMs [4, 14, 8, 6, 37, 10]. Specifically, some studies [14, 6, 37, 10] maintained a constant set of exemplars throughout the process of prompt optimization, whereas others [4, 8] ignored the consideration of exemplars altogether by adopting a zero-shot setting. It is therefore imperative to develop an effective exemplar selection method for black-box LLMs.

3 Problem setting

We are given a set of data exemplars $D = \{e_i = (x_i, y_i)\}_{i=1}^n$ of size n, where x_i and y_i correspond to the input and output text, respectively. The data exemplars describe a downstream task, and we aim to select k of them to form the optimal in-context exemplars sequence E for accurate output generation when prompting a black-box LLM $f(\cdot)$. Due to the sequential nature of the natural language input, the exemplar sequence is ordered such that $E = (e_1, e_2, \ldots, e_k)$ where e_i denotes the i-th exemplar chosen. For every input x, we prepend it with a sequence of exemplars E to generate a response $\hat{y}$ from the black-box LLM (e.g., through calling the API), following

$$\hat{y} = f([\underbrace{e_1, e_2, \ldots, e_k}_{\text{context}}, x]) = f([E, x]) \ .$$

Here, the number of exemplar k in the context can be determined by the future budget of inference in practice. A larger k corresponds to a longer sequence of context tokens to be prepended to the test input x during inference, which leads to higher query costs (e.g., associated with the API calls). Alternatively, it is common to decide k depending on the context window size of the target LLM $f(\cdot)$.

Since the model architecture and the internal working of the black-box LLM $f(\cdot)$ is unattainable, we formulate the exemplar selection as a black-box optimization problem over the space of permutations $\Omega \triangleq \{E : |E| = k\}$ of size n^k (i.e., having n choices for each of the k positions in the sequence),

$$\max_{E \in \Omega} F(E) \triangleq \mathbb{E}_{(x,y) \in D_V}[s(f(E, x), y)] \tag{1}$$

where $s(\cdot, \cdot)$ is a score function for the output against the ground truth, D_V is the held-out validation set and $|E| = k$ denotes that the number of exemplars in E is k. Therefore, the exemplar sequence E found represents a fixed ordered set of exemplars that apply to every data point in the validation set. Note that our formulation considers exemplars jointly as *ordered* text in the sequence E. This space of permutation Ω is also a lot larger than the (unordered) combinatorial search space considered in the subset selection formulation of exemplar selections in [3, 9, 12, 38]. We show the superior performance of our method against subset selection methods in our experiments (Sec. 5).

4 Automated selection of exemplars

4.1 NeuralUCB for query-efficient optimization of exemplar sequences

We propose to use neural bandits to iteratively maximize the black-box objective function (1). At each iteration t, the neural bandits algorithm selects the next input query based on the belief of the objective given all past $t - 1$ observations $O_{t-1} \triangleq \{(E_i, s_V(E_i))\}_{i=1}^{t-1}$ where E_i and $s_V(E_i)$ are the exemplar sequence and corresponding validation score at iteration i, respectively. Here, the validation score is a realization of the objective function (1), so $s_V(E) = 1/|D_V| \sum_{(x,y) \in D_V} s(f(E, x), y)$.

Inspired by Lin et al. [14] who have shown impressive performances of the neural upper confidence bound (NeuralUCB) acquisition function [42] on instruction optimization, we adopt the NeuralUCB approach to our novel black-box exemplar selection formulation in (1). We propose to use a neural network (NN) to directly learn the mapping from a general-purpose hidden embedding of input exemplar sequences to the validation score. Specifically, we propose to use a embedding model $h(\cdot)$ and train a network $m(h(E); \theta_t)$ at iteration t such that

$$\theta_t = \arg\min_\theta \mathbb{E}_{(E, s_V(E)) \in O_{t-1}} \ell(m(h(E); \theta), s_V(E)) \tag{2}$$

where ℓ is the mean squared error (MSE) loss function. The embedding model can be pre-trained on a large corpus of text and hence provides a powerful latent representation of the input sentence. For example, pre-trained sentence-transformer models like MPNet [31] and black-box APIs like OpenAI text embedding are both commonly used for downstream clustering, semantic search and classification. Importantly, even for the same subset of exemplars, a different ordering leads to different embedding, hence $h(E)$ captures both the content and the ordering of the exemplars in E.

Then, the trained NN can be used to iteratively select the next exemplar sequence E_t to query:

$$E_t = \arg\max_{E \in \Omega} \text{NeuralUCB}_t(E),$$
$$\text{NeuralUCB}_t(E) \triangleq m(h(E); \theta_t) + \nu_t \sigma_{t-1}(h(E); \theta_t), \quad (3)$$

where $m(h(E); \theta_t)$ is the predicted score, $\sigma_{t-1}(h(E); \theta_t)$ is the NN's uncertainty about the score of $h(E)$ and ν_t is a hyperparameter that balances the two terms. Using NeuralUCB, our EASE balances the *exploration* of the space of exemplar sequences E and the *exploitation* of the predicted score from the NN in a principled way.

This application of NeuralUCB in our work differs from that of Lin et al. [14] in three main aspects. Firstly, the input to our NN is the embedding of *exemplar sequences* rather than the latent representation of the *instructions* from [14]. Secondly, we relax the requirement of a separate white-box LLM from [14] and only need black-box access to the embedding model $h(\cdot)$. Thirdly, our application to exemplar selection dramatically increases the search space from finite candidate instructions [14] to the space of exemplar sequences. Particularly, the explosion of the size of the search space poses significant difficulties to optimization, which we will address in the next section.

4.2 Reducing computational cost through optimal transport (OT)

Directly applying NeuralUCB to exemplar selection is challenging due to the enormous search space of all permutations of exemplars on which the acquisition values (3) have to be evaluated. For example, selecting 5 exemplars out of 100 requires 100^5 evaluations which is far from being practical. It is natural to search over a *domain space* $Q_t = \{E^{(j,t)}\}_{j=1}^q$ with a smaller number q of exemplar sequences in each iteration t. However, uniformly sampling such a reduced space Q_t from Ω could be sub-optimal, because a small q, which is required to make our algorithm computationally feasible, is highly likely to discard important exemplar sequences (we verify this in ablation experiments in Sec. 5.5 and App. C.3). To this end, we make a large q feasible by introducing a novel technique based on optimal transport (OT) to further select from Q_t a subset Q_t' of $q' < q$ *relevant* exemplar sequences. This technique can simultaneously (a) reduce the computational cost (since $q' \ll \Omega$) and (b) preserve our performance (since we can now search over a large domain space Q_t) by imposing an implicit preference towards exemplars in Q_t that are more relevant to the task.

Given probability measures μ_s and μ_v over space $\mathcal{Z}$, the OT distance between μ_s and μ_v is defined as

$$OT(\mu_s, \mu_v) = \min_{\pi \in \Pi(\mu_s, \mu_v)} \int_{\mathcal{Z}^2} c(z, z') d\pi(z, z') \quad (4)$$

where $\Pi(\mu_s, \mu_v) = \{\pi \in \mathcal{P}(\mathcal{Z} \times \mathcal{Z}) | \int_{\mathcal{Z}} \pi(z, z') dz = \mu_s, \int_{\mathcal{Z}} \pi(z, z') dz' = \mu_v\}$ is a collection of couplings between two the distribution μ_s and μ_v, and $c : \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}^+$ is some symmetric positive cost function. In our problem, we propose to use the space of embedding from $h(\cdot)$ as $\mathcal{Z}$ because the embedding captures the semantics of the exemplars with a fixed-dimensional vector. As cosine similarity is usually used to train embedding models such as MPNet [31] and sentence-BERT [27], we propose to use the following cost function $c(h(e), h(e')) = 1 - sim_{cos}(h(e), h(e'))$ where $sim_{cos}(h(e), h(e'))$ measures the cosine similarity between the embedding of two exemplars e and e'. Given a sampled subset $S = \{e_i\}_{i=1}^k$, we define a discrete measure $\mu_s = \frac{1}{k} \sum_{i=1}^k \delta(h(e_i))$ where δ is the Dirac function. Likewise, we define μ_v for the validation set D_V.

Intuitively, a smaller value of (4) indicates that the subset of exemplars is more similar to the validation set and is hence more *relevant* to the task. This allows us to select the relevant exemplar sequences from Q_t to form the smaller subset Q_t', on which the acquisition values (3) are evaluated. Consequently, we can increase the size q of Q_t by hundreds of folds while being computationally feasible since the most expensive computation (i.e., computing embeddings for all exemplar sequences in Q_t) is not needed. Note that the extra computation incurred by OT is minimal since the embedding of every data exemplar in D and D_V can be pre-computed and reused. Therefore, OT helps us

examine a large number of permutations among the space of all permutations without significantly increasing the computation, which helps mitigate the problem of the exploding search space.

4.3 Natural extension to jointly optimize instructions and exemplars

Our algorithm can further boost the performance of ICL for LLMs by jointly optimizing the exemplars and the instruction. A natural extension of our formulation in Sec. 3 allows the instruction, being another essential component of the LLM prompt, to be simultaneously optimized with exemplars. This ensures the optimal matching between instructions and exemplars to achieve a fully automated pipeline with superior performance. Specifically, our formulation naturally extends to $E = (p, e_1, e_2, \ldots, e_k)$ where $p \in P$ is an instruction from a candidate space/set P, i.e., $Q_t' \leftarrow P \times Q_t'$. Subsequently, p can be intuitively treated as another type of "exemplar" from a new space P and optimized in conjuction with the exemplars. In practice, the fixed set P of candidate instructions can be generated by the black-box model through techniques such as APE [43], PromptBreeder [6], etc. This extension is not only simple in its implementation but also proven to be effective in our experiments in Sec. 5.4.

4.4 Our EASE algorithm

In iteration t of EASE, we first use the historical observations of the exemplar sequence and score pairs $\{(E_i, s_V(E_i))\}_{i=1}^{t-1}$ to train the score prediction NN. Then, we perform sampling to obtain the domain space Q_t, which will be further refined to a set Q_t' of top-q' candidates based on OT distances. The space of exemplar sequence E can be optionally augmented with instructions $p \in P$ from a set P of instruction candidates. Subsequently, we find the optimal E that maximizes the NeuralUCB acquisition function. The selected exemplar sequence E is then evaluated against the black-box LLM $f(\cdot)$ using the validation dataset D_V, obtaining a new observed pair of $(E, s_V(E))$. This process is repeated until the query budget T is exhausted. An overview of the algorithm is presented in Alg. 1.

Algorithm 1: EASE

Require : Data exemplars set D, validation set D_V, length of exemplars k, total budget T, number of initial rounds T_{init}, sampling size q', black-box target model $f(\cdot)$, embedding model $h(\cdot)$, neural network $m(\cdot; \theta)$, *(optional)* instruction set P.

1 Initialize $\{(E_i, s_V(E_i))\}_{i=1}^{T_{\text{init}}}$ with T_{init} randomly sampled $\{E_i\}_{i=1}^{T_{\text{init}}}$ from D;
2 **for** $t = T_{init}$ **to** T **do**
3 $\quad$ Following (2), use observations $\{(E_i, s_V(E_i))\}_{i=1}^{t-1}$ to train the NN $m(\cdot; \theta_t)$ parameter θ_t;
4 $\quad$ Sample sequences Q_t and select top-q' sequences Q_t' with smallest OT distances;
5 $\quad$ *(Optional)* Augment each $E \in Q_t'$ with instructions $p \in P$, i.e., $Q_t' \leftarrow P \times Q_t'$;
6 $\quad$ Following (3), select the next query $E_t = \arg\max_{E \in Q_t'} \text{NeuralUCB}_t(E)$;
7 $\quad$ Evaluate E_t on the black-box model to obtain the validation score $s_V(E_t)$;
8 **return** $E^* = \arg\max_{E_t : t \in \{1, \ldots, T\}} s_V(E_t)$

---

5 Experiments

We compare EASE with the following comprehensive suite of baselines. They include *subset selection methods* with (a) a determinantal point process (**DPP**) metric adapted from [38], and (b) maximum mean discrepancy (**MMD**) [30], (c) optimal transport (**OT**) [33] metrics. We also adapt retrieval-based methods to our setting using a new retrieve-then-sample strategy based on the validation set (details in App. B.2 and Sec. 5.5), specifically with the classical (d) **Cosine** similarity and (e) **BM25** [28] retrievers. We also compare with existing exemplar selection baselines using (f) an active selection policy learned using reinforcement learning (**Active**) [39], and (g) an exemplar influence metric (**Inf**) [23]. Additionally, we propose two more new strong baselines: (h) **Evo** which mutates exemplars through evolutionary strategies and (i) **Best-of-N** which explores the exemplar space uniformly until the query budget is exhausted. More implementation details are found in App. B.2. The number of exemplars in the in-context prompt is set to $k = 5$. The black-box query budget is 165 evaluations following Lin et al. [14]. Since the effectiveness of the optimization is directly reflected by the value of the objective function in (1), we report the validation accuracy in the following experiments unless otherwise specified. The test accuracy tables are presented in App. C.10.

6

Table 1: Average accuracy $\pm$ standard error achieved by the best exemplar sequence discovered by different algorithms over 3 independent trials. For better distinguishability, we do not include easy tasks here (i.e., with 100% accuracy across baselines) and show full results in Tab. 5 of App. C.1.

	DPP	MMD	OT	Cosine	BM25	Active	Inf	Evo	Best-of-N	EASE
antonyms	70.0$_{\pm0.0}$	80.0$_{\pm0.0}$	81.7$_{\pm1.4}$	85.0$_{\pm0.0}$	85.0$_{\pm0.0}$	80.0$_{\pm0.0}$	86.7$_{\pm1.4}$	88.3$_{\pm1.4}$	**90.0$_{\pm0.0}$**	**90.0$_{\pm0.0}$**
auto_categorization	3.3$_{\pm1.4}$	8.3$_{\pm1.4}$	0.0$_{\pm0.0}$	25.0$_{\pm0.0}$	16.7$_{\pm1.4}$	10.0$_{\pm2.4}$	21.7$_{\pm1.4}$	21.7$_{\pm1.4}$	20.0$_{\pm0.0}$	**30.0$_{\pm0.0}$**
diff	0.0$_{\pm0.0}$	0.0$_{\pm0.0}$	0.0$_{\pm0.0}$	0.0$_{\pm0.0}$	0.0$_{\pm0.0}$	0.0$_{\pm0.0}$	**100.0$_{\pm0.0}$**	**100.0$_{\pm0.0}$**	**100.0$_{\pm0.0}$**	**100.0$_{\pm0.0}$**
larger_animal	70.0$_{\pm0.0}$	91.7$_{\pm1.4}$	**100.0$_{\pm0.0}$**	**100.0$_{\pm0.0}$**	**100.0$_{\pm0.0}$**	66.7$_{\pm1.4}$	**100.0$_{\pm0.0}$**	**100.0$_{\pm0.0}$**	**100.0$_{\pm0.0}$**	**100.0$_{\pm0.0}$**
negation	**95.0$_{\pm0.0}$**	**95.0$_{\pm0.0}$**	**95.0$_{\pm0.0}$**	**95.0$_{\pm0.0}$**	**95.0$_{\pm0.0}$**	**95.0$_{\pm0.0}$**	**95.0$_{\pm0.0}$**	**95.0$_{\pm0.0}$**	**95.0$_{\pm0.0}$**	**95.0$_{\pm0.0}$**
object_counting	55.0$_{\pm2.4}$	56.7$_{\pm1.4}$	48.3$_{\pm1.4}$	61.7$_{\pm1.4}$	66.7$_{\pm1.4}$	51.7$_{\pm1.4}$	63.3$_{\pm3.6}$	70.0$_{\pm0.0}$	70.0$_{\pm0.0}$	**73.3$_{\pm1.4}$**
orthography_starts_with	20.0$_{\pm2.4}$	35.0$_{\pm0.0}$	61.7$_{\pm1.4}$	**78.3$_{\pm1.4}$**	70.0$_{\pm0.0}$	43.3$_{\pm1.4}$	70.0$_{\pm2.4}$	75.0$_{\pm0.0}$	**78.3$_{\pm1.4}$**	**78.3$_{\pm1.4}$**
rhymes	60.0$_{\pm1.4}$	51.7$_{\pm1.4}$	0.0$_{\pm0.0}$	**100.0$_{\pm0.0}$**	80.0$_{\pm0.0}$	65.0$_{\pm8.2}$	70.0$_{\pm10.8}$	**100.0$_{\pm0.0}$**	**100.0$_{\pm0.0}$**	**100.0$_{\pm0.0}$**
second_word_letter	10.0$_{\pm2.4}$	30.0$_{\pm0.0}$	28.3$_{\pm1.4}$	**50.0$_{\pm0.0}$**	**50.0$_{\pm0.0}$**	26.7$_{\pm8.3}$	40.0$_{\pm0.0}$	46.7$_{\pm1.4}$	**50.0$_{\pm0.0}$**	**50.0$_{\pm0.0}$**
sentence_similarity	20.0$_{\pm0.0}$	21.7$_{\pm2.7}$	40.0$_{\pm2.4}$	46.7$_{\pm1.4}$	53.3$_{\pm1.4}$	5.0$_{\pm4.1}$	18.3$_{\pm5.4}$	45.0$_{\pm0.0}$	51.7$_{\pm1.4}$	**56.7$_{\pm1.4}$**
sentiment	85.0$_{\pm0.0}$	90.0$_{\pm0.0}$	85.0$_{\pm0.0}$	96.7$_{\pm1.4}$	**100.0$_{\pm0.0}$**	85.0$_{\pm4.1}$	91.7$_{\pm1.4}$	**100.0$_{\pm0.0}$**	**100.0$_{\pm0.0}$**	**100.0$_{\pm0.0}$**
sum	0.0$_{\pm0.0}$	0.0$_{\pm0.0}$	0.0$_{\pm0.0}$	0.0$_{\pm0.0}$	0.0$_{\pm0.0}$	0.0$_{\pm0.0}$	**100.0$_{\pm0.0}$**	**100.0$_{\pm0.0}$**	**100.0$_{\pm0.0}$**	**100.0$_{\pm0.0}$**
synonyms	10.0$_{\pm0.0}$	25.0$_{\pm0.0}$	20.0$_{\pm0.0}$	**35.0$_{\pm0.0}$**	30.0$_{\pm0.0}$	3.3$_{\pm1.4}$	26.7$_{\pm1.4}$	30.0$_{\pm0.0}$	30.0$_{\pm0.0}$	30.0$_{\pm0.0}$
taxonomy_animal	43.3$_{\pm3.6}$	40.0$_{\pm2.4}$	46.7$_{\pm1.4}$	85.0$_{\pm2.4}$	80.0$_{\pm0.0}$	45.0$_{\pm6.2}$	70.0$_{\pm4.1}$	80.0$_{\pm0.0}$	80.0$_{\pm0.0}$	**88.3$_{\pm2.7}$**
translation_en-de	**90.0$_{\pm0.0}$**	80.0$_{\pm0.0}$	80.0$_{\pm0.0}$	**90.0$_{\pm0.0}$**	85.0$_{\pm0.0}$	56.7$_{\pm13.0}$	**90.0$_{\pm0.0}$**	**90.0$_{\pm0.0}$**	**90.0$_{\pm0.0}$**	**90.0$_{\pm0.0}$**
translation_en-es	90.0$_{\pm0.0}$	**100.0$_{\pm0.0}$**	96.7$_{\pm1.4}$	**100.0$_{\pm0.0}$**	**100.0$_{\pm0.0}$**	96.7$_{\pm1.4}$	98.3$_{\pm1.4}$	**100.0$_{\pm0.0}$**	**100.0$_{\pm0.0}$**	**100.0$_{\pm0.0}$**
translation_en-fr	76.7$_{\pm1.4}$	76.7$_{\pm1.4}$	81.7$_{\pm1.4}$	85.0$_{\pm0.0}$	85.0$_{\pm0.0}$	81.7$_{\pm1.4}$	85.0$_{\pm0.0}$	86.7$_{\pm1.4}$	85.0$_{\pm0.0}$	**88.3$_{\pm1.4}$**
word_sorting	26.7$_{\pm1.4}$	88.3$_{\pm1.4}$	88.3$_{\pm1.4}$	90.0$_{\pm0.0}$	71.7$_{\pm1.4}$	80.0$_{\pm0.0}$	88.3$_{\pm1.4}$	**93.3$_{\pm1.4}$**	91.7$_{\pm1.4}$	91.7$_{\pm1.4}$
word_unscrambling	68.3$_{\pm1.4}$	56.7$_{\pm1.4}$	71.7$_{\pm1.4}$	75.0$_{\pm0.0}$	76.7$_{\pm1.4}$	63.3$_{\pm3.6}$	66.7$_{\pm1.4}$	75.0$_{\pm0.0}$	75.0$_{\pm0.0}$	**78.3$_{\pm2.7}$**
# best-performing tasks	2	2	2	8	5	1	5	9	11	**17**

5.1 Empirical study of performance gains on various benchmark tasks

To study the effectiveness of EASE, we conduct empirical comparisons with baseline methods using the Instruction Induction (II) benchmark tasks [4]. We test on tasks that contain more than 100 training examples for selection. The results are shown in Tab. 1. Our EASE achieves the best performance in 17 out of 19 language tasks. We observe that the subset selection methods such as DPP, MMD and OT are ineffective in practice, implying that choosing subsets alone without considering the order information is inadequate. While the retrieval-based methods Cosine and BM25 have demonstrated success in retrieving test-sample-based exemplars [29], it is not as effective in finding the best exemplars that generalize to the entire task. This is because the exemplars retrieved at the instance level may not work well for the entire validation set. The Active baseline requires training an active exemplar selection policy through the data-intensive reinforcement learning process, which explains its ineffectiveness under our budget-constrained setting. The Inf baseline is not efficient in subset sampling and disregards exemplar ordering, leading to worse results. Our proposed Evo and Best-of-N are the most competitive baselines with best performance in 9 and 11 tasks, respectively.

We discover that while EASE consistently outperforms or matches the baselines across tasks, for some tasks, most baselines achieve good performances. We hypothesize that *the effect of exemplar selection diminishes as the model gains more knowledge about the task*. This explains the instances where the choice of in-context exemplars has minimal impact on the performance, as the black-box model (i.e., GPT-3.5 Turbo) has been pre-trained on these publicly available language tasks. As suggested by Brown et al. [2], it is highly likely that GPT-3 has been trained on common benchmark datasets potentially contained in Common Crawl due to data contamination. Hence, further providing high-quality in-context exemplars could hardly improve the performance of the model on these well-trained tasks. Another possibility could be that the exemplars mostly serve as the context for adapting to the new formatting rules of the specific task, i.e., the LLM is not utilizing the semantic contents of the exemplars to learn the underlying input-output relations. This aligns with the discoveries of Min et al. [19] and provides a potential explanation for the surprising behavior of LLMs. Therefore, the II dataset might not be the most suitable one to test EASE. Next, we verify the above hypothesis that the effect of exemplar selection diminishes as the model gains more knowledge about the task in Sec. 5.2, and propose more suitable families of datasets for exemplar selection in Sec. 5.3.

5.2 Empirical validation of the hypothesis through progressive finetuning

We hypothesize that *the effect of exemplar selection diminishes as the model gains more knowledge about the task*. To gain insights on this hypothesis, we study an open-source white-source Vicuna model instead of the black-box GPT-3.5 model that is only accessible through API. We progressively finetune the while-box language model, and examine whether the extent of finetuning on a specific task diminishes the importance of in-context exemplars when prompted.

Our results are in Fig. 1. Compared to the most competitive baseline, Best-of-N, *the gain from exemplar selection using our EASE diminishes as the model is finetuned on the dataset of the*

respective tasks for more epochs. Across the three tasks shown in Fig. 1, using `EASE` originally has a performance gain of about 3%-10% and this gain slowly diminishes to 0 as finetuning progresses. This verifies our hypothesis and calls for further investigation of the exemplar selection performance of our `EASE` on tasks that have not been seen by the model, which we conduct in Sec. 5.3.
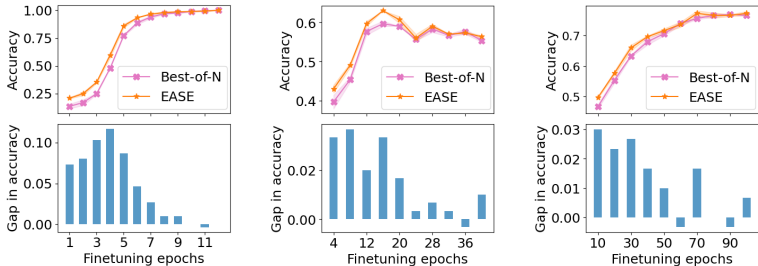

Figure 1: From left to right, the tasks are taxonomy animal, sentence similarity and object counting. The performance gaps between `EASE` and the Best-of-N baseline diminish as the LLM is finetuned.

5.3 New families of "out-of-distribution" tasks that emphasize in-context reasoning

We propose three new families of "out-of-distribution" tasks (i.e., loosely referred to as tasks on which the LLM is not already well trained) that highlight the importance of high-quality exemplars, which could also be of independent interest. The new tasks require the LLM to learn the underlying function/relationship in the input prompt in order to perform reasoning during inference, and are hence more sensitive to the quality of the exemplars.

Rule-based tasks. Given our insight on the impact of the model's existing knowledge about the task (Sec. 5.2), we propose rule-based tasks that contain novel rules that the LLM has not learned before. A key characteristic of these tasks is that the model has to extract the underlying relationships among the provided in-context exemplars and directly use the relationship for test-time inferences. For example, we construct the linear regression (**LR**) task where the input takes the form demonstrated in Example 2(see App. B.1). The underlying relationship in this example is $y = ax + b$, with $a = -4$ and $b = 6$ in this specific Example 2. Without further instructions, the model is supposed to rely on the provided in-context exemplars to implicitly infer the regression task, recover the coefficients a, b for linear regression, and then directly apply it to the test sample (e.g., for test input 117, compute $-4 \times 117 + 6 = -462$ and output 462). The results are in Tab. 2. For the clean dataset (i.e., 0% noise), `EASE` outperforms the most competitive baseline by 10% in absolute accuracy. Note that `EASE` has greater advantages in settings with noisy data, which will be discussed later in this section.

Another example of our proposed rule-based tasks is constructed by changing the rules for language puzzles (**LP**). A prominent example is "pig Latin" which follows two simple rules: (a) For words that begin with a vowel, one just adds "yay" to the end; (b) for words that begin with consonants, the initial consonants are moved to the end of the word, then "ay" is added. For example, translating "Hello, how are you today?" to pig Latin gives "Ellohay, owhay areyay ouyay odaytay?". Note that this task also requires the model to reason about the language translation rules (i.e., rules (a) and (b)) before predicting for the test sample. The rules can be freely modified (e.g., by changing the suffix word "yay" to others) to create diverse tasks that require in-context reasoning from the LLM. Hence, we create a new variant of the LP task, named **LP-variant**, by using the "ay" suffix for both rules (a) and (b). An example of the task query is given in Example 3 (see App. B.1). As shown in Tab. 2, in this task, our `EASE` also demonstrates competitive results and outperforms all baselines.

Remapped label tasks. By remapping the labels of existing classification tasks to new ones, we construct novel tasks that are against the model's existing knowledge. In the commonly used AG News dataset, the news articles are classified into four categories: "World", "Sports", "Business" and "Sci/Tech". We construct a remapped dataset **AG News Remap** such that "World" news is now labeled as "Sports" news, "Sports" is now labeled as "Business", etc. This is against the LLM's knowledge since the output now does not correspond to the context descriptions of the input (i.e., news articles). So, the LLM has to learn these remapping rules from the in-context exemplars. Similarly, we construct another dataset **SST5 Reversed** by reversing the labels of the sentiment analysis dataset SST5, such that "very negative" labels are swapped with "very positive" labels, and "negative" labels

Table 2: Average accuracy $\pm$ standard error over 3 independent trials achieved by different algorithms on the new families of out-of-distribution tasks.

Type	Task	Noise	DPP	MMD	OT	Cosine	BM25	Active	Inf	Evo	Best-of-N	EASE
Rule-based tasks	LR	0%	$31.7_{\pm1.4}$	$38.3_{\pm2.7}$	$50.0_{\pm0.0}$	$71.7_{\pm1.4}$	$70.0_{\pm0.0}$	$36.7_{\pm1.4}$	$56.7_{\pm5.9}$	$61.7_{\pm1.4}$	$66.7_{\pm1.4}$	$\mathbf{81.7_{\pm3.6}}$
		10%	$8.3_{\pm1.4}$	$36.7_{\pm1.4}$	$48.3_{\pm1.4}$	$61.7_{\pm1.4}$	$61.7_{\pm1.4}$	$0.0_{\pm0.0}$	$58.3_{\pm3.6}$	$60.0_{\pm0.0}$	$65.0_{\pm2.4}$	$\mathbf{73.3_{\pm3.6}}$
		30%	$10.0_{\pm0.0}$	$28.3_{\pm1.4}$	$46.7_{\pm1.4}$	$63.3_{\pm1.4}$	$60.0_{\pm0.0}$	$40.0_{\pm2.4}$	$35.0_{\pm2.4}$	$53.3_{\pm1.4}$	$50.0_{\pm0.0}$	$\mathbf{78.3_{\pm1.4}}$
		50%	$0.0_{\pm0.0}$	$38.3_{\pm1.4}$	$45.0_{\pm0.0}$	$65.0_{\pm0.0}$	$53.3_{\pm1.4}$	$0.0_{\pm0.0}$	$53.3_{\pm1.4}$	$46.7_{\pm1.4}$	$45.0_{\pm0.0}$	$\mathbf{71.7_{\pm2.7}}$
		70%	$0.0_{\pm0.0}$	$55.0_{\pm0.0}$	$38.3_{\pm2.7}$	$65.0_{\pm0.0}$	$50.0_{\pm0.0}$	$26.7_{\pm5.4}$	$30.0_{\pm4.7}$	$33.3_{\pm1.4}$	$33.3_{\pm1.4}$	$\mathbf{66.7_{\pm3.6}}$
		90%	$0.0_{\pm0.0}$	$21.7_{\pm1.4}$	$26.7_{\pm1.4}$	$46.7_{\pm1.4}$	$3.3_{\pm1.4}$	$0.0_{\pm0.0}$	$6.7_{\pm2.7}$	$8.3_{\pm1.4}$	$15.0_{\pm0.0}$	$\mathbf{53.3_{\pm2.7}}$
	LP-variant	0%	$48.3_{\pm2.7}$	$40.0_{\pm2.4}$	$41.7_{\pm1.4}$	$65.0_{\pm0.0}$	$58.3_{\pm1.4}$	$30.0_{\pm0.0}$	$61.7_{\pm1.4}$	$75.0_{\pm2.4}$	$71.7_{\pm1.4}$	$\mathbf{75.0_{\pm0.0}}$
		10%	$0.0_{\pm0.0}$	$36.7_{\pm1.4}$	$40.0_{\pm0.0}$	$63.3_{\pm2.7}$	$60.0_{\pm0.0}$	$36.7_{\pm2.7}$	$65.0_{\pm2.4}$	$70.0_{\pm2.4}$	$73.3_{\pm1.4}$	$\mathbf{75.0_{\pm2.4}}$
		30%	$0.0_{\pm0.0}$	$48.3_{\pm2.7}$	$40.0_{\pm2.4}$	$60.0_{\pm0.0}$	$55.0_{\pm0.0}$	$40.0_{\pm7.1}$	$53.3_{\pm4.9}$	$65.0_{\pm2.4}$	$65.0_{\pm0.0}$	$\mathbf{73.3_{\pm1.4}}$
		50%	$0.0_{\pm0.0}$	$65.0_{\pm0.0}$	$35.0_{\pm2.4}$	$63.3_{\pm2.7}$	$60.0_{\pm0.0}$	$38.3_{\pm3.6}$	$48.3_{\pm3.6}$	$61.7_{\pm1.4}$	$65.0_{\pm0.0}$	$\mathbf{76.7_{\pm2.7}}$
		70%	$0.0_{\pm0.0}$	$46.7_{\pm2.7}$	$35.0_{\pm0.0}$	$70.0_{\pm0.0}$	$60.0_{\pm0.0}$	$25.0_{\pm8.2}$	$60.0_{\pm4.1}$	$56.7_{\pm1.4}$	$56.7_{\pm1.4}$	$\mathbf{75.0_{\pm0.0}}$
		90%	$0.0_{\pm0.0}$	$35.0_{\pm2.4}$	$50.0_{\pm0.0}$	$65.0_{\pm2.4}$	$0.0_{\pm0.0}$	$30.0_{\pm12.5}$	$50.0_{\pm2.4}$	$38.3_{\pm1.4}$	$55.0_{\pm2.4}$	$63.3_{\pm1.4}$
Re-mapped label tasks	AG News Remap	0%	$20.0_{\pm2.4}$	$15.0_{\pm0.0}$	$26.7_{\pm1.4}$	$43.3_{\pm1.4}$	$43.3_{\pm2.7}$	$5.0_{\pm2.4}$	$25.0_{\pm4.1}$	$40.0_{\pm0.0}$	$40.0_{\pm0.0}$	$\mathbf{53.3_{\pm3.6}}$
		10%	$5.0_{\pm0.0}$	$15.0_{\pm0.0}$	$15.0_{\pm0.0}$	$41.7_{\pm1.4}$	$38.3_{\pm1.4}$	$3.3_{\pm1.4}$	$26.7_{\pm2.7}$	$36.7_{\pm1.4}$	$40.0_{\pm0.0}$	$\mathbf{56.7_{\pm2.7}}$
		30%	$10.0_{\pm0.0}$	$5.0_{\pm0.0}$	$5.0_{\pm0.0}$	$40.0_{\pm0.0}$	$36.7_{\pm1.4}$	$1.7_{\pm1.4}$	$10.0_{\pm0.0}$	$40.0_{\pm0.0}$	$43.3_{\pm1.4}$	$\mathbf{51.7_{\pm1.4}}$
		50%	$5.0_{\pm0.0}$	$10.0_{\pm0.0}$	$5.0_{\pm0.0}$	$43.3_{\pm1.4}$	$35.0_{\pm0.0}$	$3.3_{\pm1.4}$	$20.0_{\pm4.1}$	$35.0_{\pm0.0}$	$35.0_{\pm0.0}$	$\mathbf{56.7_{\pm1.4}}$
		70%	$5.0_{\pm0.0}$	$25.0_{\pm0.0}$	$8.3_{\pm1.4}$	$50.0_{\pm0.0}$	$35.0_{\pm0.0}$	$1.7_{\pm1.4}$	$11.7_{\pm5.4}$	$38.3_{\pm1.4}$	$46.7_{\pm1.4}$	$\mathbf{51.7_{\pm1.4}}$
		90%	$5.0_{\pm0.0}$	$18.3_{\pm1.4}$	$5.0_{\pm0.0}$	$40.0_{\pm0.0}$	$10.0_{\pm0.0}$	$15.0_{\pm6.2}$	$35.0_{\pm0.0}$	$35.0_{\pm0.0}$	$41.7_{\pm1.4}$	$\mathbf{55.0_{\pm2.4}}$
	SST5 Reverse	0%	$20.0_{\pm0.0}$	$10.0_{\pm0.0}$	$13.3_{\pm1.4}$	$40.0_{\pm0.0}$	$40.0_{\pm0.0}$	$15.0_{\pm2.4}$	$33.3_{\pm5.4}$	$35.0_{\pm2.4}$	$40.0_{\pm0.0}$	$\mathbf{50.0_{\pm0.0}}$
		10%	$16.7_{\pm1.4}$	$10.0_{\pm0.0}$	$15.0_{\pm0.0}$	$48.3_{\pm1.4}$	$40.0_{\pm0.0}$	$13.3_{\pm2.7}$	$23.3_{\pm5.4}$	$33.3_{\pm2.7}$	$40.0_{\pm0.0}$	$\mathbf{50.0_{\pm0.0}}$
		30%	$23.3_{\pm1.4}$	$6.7_{\pm1.4}$	$25.0_{\pm2.4}$	$40.0_{\pm0.0}$	$40.0_{\pm0.0}$	$21.7_{\pm3.6}$	$26.7_{\pm1.4}$	$30.0_{\pm0.0}$	$31.7_{\pm1.4}$	$\mathbf{41.7_{\pm3.6}}$
		50%	$21.7_{\pm1.4}$	$15.0_{\pm0.0}$	$15.0_{\pm0.0}$	$43.3_{\pm1.4}$	$33.3_{\pm1.4}$	$21.7_{\pm1.4}$	$23.3_{\pm1.4}$	$28.3_{\pm1.4}$	$30.0_{\pm0.0}$	$\mathbf{43.3_{\pm1.4}}$
		70%	$25.0_{\pm0.0}$	$23.3_{\pm1.4}$	$23.3_{\pm1.4}$	$40.0_{\pm0.0}$	$30.0_{\pm0.0}$	$20.0_{\pm2.4}$	$25.0_{\pm2.4}$	$36.7_{\pm1.4}$	$36.7_{\pm1.4}$	$\mathbf{45.0_{\pm2.4}}$
		90%	$20.0_{\pm0.0}$	$15.0_{\pm2.4}$	$20.0_{\pm0.0}$	$30.0_{\pm0.0}$	$30.0_{\pm0.0}$	$13.3_{\pm2.7}$	$21.7_{\pm1.4}$	$30.0_{\pm0.0}$	$30.0_{\pm0.0}$	$\mathbf{31.7_{\pm1.4}}$

are swapped with "positive" labels. The results for these novel remapped label tasks are also in Tab. 2, which shows superior performances of EASE over baselines by about 10% in absolute accuracy.

Noisy tasks. Exemplar selection is even more important to achieve good ICL performances when the dataset is potentially noisy, since using noisy or mislabeled data as exemplars could have detrimental effects on performance. Noisy datasets also better resemble practical scenarios because clean data is expensive and difficult to obtain. We construct noisy datasets by injecting various ratios of noisy outputs into the task datasets, ranging from having 10% noisy data samples (i.e., the remaining 90% are clean) to having 90% noisy data samples (i.e., the remaining 10% are clean). We show in Tab. 2 that EASE is the best-performing method across different noise ratios and generally exhibits a lower decrease in accuracy as the tasks become more difficult with increasing noise ratios.

Note that the conclusions on test accuracy results (in App. C.10) are consistent with the main text. These tasks above represent new families of tasks that contain novel knowledge for LLMs. We show through comprehensive experiments that exemplar selection is important and useful in practice, especially for novel downstream task adaptations. Also, the noisy datasets considered here align with real-world scenarios, which typically contain noises from observation, labeling error, corruption, etc.

5.4 Jointly optimizing instruction and exemplars

To the best of our knowledge, no prior work can jointly optimize instruction and exemplars. For a fair comparison, we do not include an instruction when comparing with other exemplar selection baselines in earlier sections. As EASE is readily extended to find the optimal combination of instruction and exemplars (Sec. 4.3), we show the benefits of joint optimization in this section. According to Tab. 3, jointly optimizing these two essential components of a prompt significantly improves the performance for some difficult tasks (marked with red arrows ↑). This improvement is attributed to the ability of the optimized instruction to significantly reinforce the information captured in the corresponding exemplars. For example, an automatically optimized instruction "identify and list the animals from the given words" complements the exemplars in the taxonomy animal task. Therefore, our joint optimization of instruction and exemplars presents a *fully automated pipeline* for prompt optimization and achieves impressive practical performances.

5.5 Ablation studies

Combination with retrieval-based methods to handle larger sets of exemplars. Applying EASE to scenarios where the size n of the set of exemplars D is very large may lead to performance degradation. This is because the space Ω of exemplar sequences becomes excessively large, which cannot be

Table 3: Average accuracy $\pm$ s.e. for EASE with and without jointly optimized instructions. We removed tasks with 100% accuracy. The full results are in App C, Tab. 6.

	EASE	EASE with instructions	Improve-ment
antonyms	$90.0_{\pm0.0}$	$85.0_{\pm0.0}$	-5.0 ↓
auto_categorization	$30.0_{\pm0.0}$	$46.7_{\pm4.9}$	16.7 ↑
negation	$95.0_{\pm0.0}$	$100.0_{\pm0.0}$	5.0 ↑
object_counting	$73.3_{\pm1.4}$	$75.0_{\pm0.0}$	1.7 ↑
orthography_starts_with	$78.3_{\pm1.4}$	$81.7_{\pm1.4}$	3.3 ↑
rhymes	$100.0_{\pm0.0}$	$91.7_{\pm3.6}$	-8.3 ↓
second_word_letter	$50.0_{\pm0.0}$	$100.0_{\pm0.0}$	50.0 ↑
sentence_similarity	$56.7_{\pm1.4}$	$56.7_{\pm1.4}$	0.0 ○
synonyms	$30.0_{\pm0.0}$	$30.0_{\pm0.0}$	0.0 ○
taxonomy_animal	$88.3_{\pm2.7}$	$100.0_{\pm0.0}$	11.7 ↑
translation_en-de	$90.0_{\pm0.0}$	$90.0_{\pm0.0}$	0.0 ○
translation_en-es	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	0.0 ○
translation_en-fr	$88.3_{\pm1.4}$	$85.0_{\pm0.0}$	-3.3 ↓
word_sorting	$91.7_{\pm1.4}$	$91.7_{\pm1.4}$	0.0 ○
word_unscrambling	$78.3_{\pm2.7}$	$80.0_{\pm0.0}$	1.7 ↑
linear_4_10_noisy	$73.3_{\pm3.6}$	$41.7_{\pm9.5}$	-31.7 ↓
LP-variant (10% noise)	$75.0_{\pm2.4}$	$85.0_{\pm2.4}$	10.0 ↑
AG News Remap (10% noise)	$56.7_{\pm2.7}$	$65.0_{\pm0.0}$	8.3 ↑
SST5 Reverse (10% noise)	$50.0_{\pm0.0}$	$50.0_{\pm0.0}$	0.0 ○

Table 4: Average accuracy $\pm$ s.e. achieved by EASE and EASE with retrieval for larger exemplar set sizes.

AG News Remap (10% noise)

Size n	EASE	EASE with retrieval
1000	$41.7_{\pm1.4}$	$63.3_{\pm1.4}$
10000	$55.0_{\pm2.4}$	$65.0_{\pm0.0}$
50000	$56.7_{\pm3.6}$	$63.3_{\pm1.4}$
100000	$50.0_{\pm2.4}$	$65.0_{\pm0.0}$

SST5 Reverse (10% noise)

Size n	EASE	EASE with retrieval
1000	$46.7_{\pm1.4}$	$55.0_{\pm3.5}$
3000	$42.5_{\pm1.8}$	$51.7_{\pm1.4}$
5000	$43.3_{\pm1.4}$	$45.0_{\pm0.0}$
7000	$43.3_{\pm1.4}$	$50.0_{\pm0.0}$

sufficiently explored without significantly increasing the computation (i.e., with the size of Q_t being fixed). To resolve this issue, a natural idea is to first filter the large pool of data exemplars to eliminate those that are less relevant to the task. To this end, we propose to first use retrieval-based methods to select the exemplars that are more relevant to the task, and then run our EASE using this refined smaller set of exemplars. Specifically, we use a cosine similarity retriever and perform exemplar selection on D with a size n as large as 100000. As shown in Tab. 4, when the size n of the exemplar set is large, combining EASE with retrieval gives better performances than directly running EASE.

Effectiveness of components of EASE. Here we show the necessity of both of the main components to the success of EASE: OT and NeuralUCB. As shown in Tab. 7 of App. C.3, EASE significantly outperforms methods employing OT or NeuralUCB alone. Overall, a pure subset selection algorithm based on OT performs badly on its own, especially when the dataset's noise ratio is high. However, when used together with NeuralUCB, OT significantly improves the performance of our EASE.

Further ablations. We defer further ablation studies to App. C, including those exploring (a) speedups from OT, (b) a larger k, (c) benefits of using an ordering-aware embedding, (d) different black-box LLMs, (e) other embedding models, and (f) asking GPT to directly select exemplars.

6 Conclusion and limitation

We propose EASE, an algorithm that selects the optimal ordered set of exemplars for in-context learning of black-box LLMs in an automated fashion. EASE is query-efficient due to the adoption of the NeuralUCB algorithm and is further made computationally feasible for large spaces of exemplar sequences through a technique based on optimal transport. Additionally, our EASE has been extended to a fully automated prompt optimization pipeline that jointly optimizes exemplars and instruction for the best in-context learning performance. Furthermore, we provide practical insights indicating that exemplar selection in in-context learning is more crucial for downstream tasks that the LLM has limited knowledge about. However, the on-the-fly computation of embedding for the ordered exemplar sequences is the computational bottleneck of our method, which could be potentially improved in future work for more efficient optimization. Also, a potential limitation of EASE is the requirement for a suitable validation set, which may not be readily available in some scenarios.

References

[1] Alon Albalak, Yanai Elazar, Sang Michael Xie, Shayne Longpre, Nathan Lambert, Xinyi Wang, Niklas Muennighoff, Bairu Hou, Liangming Pan, Haewon Jeong, Colin Raffel, Shiyu Chang, Tatsunori Hashimoto, and William Yang Wang. A survey on data selection for language models. *arXiv:2402.16827*, 2024.

[2] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Proc. NeurIPS*, pages 1877–1901, 2020.

[3] Ting-Yun Chang and Robin Jia. Data curation alone can stabilize in-context learning. In *Proc. ACL*, pages 8123–8144, 2023.

[4] Lichang Chen, Jiuhai Chen, Tom Goldstein, Heng Huang, and Tianyi Zhou. InstructZero: Efficient instruction optimization for black-box large language models. *arXiv:2306.03082*, 2023.

[5] Lingjiao Chen, Matei Zaharia, and James Zou. How is ChatGPT's behavior changing over time? *arXiv:2307.09009*, 2023.

[6] Chrisantha Fernando, Dylan Banarse, Henryk Michalewski, Simon Osindero, and Tim Rocktäschel. Promptbreeder: Self-referential self-improvement via prompt evolution. *arXiv:2309.16797*, 2023.

[7] Lingyu Gao, Aditi Chaudhary, Krishna Srinivasan, Kazuma Hashimoto, Karthik Raman, and Michael Bendersky. Ambiguity-aware in-context learning with large language models. *arXiv:2309.07900*, 2024.

[8] Qingyan Guo, Rui Wang, Junliang Guo, Bei Li, Kaitao Song, Xu Tan, Guoqing Liu, Jiang Bian, and Yujiu Yang. Connecting large language models with evolutionary algorithms yields powerful prompt optimizers. In *Proc. ICLR*, 2024.

[9] Shivanshu Gupta, Matt Gardner, and Sameer Singh. Coverage-based example selection for in-context learning. In *Proc. EMNLP*, pages 13924–13950, 2023.

[10] Wenyang Hu, Yao Shu, Zongmin Yu, Zhaoxuan Wu, Xiangqiang Lin, Zhongxiang Dai, See-Kiong Ng, and Bryan Kian Hsiang Low. Localized zeroth-order prompt optimization. *arXiv:2403.02993*, 2024.

[11] Itay Levy, Ben Bogin, and Jonathan Berant. Diverse demonstrations improve in-context compositional generalization. In *Proc. ACL*, pages 1401–1422, 2023.

[12] Xiaonan Li and Xipeng Qiu. Finding support examples for in-context learning. In *Proc. EMNLP*, pages 6219–6235, 2023.

[13] Xiaonan Li, Kai Lv, Hang Yan, Tianyang Lin, Wei Zhu, Yuan Ni, Guotong Xie, Xiaoling Wang, and Xipeng Qiu. Unified demonstration retriever for in-context learning. In *Proc. ACL*, pages 4644–4668, 2023.

[14] Xiaoqiang Lin, Zhaoxuan Wu, Zhongxiang Dai, Wenyang Hu, Yao Shu, See-Kiong Ng, Patrick Jaillet, and Bryan Kian Hsiang Low. Use your INSTINCT: Instruction optimization using neural bandits coupled with transformers. In *NeurIPS Workshop on Instruction Tuning and Instruction Following*, 2023.

[15] Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. What makes good in-context examples for GPT-3? In *Proc. DeeLIO: Deep Learning Inside Out*, pages 100–114, 2022.

[16] Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proc. ACL*, pages 8086–8098, 2022.

[17] Man Luo, Xin Xu, Yue Liu, Panupong Pasupat, and Mehran Kazemi. In-context learning with retrieved demonstrations for language models: A survey. *arXiv:2401.11624*, 2024.

[18] Nestor Maslej, Loredana Fattorini, Raymond Perrault, Vanessa Parli, Anka Reuel, Erik Brynjolfsson, John Etchemendy, Katrina Ligett, Terah Lyons, James Manyika, Juan Carlos Niebles, Yoav Shoham, Russell Wald, and Jack Clark. The AI index 2024 annual report. Technical report, AI Index Steering Committee, Institute for Human-Centered AI, Stanford University, 2024.

[19] Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. Rethinking the role of demonstrations: What makes in-context learning work? In *Proc. EMNLP*, pages 11048–11064, 2022.

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

[21] Marius Mosbach, Tiago Pimentel, Shauli Ravfogel, Dietrich Klakow, and Yanai Elazar. Few-shot fine-tuning vs. in-context learning: A fair comparison and evaluation. In *Proc. ACL Findings*, pages 12284–12314, 2023.

[22] Seth Neel and Peter Chang. Privacy issues in large language models: A survey. *arXiv:2312.06717*, 2024.

[23] Tai Nguyen and Eric Wong. In-context example selection with influences. *arXiv:2302.11042*, 2023.

[24] OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, and Sam Altman et al. GPT-4 technical report. *arXiv:2303.08774*, 2024.

[25] Ethan Perez, Douwe Kiela, and Kyunghyun Cho. True few-shot learning with language models. In *Proc. NeurIPS*, pages 11054–11070, 2021.

[26] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *Proc. ICML*, pages 8748–8763, 2021.

[27] Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using siamese bert-networks. In *Proc. EMNLP*, pages 3982–3992, 2019.

[28] Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: BM25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389, 2009.

[29] Ohad Rubin, Jonathan Herzig, and Jonathan Berant. Learning to retrieve prompts for in-context learning. In *Proc. NAACL*, pages 2655–2671, 2022.

[30] Dino Sejdinovic, Bharath Sriperumbudur, Arthur Gretton, and Kenji Fukumizu. Equivalence of distance-based and RKHS-based statistics in hypothesis testing. *Annals of Statistics*, 41(5): 2263–2291, 2013.

[31] Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. MPNet: masked and permuted pre-training for language understanding. In *Proc. NeurIPS*, pages 16857–16867, 2020.

[32] Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, Anja Hauth, and Katie Millican et al. Gemini: A family of highly capable multimodal models. *arXiv:2312.11805*, 2024.

[33] Cédric Villani. *Topics in optimal transportation*, volume 58. American Mathematical Soc., 2021.

[34] Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Proc. NeurIPS*, pages 5776–5788, 2020.

[35] Xinyi Wang, Wanrong Zhu, Michael Saxon, Mark Steyvers, and William Yang Wang. Large language models are latent variable models: Explaining and finding good demonstrations for in-context learning. In *Proc. NeurIPS*, pages 15614–15638, 2023.

[36] Jeremy White. How strangers got my email address from ChatGPT's model. *The New York Times*, 2023. URL https://www.nytimes.com/interactive/2023/12/22/technology/openai-chatgpt-privacy-exploit.html.

[37] Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers. In *Proc. ICLR*, 2024.

[38] Jiacheng Ye, Zhiyong Wu, Jiangtao Feng, Tao Yu, and Lingpeng Kong. Compositional exemplars for in-context learning. In *Proc. ICML*, pages 39818–39833, 2023.

[39] Yiming Zhang, Shi Feng, and Chenhao Tan. Active example selection for in-context learning. In *Proc. EMNLP*, pages 9134–9148, 2022.

[40] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. A survey of large language models. *arXiv:2303.18223*, 2023.

[41] Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. Calibrate before use: Improving few-shot performance of language models. In *Proc. ICML*, pages 12697–12706, 2021.

[42] Dongruo Zhou, Lihong Li, and Quanquan Gu. Neural contextual bandits with UCB-based exploration. In *Proc. ICML*, pages 11492–11502, 2020.

[43] Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. Large language models are human-level prompt engineers. In *Proc. ICLR*, 2023.

A Broader impacts

As LLMs continue to gain popularity and are increasingly used by a broad user base (e.g., evidenced by OpenAI's over 1 million active users), carefully handling the ethical considerations associated with LLMs' diverse applications becomes crucial. Such a work like ours which focuses on automatically optimizing the performance of LLMs offers valuable applications and ease of usage. However, it is both challenging and essential to address the potential for malicious usage. When applied to tasks specially designed with malicious or adversarial intent, our method could be exploited to produce harmful instructions and exemplars. In such cases, responsible usage of the tool is important. There may be a need for a user-friendly platform (which integrates safety measures, instead of providing the raw code) to add an extra layer of protection. Additionally, we urge the community to focus more on potential ethical and safety issues related to the usage of LLMs and associated technologies, which should also account for additional objectives and constraints (e.g., harmfulness).

B Implementation details

In this section, we provide details for the implementation of the data processing (see App. B.1) and the baseline methods employed in this paper (see App. B.2). All experiments are conducted on a server with Intel(R) Xeon(R) CPU and NVIDIA H100 GPUs. Unless otherwise stated, we use "gpt-3.5-turbo-1106" as the target black-box model, and MPNet as the embedding model.

B.1 Details for data processing

We follow the query template proposed by APE [43] for querying LLM models. Example 1 below shows the general query template for in-context learning (ICL) used in the paper, and the LLM will generate outputs corresponding to the query. In the template, the placeholders (i.e., [INSTRUCTION], [INPUT], [OUTPUT] and [TEST INPUT]) are replaced by raw text in the datasets. The placeholder "<More exemplars...>" represents any additional exemplars written the same input-output format depending on the number of in-context exemplars k in the prompt. The "instruction" is optional in the query and we only include the instruction when jointly optimizing for both the exemplars and the instruction (Sec. 4.3 and Sec. 5.4).

Example Query 1: A General Template

(optional) Instruction: [INSTRUCTION]

Input: [INPUT]
Output: [OUTPUT]

<More exemplars...>

Input: [INPUT]
Output: [OUTPUT]

Input: [TEST INPUT]
Output:

Next, we present the details for the new families of out-of-distribution tasks that we proposed in Sec. 5.3.

Linear regression (LR). The LR task is generated with an underlying linear regression function $y = ax + b$, where a, b are the coefficients to be defined. In our specific example of the LR task presented in the paper, we arbitrarily choose $a = -4$ and $b = 6$. We let x be the input and y be the output. Note that to make the task difficult, no information about the function structure (i.e., $y = ax + b$) is passed to the LLM in the query. A concrete example is shown in Example 2.

Language puzzle variant (LP-variant). We change the rules for the classic language game "pig Latin". The original pig Latin follows 2 rules: (a) For words that begin with a vowel, one just adds "yay" to the end; (b) For words that begin with consonants, the initial consonants are moved to the end

of the word, then "ay" is added. We create a variant, LP-variant, using the "ay" suffix for both rules (a) and (b). An example of the task query is given in Example 3. Note that one could freely modify the rules, by changing the suffix word "yay" to something else for example, and create diverse tasks that require in-context reasoning from the model.

Example Query 2: LR	**Example Query 3: LP-variant**
Input: 172 Output: -682 <More exemplars...> Input: 47 Output: -182 Input: 117 Output:	Input: quick brown fox The jumps Over the Lazy dog Output: uickqay ownbray oxfay Ethay umpsjay Overyay ethay Azylay ogday <More exemplars...> Input: Tom never walks to school Output: Omtay evernay alksway otay oolschay Input: We never talk Output:

AG News Remap. For the commonly-used AG News dataset, the news articles are classified into four categories: "World", "Sports", "Business" and "Sci/Tech". We constructed a re-mapped dataset by "shifting" the label mapping, such that

- "World" news is now labeled as "Sports" news,
- "Sports" news is now labeled as "Business" news,
- "Business" news is now labeled as "Sci/Tech" news,
- "Sci/Tech" news is now labeled as "World" news.

Note that the remapping rule above is chosen arbitrarily. Therefore, one could create a variety of tasks by changing the remapping rule, or even directly changing the labels completely. For the above rule that we adopted, an example of the AG News Remap task query is given in Example 4.

Example Query 4: AG News Remap
Input: Yahoo, Adobe join hands Adobe Systems, the digital imaging, design and document technology platform provider and Internet service provider Yahoo will announce this week the launch of a co-branded Yahoo Toolbar. Output: World <More exemplars...> Input: Schumacher Sets Mark Michael Schumacher won the Hungarian Grand Prix Sunday in Budapest, setting yet another record by becoming the first Formula One driver with 12 victories in a season. Output: Business Input: LeapFrog Warns on 3Q, Year Profit View LeapFrog Enterprises Inc., a developer of technology-based educational products, on Monday lowered third-quarter and full-year profit expectations, citing difficult market conditions. Output:

SST5 Reverse. SST5 is another commonly used 5-way sentiment classification dataset, with labels being "very positive", "positive", "neutral", "negative" and "very negative". To make the task novel and unseen by the LLM before, we reverse the labels to be against the sentiment expressed in the input, such that

- "very positive" sentences are now labeled as "very negative",
- "positive" sentences are now labeled as "negative",

- "neutral" sentences are still labeled as "neutral",
- "negative" sentences are now labeled as "positive",
- "very negative" sentences are now labeled as "very positive".

This makes the task difficult as the LLM has to now output sentiments that are against the pre-trained knowledge about sentiments, which is obtained from the large corpus of pre-training data that the LLM has been trained on.

An example of the SST5 Reverse task query is given in Example 5.

Example Query 5: SST5 Reverse

Input: extremely bad .
Output: very positive

<More exemplars...>

Input: ... bright , intelligent , and humanly funny film .
Output: very negative

Input: really dumb but occasionally really funny .
Output:

Injecting noises into datasets. To construct noisy datasets with different noise ratios, we mainly modify the labels of data points. Specifically, to construct a dataset with $r\%$ noisy data, we sample $r\%$ data and replace their labels with the labels of other randomly sampled data points in the dataset. We also design specific noise structures for LR and LP-variant to simulate noises due to a systematic error. For LR, the noisy data instead follows $y = 5x - 8$. For LP-variant, the noisy data simply repeats the input sentence in the label.

B.2 Details for implementation of the methods and baselines

For all methods, we use a consistent exemplar selection set D, validation set D_V and test set for evaluating one task. We also implement all exemplar selection without replacement for all methods.

DDP. We follow Ye et al. [38] to use the DPP metric combined with relevance scores as the subset selection criterion. We do not train the embedding model and directly use cosine similarity to measure the relevance term. However, the original metric by Ye et al. [38] is dependent on the test input x_{test}, so we adapted the method to average the metric over the validation dataset. The exemplar set with the maximum values on the metric is chosen and evaluated on the black-box LLM.

MMD. We use the MMD metric in Sejdinovic et al. [30] to measure the distance between the exemplars set and the validation set. The exemplar set with the maximum values on the MMD metric is chosen and evaluated on the black-box LLM.

OT. We use the OT metric in (4) to measure the distance between the exemplars set and the validation set. Similarly, the exemplar set with the maximum values on the OT metric is chosen and evaluated on the black-box LLM.

Cosine. Retrieval-based methods are not suitable for finding a fixed set of exemplars for the entire test set in its original form. Hence, we propose the following adaptation. We first retrieve top-R candidates with the lowest distance to all validation samples on average. Then, we sample T permutations of exemplars from the R candidates to test on the black-box LLM, where T is the black-box query budget. In the above, the retriever calculates the cosine similarity between the embeddings of samples obtained from an embedding model $h(\cdot)$. In this paper, we use $R = 10$ and use MPNet as $h(\cdot)$.

BM25. This retrieval-based baseline works similarly to Cosine. The only difference is that this baseline uses the classic BM25 [28] as the retriever.

Active. We use the official implementation of Zhang et al. [39]. For a fair comparison using the same query budget T for the black-box LLM, we limit the number of episodes that can be used according to T, and then train the exemplar selection policy through reinforcement learning.

Table 5: Average validation accuracy $\pm$ standard error achieved by the best exemplar sequence discovered by different algorithms for different tasks over 3 independent trials.

	DPP	MMD	OT	Cosine	BM25	Active	Inf	Evo	Best-of-N	EASE
active_to_passive	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$
antonyms	$70.0_{\pm0.0}$	$80.0_{\pm0.0}$	$81.7_{\pm1.4}$	$85.0_{\pm0.0}$	$85.0_{\pm0.0}$	$80.0_{\pm0.0}$	$86.7_{\pm1.4}$	$88.3_{\pm1.4}$	$90.0_{\pm0.0}$	$90.0_{\pm0.0}$
auto_categorization	$3.3_{\pm1.4}$	$8.3_{\pm1.4}$	$0.0_{\pm0.0}$	$25.0_{\pm0.0}$	$16.7_{\pm1.4}$	$10.0_{\pm2.4}$	$21.7_{\pm1.4}$	$21.7_{\pm1.4}$	$20.0_{\pm0.0}$	$30.0_{\pm0.0}$
diff	$0.0_{\pm0.0}$	$0.0_{\pm0.0}$	$0.0_{\pm0.0}$	$0.0_{\pm0.0}$	$0.0_{\pm0.0}$	$0.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$
first_word_letter	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$
larger_animal	$70.0_{\pm0.0}$	$91.7_{\pm1.4}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$66.7_{\pm1.4}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$
letters_list	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$
negation	$95.0_{\pm0.0}$	$95.0_{\pm0.0}$	$95.0_{\pm0.0}$	$95.0_{\pm0.0}$	$95.0_{\pm0.0}$	$95.0_{\pm0.0}$	$95.0_{\pm0.0}$	$95.0_{\pm0.0}$	$95.0_{\pm0.0}$	$95.0_{\pm0.0}$
num_to_verbal	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$
object_counting	$55.0_{\pm2.4}$	$56.7_{\pm1.4}$	$48.3_{\pm1.4}$	$61.7_{\pm1.4}$	$66.7_{\pm1.4}$	$51.7_{\pm1.4}$	$63.3_{\pm3.6}$	$70.0_{\pm0.0}$	$70.0_{\pm0.0}$	$73.3_{\pm1.4}$
orthography_starts_with	$20.0_{\pm2.4}$	$35.0_{\pm0.0}$	$61.7_{\pm1.4}$	$78.3_{\pm1.4}$	$70.0_{\pm0.0}$	$43.3_{\pm1.4}$	$70.0_{\pm2.4}$	$75.0_{\pm0.0}$	$78.3_{\pm1.4}$	$78.3_{\pm1.4}$
rhymes	$60.0_{\pm0.0}$	$51.7_{\pm1.4}$	$0.0_{\pm0.0}$	$100.0_{\pm0.0}$	$80.0_{\pm0.0}$	$65.0_{\pm8.2}$	$70.0_{\pm10.8}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$
second_word_letter	$10.0_{\pm2.4}$	$30.0_{\pm0.0}$	$28.3_{\pm1.4}$	$50.0_{\pm0.0}$	$50.0_{\pm0.0}$	$26.7_{\pm8.3}$	$40.0_{\pm0.0}$	$46.7_{\pm1.4}$	$50.0_{\pm0.0}$	$50.0_{\pm0.0}$
sentence_similarity	$20.0_{\pm0.0}$	$21.7_{\pm2.7}$	$40.0_{\pm2.4}$	$46.7_{\pm1.4}$	$53.3_{\pm1.4}$	$5.0_{\pm4.1}$	$18.3_{\pm5.4}$	$45.0_{\pm0.0}$	$51.7_{\pm1.4}$	$56.7_{\pm1.4}$
sentiment	$85.0_{\pm0.0}$	$90.0_{\pm0.0}$	$85.0_{\pm0.0}$	$96.7_{\pm1.4}$	$100.0_{\pm0.0}$	$85.0_{\pm4.1}$	$91.7_{\pm1.4}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$
singular_to_plural	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$
sum	$0.0_{\pm0.0}$	$0.0_{\pm0.0}$	$0.0_{\pm0.0}$	$0.0_{\pm0.0}$	$0.0_{\pm0.0}$	$0.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$
synonyms	$10.0_{\pm0.0}$	$25.0_{\pm0.0}$	$20.0_{\pm0.0}$	$35.0_{\pm0.0}$	$30.0_{\pm0.0}$	$3.3_{\pm1.4}$	$26.7_{\pm1.4}$	$30.0_{\pm0.0}$	$30.0_{\pm0.0}$	$30.0_{\pm0.0}$
taxonomy_animal	$43.3_{\pm3.6}$	$40.0_{\pm2.4}$	$46.7_{\pm1.4}$	$85.0_{\pm2.4}$	$80.0_{\pm0.0}$	$45.0_{\pm6.2}$	$70.0_{\pm4.1}$	$80.0_{\pm0.0}$	$80.0_{\pm0.0}$	$88.3_{\pm2.7}$
translation_en-de	$90.0_{\pm0.0}$	$80.0_{\pm0.0}$	$80.0_{\pm0.0}$	$90.0_{\pm0.0}$	$85.0_{\pm0.0}$	$56.7_{\pm13.0}$	$90.0_{\pm0.0}$	$90.0_{\pm0.0}$	$90.0_{\pm0.0}$	$90.0_{\pm0.0}$
translation_en-es	$90.0_{\pm0.0}$	$100.0_{\pm0.0}$	$96.7_{\pm1.4}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$96.7_{\pm1.4}$	$98.3_{\pm1.4}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$	$100.0_{\pm0.0}$
translation_en-fr	$76.7_{\pm1.4}$	$76.7_{\pm1.4}$	$81.7_{\pm1.4}$	$85.0_{\pm0.0}$	$85.0_{\pm0.0}$	$81.7_{\pm1.4}$	$85.0_{\pm0.0}$	$86.7_{\pm1.4}$	$85.0_{\pm0.0}$	$88.3_{\pm1.4}$
word_sorting	$26.7_{\pm1.4}$	$88.3_{\pm1.4}$	$88.3_{\pm1.4}$	$90.0_{\pm0.0}$	$71.7_{\pm1.4}$	$80.0_{\pm0.0}$	$88.3_{\pm1.4}$	$93.3_{\pm1.4}$	$91.7_{\pm1.4}$	$91.7_{\pm1.4}$
word_unscrambling	$68.3_{\pm1.4}$	$56.7_{\pm1.4}$	$71.7_{\pm1.4}$	$75.0_{\pm0.0}$	$76.7_{\pm1.4}$	$63.3_{\pm3.6}$	$66.7_{\pm1.4}$	$75.0_{\pm0.0}$	$75.0_{\pm0.0}$	$78.3_{\pm2.7}$
# best-performing tasks	7	7	7	13	10	6	10	14	16	**22**

Inf. For Inf, we first sample random permutations of exemplar sequences to be evaluated on the black-box LLM to obtain the scores. Then, we follow the influence metric proposed by Nguyen and Wong [23] to select k individual exemplars to form the best exemplar sequence. The best exemplar sequence is then finally evaluated on the black-box LLM.

Evo. We propose a new baseline using evolutionary strategies. For each iteration, the mutation operator changes one exemplar in the current best exemplar sequence to another random exemplar. Then, we evaluate the exemplars on the black-box LLM. In this way, we exploit the current knowledge about the best exemplars and perform local mutations.

Best-of-N. This is a straightforward baseline that explores the whole space of all exemplar permutations uniformly by sampling random permutations until the T query budget is exhausted.

EASE. The details for our proposed method EASE are described thoroughly in Sec. 4. We use a sampling size of $q = 50000$ exemplar permutations per iteration after OT is introduced.

C More experiment results

C.1 Full results table for the 24 tasks

Tab. 5 is the full table containing all 24 tasks (with more than 100 training data samples) from the Instruction Induction dataset (compared to Tab. 1 which drops tasks with 100% accuracy across baselines). Our EASE outperforms all baselines.

C.2 Full results table for EASE with instructions

We present the full table of results for EASE with joint optimization of exemplars and instructions in Tab. 6 (compared to Tab. 3 which drops tasks with 100% accuracy both with and without instructions). The conclusion is still consistent with the main text that incorporating instructions jointly optimized to best match the chosen exemplars could improve the ICL performance, especially for difficult tasks (e.g., auto categorization, taxonomy animal, etc.).

C.3 Ablation studies for the NeuralUCB and OT components

The following Tab. 7 and Tab. 8 show the necessity of both of the main components, NeuralUCB and OT, in the success of the proposed EASE algorithm. A pure subset selection algorithm based on OT performs badly on its own, especially when the noise ratio is high in the datasets. However, it

Table 6: Average accuracy $\pm$ standard error comparison for EASE with and without jointly optimized instructions.

	EASE	EASE **with instructions**	improvement
antonyms	**90.0**$_{\pm0.0}$	85.0$_{\pm0.0}$	-5.0 ↓
auto_categorization	30.0$_{\pm0.0}$	**46.7**$_{\pm4.9}$	16.7 ↑
diff	**100.0**$_{\pm0.0}$	**100.0**$_{\pm0.0}$	0.0 ∘
larger_animal	**100.0**$_{\pm0.0}$	**100.0**$_{\pm0.0}$	0.0 ∘
negation	95.0$_{\pm0.0}$	**100.0**$_{\pm0.0}$	5.0 ↑
object_counting	73.3$_{\pm1.4}$	**75.0**$_{\pm0.0}$	1.7 ↑
orthography_starts_with	78.3$_{\pm1.4}$	**81.7**$_{\pm1.4}$	3.3 ↑
rhymes	**100.0**$_{\pm0.0}$	91.7$_{\pm3.6}$	-8.3 ↓
second_word_letter	50.0$_{\pm0.0}$	**100.0**$_{\pm0.0}$	50.0 ↑
sentence_similarity	**56.7**$_{\pm1.4}$	**56.7**$_{\pm1.4}$	0.0 ∘
sentiment	**100.0**$_{\pm0.0}$	**100.0**$_{\pm0.0}$	0.0 ∘
sum	**100.0**$_{\pm0.0}$	**100.0**$_{\pm0.0}$	0.0 ∘
synonyms	**30.0**$_{\pm0.0}$	**30.0**$_{\pm0.0}$	0.0 ∘
taxonomy_animal	88.3$_{\pm2.7}$	**100.0**$_{\pm0.0}$	11.7 ↑
translation_en-de	**90.0**$_{\pm0.0}$	**90.0**$_{\pm0.0}$	0.0 ∘
translation_en-es	**100.0**$_{\pm0.0}$	**100.0**$_{\pm0.0}$	0.0 ∘
translation_en-fr	**88.3**$_{\pm1.4}$	85.0$_{\pm0.0}$	-3.3 ↓
word_sorting	**91.7**$_{\pm1.4}$	**91.7**$_{\pm1.4}$	0.0 ∘
word_unscrambling	78.3$_{\pm2.7}$	**80.0**$_{\pm0.0}$	1.7 ↑
linear_4_10_noisy	**73.3**$_{\pm3.6}$	41.7$_{\pm9.5}$	-31.7 ↓
pig_latin_variant2_encode_10_noisy	75.0$_{\pm2.4}$	**85.0**$_{\pm2.4}$	10.0 ↑
ag_news_textlabel_redirect1_10_noisy	56.7$_{\pm2.7}$	**65.0**$_{\pm0.0}$	8.3 ↑
sst5_reverse_10_noisy	**50.0**$_{\pm0.0}$	**50.0**$_{\pm0.0}$	0.0 ∘

greatly improves the performance of NeuralUCB when used together with NeuralUCB in our EASE. This is attributed to the ability to examine more permutations of the exemplars without increasing the computational cost, since only selected permutations through OT (which places an implicit bias towards more relevant exemplars) will be evaluated for the acquisition values in NeuralUCB.

Table 7: Average accuracy $\pm$ standard error for ablation studies of the OT and NeuralUCB components in EASE.

Task	Noise	OT	NeuralUCB	EASE
LR	0%	50.0$_{\pm0.0}$	68.3$_{\pm1.4}$	**81.7**$_{\pm3.6}$
	10%	48.3$_{\pm1.4}$	66.7$_{\pm3.6}$	**73.3**$_{\pm3.6}$
	30%	46.7$_{\pm1.4}$	65.0$_{\pm4.1}$	**78.3**$_{\pm1.4}$
	50%	45.0$_{\pm0.0}$	65.0$_{\pm0.0}$	**71.7**$_{\pm2.7}$
	70%	38.3$_{\pm2.7}$	58.3$_{\pm2.7}$	**66.7**$_{\pm3.6}$
	90%	26.7$_{\pm1.4}$	30.0$_{\pm2.4}$	**53.3**$_{\pm2.7}$
LP-variant	0%	41.7$_{\pm1.4}$	73.3$_{\pm1.4}$	**75.0**$_{\pm0.0}$
	10%	40.0$_{\pm0.0}$	**75.0**$_{\pm0.0}$	75.0$_{\pm2.4}$
	30%	40.0$_{\pm2.4}$	70.0$_{\pm0.0}$	**73.3**$_{\pm1.4}$
	50%	35.0$_{\pm2.4}$	71.7$_{\pm1.4}$	**76.7**$_{\pm2.7}$
	70%	35.0$_{\pm0.0}$	71.7$_{\pm2.7}$	**75.0**$_{\pm0.0}$
	90%	50.0$_{\pm0.0}$	61.7$_{\pm1.4}$	**63.3**$_{\pm1.4}$
AG News Remap	0%	26.7$_{\pm1.4}$	51.7$_{\pm1.4}$	**53.3**$_{\pm3.6}$
	10%	15.0$_{\pm0.0}$	50.0$_{\pm0.0}$	**56.7**$_{\pm2.7}$
	30%	5.0$_{\pm0.0}$	**51.7**$_{\pm1.4}$	51.7$_{\pm1.4}$
	50%	5.0$_{\pm0.0}$	48.3$_{\pm1.4}$	**56.7**$_{\pm1.4}$
	70%	8.3$_{\pm1.4}$	51.7$_{\pm3.6}$	51.7$_{\pm1.4}$
	90%	5.0$_{\pm0.0}$	41.7$_{\pm3.6}$	**55.0**$_{\pm2.4}$
SST5 Reverse	0%	13.3$_{\pm1.4}$	43.3$_{\pm1.4}$	**50.0**$_{\pm0.0}$
	10%	15.0$_{\pm0.0}$	41.7$_{\pm3.6}$	**50.0**$_{\pm0.0}$
	30%	25.0$_{\pm2.4}$	**41.7**$_{\pm2.7}$	41.7$_{\pm3.6}$
	50%	15.0$_{\pm0.0}$	33.3$_{\pm2.7}$	**43.3**$_{\pm1.4}$
	70%	23.3$_{\pm1.4}$	40.0$_{\pm4.1}$	**45.0**$_{\pm2.4}$
	90%	20.0$_{\pm0.0}$	30.0$_{\pm0.0}$	**31.7**$_{\pm1.4}$

Table 8: Average test accuracy $\pm$ standard error for ablation studies of the OT and NeuralUCB components in EASE.

Task	Noise	OT	NeuralUCB	EASE
LR	0%	34.0$_{\pm0.8}$	48.7$_{\pm1.2}$	**54.0**$_{\pm4.5}$
	10%	29.3$_{\pm0.3}$	45.7$_{\pm2.6}$	**48.3**$_{\pm1.5}$
	30%	30.0$_{\pm0.8}$	46.7$_{\pm2.7}$	**51.0**$_{\pm1.6}$
	50%	23.0$_{\pm0.5}$	44.7$_{\pm1.7}$	**55.0**$_{\pm1.9}$
	70%	28.7$_{\pm0.5}$	42.7$_{\pm2.3}$	**45.7**$_{\pm1.4}$
	90%	**32.0**$_{\pm0.5}$	21.7$_{\pm2.2}$	30.0$_{\pm3.3}$
LP-variant	0%	34.7$_{\pm0.5}$	**56.0**$_{\pm1.9}$	49.3$_{\pm2.0}$
	10%	34.0$_{\pm0.5}$	**55.3**$_{\pm1.0}$	46.0$_{\pm0.8}$
	30%	36.0$_{\pm0.9}$	44.3$_{\pm1.8}$	**49.3**$_{\pm1.1}$
	50%	31.0$_{\pm0.5}$	47.0$_{\pm4.0}$	**54.3**$_{\pm2.1}$
	70%	30.3$_{\pm0.7}$	44.0$_{\pm3.3}$	**49.0**$_{\pm3.3}$
	90%	39.7$_{\pm0.3}$	**42.7**$_{\pm1.0}$	41.3$_{\pm1.9}$
AG News Remap	0%	12.3$_{\pm0.5}$	**33.7**$_{\pm1.8}$	29.7$_{\pm1.9}$
	10%	13.0$_{\pm0.9}$	29.7$_{\pm2.7}$	**36.3**$_{\pm1.0}$
	30%	4.7$_{\pm0.3}$	**37.0**$_{\pm2.2}$	32.3$_{\pm3.7}$
	50%	3.7$_{\pm0.3}$	31.7$_{\pm1.8}$	**43.7**$_{\pm2.8}$
	70%	7.0$_{\pm0.0}$	**38.3**$_{\pm5.0}$	31.0$_{\pm2.5}$
	90%	2.0$_{\pm0.0}$	30.0$_{\pm2.5}$	**40.7**$_{\pm3.1}$
SST5 Reverse	0%	9.7$_{\pm0.7}$	**36.0**$_{\pm3.4}$	32.3$_{\pm2.8}$
	10%	9.3$_{\pm0.3}$	26.3$_{\pm7.1}$	**30.7**$_{\pm1.9}$
	30%	11.7$_{\pm0.5}$	25.7$_{\pm2.0}$	**26.3**$_{\pm2.2}$
	50%	12.0$_{\pm0.0}$	17.3$_{\pm4.0}$	**29.0**$_{\pm2.9}$
	70%	14.3$_{\pm0.3}$	26.3$_{\pm4.8}$	**33.0**$_{\pm1.4}$
	90%	**16.0**$_{\pm0.5}$	12.0$_{\pm0.8}$	11.3$_{\pm1.2}$

C.4 Ablation studies for the speedups from OT

Carrying on from the previous section about the necessity of both the NeuralUCB and OT components, we can quantify through experiments the amount of speedups brought by OT. Specifically, we measure the wall clock time speedups for different sizes of the domain space Q_t that we sample. As shown in Tab. 9, the larger the domain space $|Q_t|$, the greater the relative gain in the speedups. This mitigates the computational issues of applying NeuralUCB to the problem of exemplar selection and contributes positively to the success of EASE.

Table 9: Wall clock time speedups of using OT. The time in the table measures the wall clock time for each iteration of the algorithms.

Domain size $|Q_t|$	Time without OT (Sec)	Time with OT (Sec)	Speedup
5000	$17.75_{\pm 0.05}$	$4.35_{\pm 0.03}$	$4.1\times$
10000	$35.69_{\pm 0.13}$	$5.44_{\pm 0.02}$	$6.6\times$
20000	$71.57_{\pm 0.14}$	$7.86_{\pm 0.07}$	$9.1\times$
50000	$179.39_{\pm 0.12}$	$14.40_{\pm 0.02}$	$12.5\times$

C.5 Ablation studies for other numbers of k

We validate that EASE is able to select a larger set of exemplars (i.e., larger k) and also from a larger exemplar set of size n. We conduct comparisons of EASE to the two most competitive baselines, Evo and Best-of-N, on the AG News Remap and SST5 Reverse datasets of size $n = 1000$ with 10% noise. According to Tab. 10 and Tab. 11, EASE performs much better than the two baselines. Therefore, EASE is able to work in other general setups with a larger k and n.

We could potentially choose more exemplars to maximize the context window, but this will incur a very high future cost of inference (i.e., the cost of actually using the exemplars for inference during production). For example, to maximize the GPT-3.5-Turbo's context window of 16385 tokens (which OpenAI charges US$0.5 per 1M tokens), each inference will cost US$0.0081925 and each run of the algorithm costs US$27.04 (assuming 165 iterations of query budget and 20 validation data samples). Similarly for GPT-4-Turbo with a context window of 128000 tokens (which OpenAI charges US$10 per 1M tokens), each inference of the algorithm now costs US$1.28 and each run of the algorithm costs US$4224. Therefore, while it is theoretically possible to use even larger k which exhausts the context window, it is not economically viable to conduct experiments or adopt such an approach in practice.

Table 10: Average validation accuracy $\pm$ standard error for $k = 50$ and $n = 1000$.

	Evo	Best-of-N	EASE
AG News Remap	$50.0_{\pm 2.4}$	$55.0_{\pm 2.4}$	$\mathbf{66.7_{\pm 1.4}}$
SST5 Reverse	$58.3_{\pm 1.4}$	$56.7_{\pm 1.4}$	$\mathbf{65.0_{\pm 2.4}}$

Table 11: Average test accuracy $\pm$ standard error for $k = 50$ and $n = 1000$.

	Evo	Best-of-N	EASE
AG News Remap	$12.7_{\pm 2.6}$	$26.3_{\pm 0.7}$	$\mathbf{37.3_{\pm 2.4}}$
SST5 Reverse	$39.0_{\pm 0.5}$	$38.7_{\pm 1.0}$	$\mathbf{40.7_{\pm 1.0}}$

C.6 Ablation studies for the benefit of having an ordering-aware embedding in EASE

Note that even for the same subset of exemplars, a different ordering leads to different pre-trained embedding from embedding model $h(\cdot)$. In this section, we investigate the impact of capturing the ordering of the exemplars in the exemplar sequence E. To contrast the adopted approach, we propose a new embedding that simply averages the embeddings of all exemplars in the chosen subset. So, this embedding will be invariant with regard to the ordering of exemplars and we call it AvgEmb. As demonstrated in Tab. 12 and Tab. 13, adopting an ordering-aware embedding in EASE results in better overall performance as compared to AvgEmd which disregards exemplar ordering.

C.7 Ablation studies for optimizing exemplars for different black-box LLMs in EASE

We perform exemplar selection for other target black-box models that are not GPT-3.5 which we used for all other experiments previously. We use GPT-4-V (i.e., "gpt-4-turbo-2024-04-09" with vision

Table 12: Average validation accuracy ± standard error for ablation studies of using an embedding without considering order.

Task	Noise	AvgEmb	EASE
LR	0%	76.7±1.4	**81.7±3.6**
	10%	73.3±3.6	73.3±3.6
	30%	70.0±0.0	**78.3±1.4**
	50%	68.3±3.6	**71.7±2.7**
	70%	**66.7±1.4**	66.7±3.6
	90%	51.7±1.4	**53.3±2.7**
LP-variant	0%	**76.7±1.4**	75.0±0.0
	10%	**75.0±0.0**	75.0±2.4
	30%	70.0±0.0	**73.3±1.4**
	50%	**78.3±1.4**	76.7±2.7
	70%	75.0±2.4	**75.0±0.0**
	90%	61.7±1.4	**63.3±1.4**
AG News Remap	0%	**53.3±1.4**	53.3±3.6
	10%	50.0±4.1	**56.7±2.7**
	30%	**51.7±1.4**	**51.7±1.4**
	50%	50.0±0.0	**56.7±1.4**
	70%	**58.3±1.4**	51.7±1.4
	90%	53.3±4.9	**55.0±2.4**
SST5 Reverse	0%	46.7±5.4	**50.0±0.0**
	10%	50.0±2.4	**50.0±0.0**
	30%	**45.0±2.4**	41.7±3.6
	50%	**48.3±1.4**	43.3±1.4
	70%	41.7±3.6	**45.0±2.4**
	90%	**31.7±1.4**	**31.7±1.4**
# best-performing tasks		11	16

Table 13: Average test accuracy ± standard error for ablation studies of using an embedding without considering order.

Task	Noise	AvgEmb	EASE
LR	0%	48.7±1.0	**54.0±4.5**
	10%	**48.7±2.3**	48.3±1.5
	30%	**53.0±4.0**	51.0±1.6
	50%	46.0±4.1	**55.0±1.9**
	70%	44.3±5.2	**45.7±1.4**
	90%	28.7±2.2	**30.0±3.3**
LP-variant	0%	47.7±1.5	**49.3±2.0**
	10%	**48.0±1.4**	46.0±0.8
	30%	49.3±1.4	**49.3±1.1**
	50%	48.0±1.2	**54.3±2.1**
	70%	**54.3±1.1**	49.0±3.3
	90%	38.3±0.7	**41.3±1.9**
AG News Remap	0%	**33.3±1.9**	29.7±1.9
	10%	32.0±0.5	**36.3±1.0**
	30%	29.7±1.0	**32.3±3.7**
	50%	28.7±1.2	**43.7±2.8**
	70%	**40.0±4.1**	31.0±2.5
	90%	36.3±3.3	**40.7±3.1**
SST5 Reverse	0%	31.0±3.7	**32.3±2.8**
	10%	**31.3±2.4**	30.7±1.9
	30%	**27.7±1.2**	26.3±2.2
	50%	**33.7±1.2**	29.0±2.9
	70%	25.7±5.9	**33.0±1.4**
	90%	**13.7±0.3**	11.3±1.2
# best-performing tasks		10	14

capability), GPT-4-Turbo (i.e., 'gpt-4-1106-preview' without vision capability) and Gemini Pro [32] (i.e., "gemini-1.0-pro") for our experiments. Tab. 14 and Tab. 15 show that our EASE is able to select effective exemplars for different black-box models. Note that the performance of GPT-4-V and GPT-4-Turbo is comparable to or worse than that of GPT-3.5 in some tasks. This may be attributed to significant variations in performance across different versions (i.e., checkpoints) of the GPT models. For example, existing work by Chen et al. [5] has shown that GPT-4's mathematical ability varies a lot across different checkpoints, with some exhibiting poor performance on mathematical problems. This variability partially explains the suboptimal performance of GPT-4-V and GPT-4-Turbo on the task of LR. Additionally, it is worth investigating, in future work, the significance of exemplar selection as the LLMs continue to become increasingly powerful.

Table 14: Average validation accuracy ± standard error when using our EASE to select exemplars for different target black-box models.

Task (with 10% noise)	GPT-4-V	GPT-4-Turbo	Gemini Pro
LR	3.3±1.4	0.0±0.0	80.0±2.4
LP-variant	88.3±1.4	86.7±1.4	31.7±1.4
AG News Remap	36.7±2.7	30.0±0.0	53.3±1.4
SST5 Reverse	21.7±2.7	48.3±1.4	43.3±1.4

Table 15: Average test accuracy ± standard error when using our EASE to select exemplars for different target black-box models.

Task (with 10% noise)	GPT-4-V	GPT-4-Turbo	Gemini Pro
LR	0.0±0.0	0.0±0.0	56.3±0.7
LP-variant	79.3±1.4	73.3±0.7	15.3±1.7
AG News Remap	20.7±3.7	26.7±1.5	37.0±0.8
SST5 Reverse	12.3±1.7	30.7±1.9	19.7±1.4

C.8 Ablation studies for using different embedding models in EASE

For our main experiments, we use MPNet as the embedding model for our EASE. Here, we use MiniLM [34] and CLIP [26] to obtain the embedding for our EASE, respectively. Tab. 16 and Tab. 17 show that our EASE achieves competitive performance using different embedding models. Therefore, our EASE is general in the sense that different embedding models can be used.

Table 16: Average validation accuracy $\pm$ standard error when using different embedding models for our EASE.

Task (with 10% noise)	MiniLM	CLIP
LR	$75.0_{\pm 0.0}$	$66.7_{\pm 1.4}$
LP-variant	$78.3_{\pm 1.4}$	$70.0_{\pm 0.0}$
AG News Remap	$46.7_{\pm 1.4}$	$45.0_{\pm 0.0}$
SST5 Reverse	$43.3_{\pm 2.7}$	$46.7_{\pm 1.4}$

Table 17: Average test accuracy $\pm$ standard error when using different embedding models for our EASE.

Task (with 10% noise)	MiniLM	CLIP
LR	$49.7_{\pm 3.3}$	$45.3_{\pm 4.1}$
LP-variant	$48.0_{\pm 2.2}$	$51.3_{\pm 1.2}$
AG News Remap	$27.0_{\pm 2.9}$	$31.0_{\pm 1.2}$
SST5 Reverse	$36.7_{\pm 1.9}$	$31.7_{\pm 1.2}$

C.9 Ablation studies: Asking GPT to directly select exemplars

One may wonder whether ChatGPT can directly help us select the exemplars for in-context learning. To test the possibility of directly utilizing ChatGPT, we use the following prompt to query the GPT:

> *"You are asked to perform a task that is described by the examples below, the goal is to correctly give output based on the input. Given the numbered list of examples below, pick 5 of them that will best serve as examples for in-context learning for this task. Only output the list of numbers."*

We provide all the exemplars in D in the form of a numbered list to GPT together with the prompt above, and obtain the GPT-selected exemplars. We call this method GPT Select. As shown in Tab. 18 and Tab. 19, the exemplars directly selected by GPT perform worse than EASE for ICL.

Table 18: Average validation accuracy $\pm$ standard error when asking GPT to directly output the best exemplars.

Task (with 10% noise)	GPT Select	EASE
LR	$35.0_{\pm 6.2}$	$\mathbf{73.3}_{\pm 3.6}$
LP-variant	$58.3_{\pm 2.7}$	$\mathbf{75.0}_{\pm 2.4}$
AG News Remap	$10.0_{\pm 0.0}$	$\mathbf{56.7}_{\pm 2.7}$
SST5 Reverse	$10.0_{\pm 0.0}$	$\mathbf{50.0}_{\pm 0.0}$

Table 19: Average test accuracy $\pm$ standard error when asking GPT to directly output the best exemplars.

Task (with 10% noise)	GPT Select	EASE
LR	$29.0_{\pm 7.4}$	$\mathbf{48.3}_{\pm 1.5}$
LP-variant	$44.0_{\pm 0.8}$	$\mathbf{46.0}_{\pm 0.8}$
AG News Remap	$12.7_{\pm 0.3}$	$\mathbf{36.3}_{\pm 1.0}$
SST5 Reverse	$10.0_{\pm 0.0}$	$\mathbf{30.7}_{\pm 1.9}$

C.10 Test accuracies for all tables

We present the test accuracies for all tables presented in the main paper. Note that we report validation accuracies in the main tables because the effectiveness of the optimization strategy is directly reflected by the maximized value of the objective function in (1) at the end of all iterations. Nevertheless, we show all test accuracies here for reference and completeness. The test accuracy can be affected by the difference in the distribution of the validation and test data, which is out of our control. This difference in distribution is significant in our case because the validation set only contains 20 data points (limited by the fact that querying the GPT-3.5 API is expensive). This setup is disadvantageous for EASE because the optimized exemplar is "overfitted" to the validation set. This test performance is expected to improve much if we use a larger validation set (e.g., with 100 validation samples).

Table 20: Test accuracies counterpart of Tab. 1 over 3 independent trials.

	DPP	MMD	OT	Cosine	BM25	Active	Inf	Evo	Best-of-N	EASE
active_to_passive	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0
antonyms	75.7±0.3	**86.0**±0.0	85.3±0.5	77.7±0.3	80.7±0.7	80.0±2.5	82.3±0.3	82.0±0.0	84.3±0.3	81.0±0.0
auto_categorization	28.7±0.3	25.7±0.5	24.0±0.5	**39.3**±0.7	27.0±0.8	32.7±1.0	35.0±1.2	17.3±0.3	30.7±0.5	31.3±0.5
diff	2.0±0.0	2.0±0.0	2.0±0.0	2.0±0.0	2.0±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0
first_word_letter	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0
larger_animal	69.0±0.5	78.7±1.0	83.7±0.5	77.7±0.5	84.0±0.8	61.0±1.7	84.7±0.3	**90.0**±0.5	87.7±2.3	87.7±1.5
letters_list	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0
negation	85.7±0.5	87.3±0.5	86.3±0.3	87.3±0.3	83.3±0.3	85.7±1.1	83.7±0.3	86.3±0.3	85.0±0.5	**89.0**±0.5
num_to_verbal	97.3±0.3	98.7±0.3	**99.0**±0.0	**99.0**±0.0	96.0±0.0	97.7±0.7	98.0±0.0	96.7±0.3	96.7±0.3	96.3±0.3
object_counting	45.3±0.7	48.3±0.7	40.7±0.3	27.0±2.0	31.3±3.1	48.0±2.0	**58.0**±0.5	42.3±0.3	52.3±2.7	55.3±5.2
orthography_starts_with	30.3±0.7	42.0±0.5	65.7±0.3	65.7±0.7	**71.0**±0.5	40.3±7.8	69.3±0.7	47.0±0.8	63.3±2.0	67.3±1.1
rhymes	58.0±0.8	42.0±1.5	11.3±0.5	**99.0**±0.0	64.3±1.5	47.3±6.9	59.3±12.6	98.3±0.3	96.0±0.0	89.7±1.9
second_word_letter	17.3±0.7	31.0±0.0	32.3±0.5	42.0±2.5	42.3±0.7	27.0±8.2	**45.7**±1.0	40.3±0.3	38.7±0.3	42.0±0.5
sentence_similarity	19.0±0.5	13.7±0.5	38.3±1.2	**41.0**±0.5	33.7±1.9	19.0±1.7	20.0±4.1	18.0±0.0	31.3±2.2	30.0±2.2
sentiment	91.3±0.3	89.0±0.0	87.0±0.0	90.3±0.3	90.7±0.5	**91.7**±0.7	89.0±0.0	89.7±0.3	89.3±0.3	88.0±0.8
singular_to_plural	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0
sum	2.0±0.0	2.0±0.0	2.0±0.0	2.0±0.0	2.0±0.0	34.7±26.7	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0
synonyms	14.0±0.5	14.3±0.5	11.0±0.0	14.0±0.0	13.7±0.3	13.0±0.5	14.7±0.7	13.3±0.7	**18.3**±0.3	14.0±1.2
taxonomy_animal	47.7±0.3	44.7±1.0	**77.3**±1.5	73.3±0.3	67.3±4.8	46.3±7.1	62.3±7.3	32.7±0.5	73.7±3.4	74.7±5.5
translation_en-de	83.7±0.3	84.3±0.3	83.3±0.3	83.0±0.5	83.3±1.1	80.3±1.9	**85.0**±0.9	83.3±0.3	82.3±0.3	83.7±0.5
translation_en-es	83.7±0.3	**89.7**±0.3	88.3±0.3	87.3±1.0	87.3±1.0	88.0±0.8	86.7±1.2	84.3±0.3	84.3±0.3	85.0±0.0
translation_en-fr	83.0±0.0	87.7±0.3	87.3±0.3	87.0±0.0	87.0±0.0	83.3±1.7	87.7±0.5	86.7±0.3	**88.3**±0.3	87.3±0.7
word_sorting	8.7±0.5	65.0±0.5	66.7±1.0	66.0±0.9	43.7±1.9	**71.0**±1.4	69.3±1.4	61.3±1.0	63.3±0.5	68.0±2.0
word_unscrambling	61.3±0.3	58.7±0.5	**63.3**±0.3	57.0±0.9	61.3±0.5	60.7±1.7	58.3±1.2	53.7±0.3	60.3±0.3	61.7±1.0
# best-performing tasks	4	6	7	8	5	7	9	7	8	7

Table 21: Test accuracies counterpart of Tab. 2 over 3 independent trials.

Type	Task	Noise	DPP	MMD	OT	Cosine	BM25	Active	Inf	Evo	Best-of-N	EASE
Rule-based tasks	LR	0%	39.3±0.5	35.0±0.5	34.7±0.5	47.3±0.3	36.3±1.2	34.0±3.7	**53.3**±1.1	38.3±0.7	51.7±2.7	49.3±2.0
		10%	0.0±0.0	37.0±0.5	34.0±0.5	49.0±0.9	39.3±1.8	39.0±0.8	**52.3**±2.1	37.7±0.5	48.0±0.9	46.0±0.8
		30%	0.0±0.0	44.3±0.5	36.0±0.9	38.7±3.5	34.3±1.5	36.7±2.4	48.0±2.5	35.7±0.3	**49.3**±0.5	49.3±1.1
		50%	0.0±0.0	53.3±1.0	31.0±0.5	47.0±0.5	34.3±1.1	12.7±5.2	42.7±1.7	34.0±0.5	49.3±0.3	**54.3**±2.1
		70%	0.0±0.0	39.0±1.2	30.3±0.7	46.7±1.1	36.3±0.5	4.3±3.5	**51.0**±1.4	33.3±0.5	31.3±0.7	49.0±3.3
		90%	0.0±0.0	35.0±0.5	39.7±0.3	35.3±1.5	0.0±0.0	0.0±0.0	41.0±1.4	16.7±0.7	28.0±0.5	**41.3**±1.9
	LP-variant	0%	39.3±0.5	35.0±0.5	34.7±0.5	47.3±0.3	36.3±1.2	34.0±3.7	**53.3**±1.1	38.3±0.7	51.7±2.7	49.3±2.0
		10%	0.0±0.0	37.0±0.5	34.0±0.5	49.0±0.9	39.3±1.8	39.0±0.8	**52.3**±2.1	37.7±0.5	48.0±0.9	46.0±0.8
		30%	0.0±0.0	44.3±0.5	36.0±0.9	38.7±3.5	34.3±1.5	36.7±2.4	48.0±2.5	35.7±0.3	**49.3**±0.5	49.3±1.1
		50%	0.0±0.0	53.3±1.0	31.0±0.5	47.0±0.5	34.3±1.1	12.7±5.2	42.7±1.7	34.0±0.5	49.3±0.3	54.3±2.1
		70%	0.0±0.0	39.0±1.2	30.3±0.7	46.7±1.1	36.3±0.5	4.3±3.5	**51.0**±1.4	33.3±0.5	31.3±0.7	49.0±3.3
		90%	0.0±0.0	35.0±0.5	39.7±0.3	35.3±1.5	0.0±0.0	0.0±0.0	41.0±1.4	16.7±0.7	28.0±0.5	**41.3**±1.9
Re-mapped label tasks	AG News Remap	0%	7.0±0.5	7.0±0.0	12.3±0.5	**30.3**±1.2	28.3±1.1	4.3±1.1	9.0±2.0	11.3±0.3	19.3±0.5	29.7±1.9
		10%	2.0±0.0	7.0±0.0	13.0±0.9	17.0±3.3	19.0±3.7	6.0±0.8	13.0±2.6	12.0±0.5	30.0±0.5	**36.3**±1.0
		30%	3.7±0.3	1.0±0.0	4.7±0.3	28.0±1.6	22.3±0.5	7.7±2.2	5.0±0.5	12.3±0.3	28.0±4.1	**32.3**±3.7
		50%	4.7±0.3	3.0±0.0	3.7±0.3	27.3±1.9	21.3±2.2	8.7±3.1	13.0±2.9	6.7±0.3	21.7±1.9	**43.7**±2.8
		70%	2.0±0.0	23.3±0.3	7.0±0.0	20.0±0.8	16.0±0.5	15.7±7.1	5.0±0.9	4.7±0.3	**36.0**±0.5	31.0±2.5
		90%	8.0±0.5	21.0±0.9	2.0±0.0	22.7±5.7	2.0±0.0	11.7±1.9	23.0±1.2	6.3±0.3	27.3±1.5	**40.7**±3.1
	SST5 Reverse	0%	13.3±0.5	11.7±0.5	9.7±0.7	22.0±2.9	18.3±0.5	10.3±0.3	21.7±4.5	11.3±0.3	23.7±0.5	**32.3**±2.8
		10%	13.3±0.7	11.3±0.3	9.3±0.3	20.0±2.2	18.0±0.8	11.7±1.5	27.0±0.9	12.0±0.5	23.0±0.5	**30.7**±1.9
		30%	15.7±0.3	13.3±0.3	11.7±0.5	19.3±1.9	23.7±0.5	9.7±0.3	21.3±2.2	11.0±0.5	13.0±1.2	**26.3**±2.2
		50%	13.0±0.5	12.3±0.3	12.0±0.0	18.3±2.4	14.0±0.5	12.7±0.7	14.3±0.7	9.3±0.3	14.0±1.6	**29.0**±2.9
		70%	12.0±0.0	12.0±0.5	14.3±0.3	24.7±0.5	12.0±0.9	11.7±0.5	18.7±2.3	15.7±0.3	**33.0**±0.8	33.0±1.4
		90%	**16.0**±0.5	9.3±0.5	**16.0**±0.5	12.7±0.3	13.3±0.7	12.7±0.3	14.0±0.5	11.7±0.5	10.7±0.3	11.3±1.2
	# best-performing tasks		1	0	1	1	0	0	6	0	4	13

22

Table 22: Test accuracies counterpart of Tab. 3 over 3 independent trials.

	EASE	EASE **with instructions**	improvement
antonyms	**81.0**$_{\pm 0.0}$	80.0$_{\pm 1.2}$	-1.0 ↓
auto_categorization	31.3$_{\pm 0.5}$	**35.3**$_{\pm 4.5}$	4.0 ↑
diff	**100.0**$_{\pm 0.0}$	**100.0**$_{\pm 0.0}$	0.0 ○
larger_animal	**87.7**$_{\pm 1.5}$	62.0$_{\pm 0.5}$	-25.7 ↓
negation	**89.0**$_{\pm 0.5}$	86.7$_{\pm 0.7}$	-2.3 ↓
object_counting	55.3$_{\pm 5.2}$	**61.3**$_{\pm 1.4}$	6.0 ↑
orthography_starts_with	67.3$_{\pm 1.1}$	**72.3**$_{\pm 1.4}$	5.0 ↑
rhymes	**89.7**$_{\pm 1.9}$	65.7$_{\pm 2.7}$	-24.0 ↓
second_word_letter	42.0$_{\pm 0.5}$	**100.0**$_{\pm 0.0}$	58.0 ↑
sentence_similarity	30.0$_{\pm 2.2}$	**33.7**$_{\pm 2.6}$	3.7 ↑
sentiment	88.0$_{\pm 0.8}$	**92.3**$_{\pm 0.5}$	4.3 ↑
sum	**100.0**$_{\pm 0.0}$	**100.0**$_{\pm 0.0}$	0.0 ○
synonyms	14.0$_{\pm 1.2}$	**14.3**$_{\pm 0.7}$	0.3 ↑
taxonomy_animal	74.7$_{\pm 5.5}$	**75.0**$_{\pm 2.5}$	0.3 ↑
translation_en-de	**83.7**$_{\pm 0.5}$	83.0$_{\pm 0.0}$	-0.7 ↓
translation_en-es	85.0$_{\pm 0.0}$	**88.7**$_{\pm 0.7}$	3.7 ↑
translation_en-fr	**87.3**$_{\pm 0.7}$	**87.3**$_{\pm 0.5}$	0.0 ○
word_sorting	68.0$_{\pm 2.0}$	**73.7**$_{\pm 0.7}$	5.7 ↑
word_unscrambling	**61.7**$_{\pm 1.0}$	60.7$_{\pm 0.3}$	-1.0 ↓
linear_4_10_noisy	**48.3**$_{\pm 1.5}$	27.0$_{\pm 10.5}$	-21.3 ↓
pig_latin_variant2_encode_10_noisy	46.0$_{\pm 0.8}$	**57.3**$_{\pm 1.0}$	11.3 ↑
ag_news_textlabel_redirect1_10_noisy	36.3$_{\pm 1.0}$	**41.0**$_{\pm 2.4}$	4.7 ↑
sst5_reverse_10_noisy	30.7$_{\pm 1.9}$	**35.0**$_{\pm 0.8}$	4.3 ↑

Table 23: Test accuracies counterpart of Tab. 4 over 3 independent trials.

AG News Remap (10% noise)

Size n	EASE	EASE with retrieval
1000	27.0$_{\pm 7.8}$	**41.7**$_{\pm 2.8}$
10000	34.0$_{\pm 4.5}$	**36.7**$_{\pm 2.8}$
50000	37.7$_{\pm 2.9}$	**38.3**$_{\pm 2.6}$
100000	32.0$_{\pm 5.7}$	**37.3**$_{\pm 2.8}$

SST5 Reverse (10% noise)

Size n	EASE	EASE with retrieval
1000	**37.3**$_{\pm 4.3}$	34.5$_{\pm 0.4}$
3000	33.5$_{\pm 1.8}$	**34.3**$_{\pm 5.8}$
5000	32.3$_{\pm 1.4}$	**34.0**$_{\pm 0.5}$
7000	30.3$_{\pm 4.0}$	**39.5**$_{\pm 0.4}$