# IclForge: Enhancing In-Context Learning with Evolutionary Algorithms under Budgeted Annotation

Vijit Malik
Amazon
vijitvm@amazon.com

Atul Pande[*]
Amazon
atuypand@amazon.com

Anirban Majumder
Amazon
majumda@amazon.com

## Abstract

In-context learning (ICL) has emerged as a powerful paradigm for adapting Large Language Models (LLMs) to specific tasks without parameter updates. While various strategies exist for selecting relevant ICL exemplars from a labeled pool, the fundamental challenge of constructing this high-quality pool remains largely unexplored, especially for new tasks or domains with limited labeled data. We present IclForge , a novel active learning framework that efficiently selects informative examples from unlabeled datasets to be annotated and included in the ICL pool. Unlike traditional active learning methods that optimize for individual example informativeness, IclForge explicitly considers the interdependence of examples within the ICL context. Through extensive experiments across diverse datasets and LLM architectures, we show that IclForge outperforms standard active learning baselines by +180−450 basis points while requiring 50% fewer annotations. Our framework is complementary to existing ICL selection strategies and extends naturally to generative applications, which we demonstrate through experiments on Math Word Problem (MWP) tasks. These results highlight IclForge 's effectiveness in constructing high-quality ICL exemplar pools in resource-constrained scenarios.

## CCS Concepts

• **Computing methodologies → Active learning settings**; **Natural language processing**.

## Keywords

Large Language Models, Active Learning, In-Context Learning, Evolutionary Algorithms

**ACM Reference Format:**
Vijit Malik, Atul Pande, and Anirban Majumder. 2025. IclForge: Enhancing In-Context Learning with Evolutionary Algorithms under Budgeted Annotation. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management (CIKM '25), November 10–14, 2025, Seoul, Republic of Korea.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3746252.3761536

[*]Work done during internship at Amazon.

## 1 Introduction

Large Language Models (LLMs) [3, 11] have shown remarkable performance in many real-world applications ranging from chatbots to content generation and information retrieval. Their effectiveness relies on in-context learning (ICL), where few carefully selected exemplars guide the model toward desired outcomes. Unlike traditional supervised learning, incorporating ICL in a prompt doesn't require any parameter update - an effective solution where fine-tuning of LLM is either prohibitively expensive or not a viable option (LLMs with API-only access). Despite their effectiveness, the performance of ICL greatly depends on the selection of exemplars [38] and their ordering in the prompt [20], with different sets of ICLs yielding performance ranging from nearly random to comparable with state-of-the-art LLMs [12].

Traditional research has focused on selecting ICL exemplars through pre-trained or fine-tuned retriever models [2, 13, 19, 28]. The retriever is invoked per query to obtain a set of exemplars most relevant to the query. Such approaches require inference-time retrieval of ICLs which may hinder their adoption because of additional latency overhead. To address this latency concern, recent research [17, 25] has focused on selecting a static set of ICL that will work well for all queries during the inference time. However, all previous approaches share a critical assumption that they have access to a large pool of high-quality ICL examples to retrieve from. In real-world settings, this assumption doesn't hold for multiple reasons. First, new tasks often face limited labeled data. Second, available ICL exemplars may not adequately cover the full spectrum of input patterns and edge cases encountered in production. Further, real-world data distributions evolve over time, requiring frequent updates to the ICL pool. Finally, maintaining consistent quality standards across a large exemplar pool becomes increasingly challenging as applications scale.

To address these challenges, we propose a novel application of Active Learning [8, 31] to the domain of ICLs, efficiently identifying and annotating the most informative examples. Active learning for ICL ([22], [23]) presents unique challenges: the selected examples must not only be informative individually but must also work effectively as a collective set within the ICL framework. The framework must optimize for multiple objectives - relevance and diversity among selected examples, while being adaptable to different task-specific datasets and LLM architectures.

We present IclForge , a novel active learning framework specifically designed to generate high-quality ICL exemplars from unlabeled datasets. IclForge combines several key innovations: 1) an uncertainty-based pseudo-labeling strategy to bootstrap the selection process, 2) a batch-wise algorithm that optimizes for diversity in the initial data selection and 3) a genetic algorithm-based selection mechanism that iteratively evolves sets of candidate exemplars

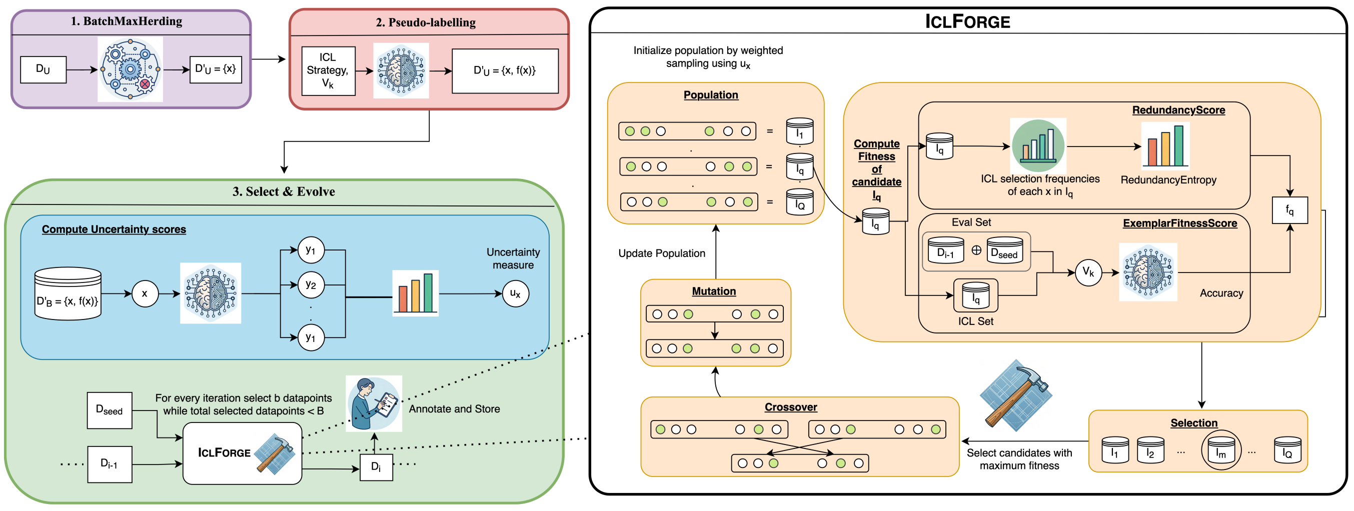

**Figure 1: Schematic diagram of IclForge. We begin by downsampling the unlabelled data using batched MaxHerding and pseudo-labelling the downsampled data. In Select and Evolve, we compute the uncertainties of the datapoints which are used to initialize the population in IclForge . We iteratively select batches of datapoints, send them for annotation and continue augmenting the ICL pool set until the annotation budget $\mathcal{B}$ is exhausted.**

to maximize their collective utility as ICL exemplars. The genetic algorithm specifically optimizes for both relevance and diversity through a carefully designed fitness function. Through a thorough and extensive set of experiments, we demonstrate that IclForge can create high-quality ICL exemplars, significantly outperforming traditional active learning approaches by +180-450 bps, across various tasks and LLMs. Notably, IclForge requires 50% fewer annotations as compared to the strongest baseline. Our work is complementary to existing ICL selection strategies - while selection techniques focus on choosing the most relevant examples from a pool of labeled data, IclForge addresses the fundamental challenge in creating this high-quality pool when labeled data is scarce. Further, our approach requires minor modifications to extend to generative tasks demonstrated in our experiments on MWP tasks.

## 2 Related Work

Recent advances in large language models (LLMs) based on Transformer architectures—such as GPT4[11] and Claude[3] —have transformed NLP by enabling models with hundreds of billions of parameters to learn deep contextual representations from massive corpora. In-Context Learning (ICL) has emerged as a powerful paradigm in the era of large language models (LLMs), allowing models to learn tasks simply by observing a few input-output pairs ([6],[9]). Broadly, ICL operates in two modes: online- per-query retrieval and offline- task-wide selection before inference. Several works have explored textual and similarity-based retrieval methods using models like BERT/sentenceBERT ([19],[13]) or BM25 ([2],[28]). Others have trained retrievers, such as [16],[34], which learns to rank demonstrations using LLM-generated scores, or [29],which applies reinforcement learning with rewards based on confidence and accuracy. Some works have explored approaches for ranking and selecting ICL examples, including information gain-based scoring [17], influence functions for measuring example impact on test set performance [25], and diversity-based methods using k-means clustering with validation set refinement [26].

Active Learning (AL) is a well-established area ([8],[31]), widely applied across various NLP tasks such as natural language inference [32], text classification ([21],[30]), etc. Recently, AL has also been explored in the context of In-Context Learning (ICL). For example, [22] perform active ICL by selecting and labeling examples per query from an unlabeled pool, which are then used as demonstrations—directly integrating AL into the ICL framework. Traditional uncertainty-based AL techniques often rely on access to model confidence scores, which are unavailable in black-box settings, making approaches like the one in [23] less applicable, as they use uncertainty derived from output probabilities. Alternatively, diversity-based methods have been proposed, such as graph-based sampling where datapoints are embedded and connected via kNN to form a graph; nodes with the most unselected neighbors are iteratively chosen [33]. Building on supervised approaches,[15] trained a regressor to predict the expected error reduction from adding new examples to a labeled set, by learning from features that combine classifier state parameters and datapoint characteristics across multiple training iterations. Sub-SA [27] formulates data selection as a submodular maximization problem to ensure representativeness and diversity in the selected subset. More aligned with our setup, IDEAL [36] construct a semantic similarity graph and use graph diffusion to propagate influence from initially selected points, iteratively activating neighbors based on edge-weighted stochastic processes. The subset with the highest influence score is then selected for annotation.

## 3 Problem Formulation

Let $\mathcal{T}$ denote a $C$-way classification task defined over an input space $\mathcal{X}$ and label space $\mathcal{Y} = \{1, 2, \ldots, C\}$. We assume black-box access to a pre-trained large language model (LLM) $\phi$, which maps an input sequence of tokens to a distribution over output tokens. In ICL setting, predictions for an unlabeled instance $x \in \mathcal{X}$ are conditioned on a set of $k$ labeled exemplars presented in the prompt.

Motivated from an active learning setup common in industry settings, we begin with a seed set $\mathcal{D}_{\text{seed}} = \{(x_i, y_i)\}_{i=1}^{N_{seed}}$ containing $N_{seed}$ labeled examples. In addition, we are given an unlabeled pool $\mathcal{D}_u = \{x_j\}_{j=1}^{N_u}$ of $N_u$ instances, from which a subset can be annotated to expand the exemplar pool. We assume an annotation budget $\mathcal{B}$, allowing us to label up to $\mathcal{B}$ instances from $\mathcal{D}_u$. Let $\mathcal{D}_{\mathcal{B}} \subseteq \mathcal{D}_u$ denote the selected subset for annotation, where $|\mathcal{D}_{\mathcal{B}}| \leq \mathcal{B}$.

At test time, for each input $x \in \mathcal{D}_{\text{test}}$, the model $\phi$ receives a prompt constructed from $k$ exemplars selected from the combined labeled set $\mathcal{D}_{\text{seed}} \cup \mathcal{D}_{\mathcal{B}}$ using a fixed exemplar selection strategy $\mathcal{V}_k$. The model then produces a predicted label $\hat{y} = \phi(x \mid \mathcal{V}_k(\mathcal{D}_{\text{seed}} \cup \mathcal{D}_{\mathcal{B}}))$.

Our objective is to select an optimal subset $\mathcal{D}^*_{\mathcal{B}}$ that maximizes the expected performance on the downstream task:

$$\mathcal{D}^*_{\mathcal{B}} = \arg \max_{\substack{\mathcal{D}_{\mathcal{B}} \subseteq \mathcal{D}_u \\ |\mathcal{D}_{\mathcal{B}}| \leq \mathcal{B}}} \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{test}}} \left[ \mathcal{M}\left(y, \phi\left(x \mid \mathcal{V}_k(\mathcal{D}_{\text{seed}} \cup \mathcal{D}_{\mathcal{B}})\right)\right)\right]$$

where $\mathcal{M}$ is a task-specific evaluation metric, such as accuracy or F1 score. This formulation assumes that the ICL strategy $\mathcal{V}_k$ is fixed and consistent across selection and evaluation.

## 4 Methodology

We propose an iterative active learning framework IclForge (Fig. 1) which performs a batch-wise selection of influential datapoints for the purpose of optimizing In-Context Learning performance of the LLM. Given our assumption of ample-sized unlabelled data, we first filter the datapoints using batch-wise MaxHerding (BatchMaxHerding) [5]. The crucial property of MaxHerding is that it selects datapoints by optimizing diversity. Using BatchMaxHerding, we downsample the unlabelled data to a few thousand datapoints. Next, we perform the pseudo-labelling of the downsampled data using the same LLM $\phi$ with ICL selection strategy $\mathcal{V}_k$ and the ICL pool set as the small seed set $\mathcal{D}_{seed}$ we initially start with.

**Algorithm 1 Select and Evolve**

```
Require: Annotation budget $\mathcal{B}$, batch size $b$, pseudo-labeled set $\mathcal{D}'_u$, seed set $\mathcal{D}_{seed}$, LLM $\phi$, ICL selection strategy $\mathcal{V}_k$
Ensure: Set $\mathcal{D}_{\mathcal{B}}$ containing $\mathcal{B}$ annotated examples
 1: Initialize: $\mathcal{D}_0 \leftarrow \emptyset$, $\mathcal{D}_{\mathcal{B}} \leftarrow \emptyset$, $i = 1$
 2: $\mathcal{U}_x \leftarrow$ CalculateUncertainties($\phi, \mathcal{D}'_u$)
 3: while $|\mathcal{D}_{\mathcal{B}}| < \mathcal{B}$ do
 4:     $\mathcal{D}_{annotate} \leftarrow$ IclForge($\mathcal{D}'_u, \mathcal{D}_{i-1}, \mathcal{D}_{seed}, \mathcal{U}_x, \phi, \mathcal{V}_k, b, \mathcal{B}$)
 5:     $\mathcal{D}'_u \leftarrow \mathcal{D}'_u - \mathcal{D}_{annotate}$
 6:     $\mathcal{D}_i \leftarrow \mathcal{D}_{annotate} \cup \mathcal{D}_{i-1}$
 7:     $\mathcal{D}_{\mathcal{B}} \leftarrow \mathcal{D}_i$
 8:     $i \leftarrow i + 1$
 9: end while
10: return $\mathcal{D}_{\mathcal{B}}$
```

Once we obtain the pseudo-labelled downsampled dataset, $\mathcal{D}'_u$, we begin the active learning process of Select and Evolve (Fig. 1). In order to support population initialization in IclForge with candidates containing difficult examples, we first compute the uncertainties over the set $\mathcal{D}'_u$ using the LLM $\phi$. Following the existing works [23], we used temperature sampling to estimate uncertainties in black-box predictions of the LLM. Recent works have shown that uncertainty in the prediction acts as an indicator of that datapoint to be a potential hard negative. The process of Select and Evolve iteratively selects batches of datapoints using IclForge until the annotation budget is exhausted. The selected datapoints are removed from the set $\mathcal{D}'_u$ and sent for human annotation. The annotated datapoints are then stored and used for the selection of next batch with IclForge.

**Algorithm 2 IclForge**

```
Require: Remaining pseudo-labeled set $\mathcal{D}'_u$, selections till last iteration $\mathcal{D}_{i-1}$, seed set $\mathcal{D}_{seed}$, uncertainties $\mathcal{U}_x$, LLM $\phi$, ICL selection strategy $\mathcal{V}_k$, batch size $b$, population size $Q$, num steps $T$
Ensure: Set $\mathcal{D}_{annotate}$ containing $b$ human annotated examples
 1: Initialize: $\mathcal{P} \leftarrow \{\mathcal{I}_1, \mathcal{I}_2, ..., \mathcal{I}_Q\}$ with weighted sampling using uncertainty scores $\mathcal{U}_x$ from $\mathcal{D}'_{\mathcal{B}}$, such that $|\mathcal{I}_j| = min(b, \mathcal{B} - |\mathcal{D}_{i-1}|) \forall j \in (1, Q)$, $\mathcal{I}_{optimal} \leftarrow \emptyset$, maxfit $\leftarrow -\infty$
 2: while $T$ do
 3:     for each $\mathcal{I}_j \in \mathcal{P}$ do
 4:         fitness[$\mathcal{I}_j$] $\leftarrow$ EvaluateFitness($\mathcal{I}_j, \mathcal{D}_{seed}, \mathcal{D}_{i-1}, \phi, \mathcal{V}_k$)
 5:     end for
 6:     $\mathcal{I}_{optimal} \leftarrow \arg\max_{\mathcal{I}_j \in \mathcal{P}}$ fitness[$\mathcal{I}_j$]
 7:     maxfit $\leftarrow$ fitness[$\mathcal{I}_{optimal}$]
 8:     $\mathcal{P}_{new} \leftarrow \{\mathcal{I}_{optimal}\}$
 9:     while $|\mathcal{P}_{new}| < Q$ do
10:         Select $\mathcal{I}_\alpha$ and $\mathcal{I}_\beta$ based on tournament selection
11:         Obtain children, $c_\alpha, c_\beta \leftarrow$ Crossover($\mathcal{I}_\alpha, \mathcal{I}_\beta$)
12:         for each child $c \in \{c_\alpha, c_\beta\}$ do
13:             $c' \leftarrow$ Mutation($c$)
14:             $\mathcal{P}_{new} \leftarrow \mathcal{P}_{new} \cup \{c'\}$
15:         end for
16:     end while
17:     $\mathcal{P} \leftarrow \mathcal{P}_{new}$
18:     $T \leftarrow T - 1$
19: end while
20: $\mathcal{D}_{annotate} \leftarrow$ SendForAnnotation($\mathcal{I}_{optimal}$)
21: return $\mathcal{D}_{annotate}$
```

Our proposed active learning algorithm IclForge fundamentally relies on genetic algorithm (GA) to perform optimal selection of a batch of datapoints from the remaining set of datapoints $\mathcal{D}'_u$[1]. Refer to Algorithm 2 for the detailed steps involved. Like a standard GA, we begin by initializing a population of candidates. Each candidate is a collection of $b$ datapoints sampled without replacement from the set $\mathcal{D}'_u$ weighted by their pre-computed uncertainty scores. For each candidate $\mathcal{I}_q$, we compute its fitness score as a combination of *ExemplarFitnessScore* and *RedundancyScore*. **ExemplarFitnessScore** measures the fitness of the candidate to act as a

[1]In every iteration of IclForge, we select a batch of examples from the unlabelled data and send for human annotation. Since those datapoints have already been selected, we remove those from the unlabelled set.

| Model | AL strategy | AgNews | Trec | DBPedia | Intent |
|---|---|---|---|---|---|
| Phi4 | _ | 0.737 (0.004) | 0.6876 (0.003) | 0.8906 (0.003) | 0.0 (0.006) |
| | Complete | 0.8624 (0.007) | 0.8444 (0.011) | 0.9748 (0.006) | +0.2343 (0.008) |
| | Random | 0.789 (0.005) | 0.761 (0.017) | 0.9518 (0.002) | +0.2287 (0.015) |
| | MaxHerding | 0.7968 (0.009) | 0.776 (0.007) | 0.956 (0.012) | +0.2087 (0.015) |
| | Ideal | 0.7921 (0.003) | 0.776 (0.007) | 0.9587 (0.003) | +0.2192 (0.038) |
| | IclForge | **0.8156 (0.006)** | **0.8008 (0.008)** | **0.9702 (0.004)** | **+0.2309 (0.005)** |
| Mistral-7B | _ | 0.8451 (0.005) | 0.737 (0.009) | 0.9369 (0.006) | +0.054 (0.017) |
| | Complete | 0.8862 (0.007) | 0.891 (0.012) | 0.9752 (0.005) | +0.3325 (0.022) |
| | Random | 0.8611 (0.004) | 0.8164 (0.016) | 0.9522 (0.001) | +0.2382 (0.016) |
| | MaxHerding | 0.87 (0.005) | 0.8297 (0.010) | 0.9508 (0.003) | +0.2544 (0.021) |
| | Ideal | 0.8679 (0.003) | 0.844 (0.006) | 0.9682 (0.005) | **+0.3042 (0.029)** |
| | IclForge | **0.8814 (0.015)** | **0.8672 (0.014)** | **0.9717 (0.009)** | +0.2994 (0.010) |
| Deepseek Qwen 14b | _ | 0.8125 (0.003) | 0.7518 (0.008) | 0.9489 (0.002) | +0.2155 (0.024) |
| | Complete | 0.8982 (0.005) | 0.9219 (0.007) | 0.9879 (0.003) | +0.2898 (0.031) |
| | Random | 0.8704 (0.001) | 0.8621 (0.009) | 0.9729 (0.002) | +0.2752 (0.020) |
| | MaxHerding | 0.8789 (0.006) | 0.8722 (0.007) | 0.9711 (0.003) | +0.274 (0.036) |
| | Ideal | 0.8708 (0.006) | 0.8815 (0.010) | **0.9814 (0.001)** | **+0.2889 (0.019)** |
| | IclForge | **0.8869 (0.007)** | **0.8972 (0.008)** | 0.9802 (0.002) | +0.2875 (0.008) |
| Claude Haiku | _ | 0.8169 (0.002) | 0.8146 (0.007) | 0.9592 (0.002) | +0.2191 (0.012) |
| | Complete | 0.8888 (0.004) | 0.932 (0.009) | 0.9868 (0.001) | +0.2959 (0.009) |
| | Random | 0.8661 (0.005) | 0.8543 (0.019) | 0.9794 (0.001) | +0.294 (0.011) |
| | MaxHerding | 0.8641 (0.006) | 0.8584 (0.009) | 0.978 (0.004) | +0.3108 (0.008) |
| | Ideal | 0.8692 (0.000) | 0.8636 (0.011) | 0.9818 (0.001) | +0.2998 (0.029) |
| | IclForge | **0.8802 (0.003)** | **0.8773 (0.004)** | **0.9833 (0.012)** | **+0.3122 (0.002)** |

**Table 1: Comparison of IclForge with other active learning strategies for a fixed annotation budget of 250. The means and standard deviations (in parentheses) of Macro F1 are computed across 3 runs. For Intent, we provide relative improvements over Phi4 performance in absence of any annotation budget.**

set of ICL examples for the ICL selection strategy $\mathcal{V}_k$. Note that, in order to use the candidate as ICL set, we consider pre-computed pseudo-labels as true labels. Using the candidate as ICL pool set, we compute the results on the combination of the initial seed set $\mathcal{D}_{seed}$ and datapoints selected and labelled till now $\mathcal{D}_{i-1}$ as our gold standard test set. This score of the candidate acts as the main objective measure of the impact of selecting the candidate to be used as ICL pool. We hypothesize that instances where only a few certain examples are getting selected as ICL with strategy $\mathcal{V}_k$ are suboptimal candidates. Therefore, when computing the ExemplarFitnessScore, we also measure the frequency of how many times the datapoint from the candidate set was selected in the $k$-shot examples for the strategy $\mathcal{V}_k$. **RedudancyEntropy** is the entropy of the computed frequencies of the datapoints in the candidate. The overall fitness of the candidate is computed as the combination of its ExemplarFitnessScore and its RedundancyEntropy. Due to the combinatorial optimization nature of the problem, we leverage GA to greedily search for candidates with maximum overall fitness. The remaining components of crossover and mutation in GA are kept standard.

## 5 Experiments and Results

In this section, we evaluate our method (IclForge) on multiple tasks. We introduce our experiment setup in Section 5.1 and discuss results in Section 5.2. We further analyze the different components in IclForge and extend it to MWP tasks in Section 6.

| Dataset | Task Type | Num Classes | Test Set Size |
|---|---|---|---|
| AgNews | Classification | 4 | 2000 |
| DBPedia | Classification | 14 | 2000 |
| Trec | Classification | 6 | 500 |
| Intent | Classification | 9 | 203 |
| GSM8k | MWP | _ | 1303 |
| ASDiv | MWP | _ | 408 |

**Table 2: Dataset Statistics**

### 5.1 Experimental Setups

*Datasets and Tasks:* Following existing works on In-Context Learning optimization, we employ 4 different multi-class classification tasks from different domains. For each dataset we use the train, validation and test splits from the Transformers library. During the final evaluation after the ICL pool has been augmented with active learning, we provide the results on the test split of each dataset. Specifically, we use the AgNews [37], Trec [18] and DBPedia [4] datasets from the public domain. For classification tasks having small labelspace, we choose AgNews which is a 4-way multi class classification task for classifying news headlines into

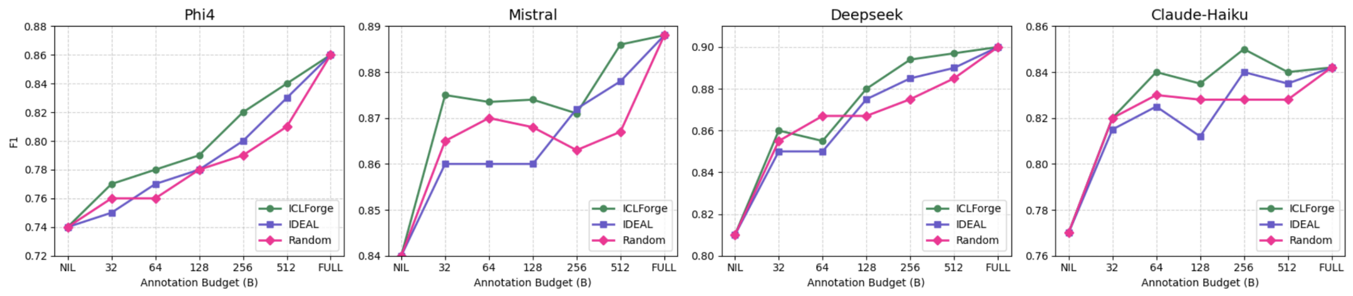

**Figure 2: Effect of scaling the annotation budget for (1) IclForge , (2) Ideal and (3) Random . on AgNews . NIL and FULL on the x-axis denote zero and unlimited annotation budget respectively.**

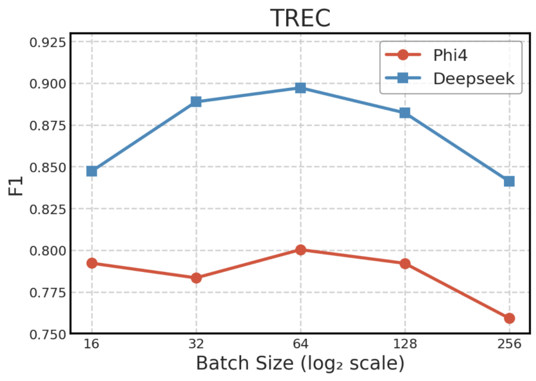

**Figure 3: Effect of scaling batch size in IclForge**

categories like 'World', 'Business' etc. Similarly, we choose Trec which is a 6-way question classification task. For datasets with bigger label space, we select DBPedia as a 14-way ontology classification task. We also benchmark our approach on an internal intent classification dataset (Intent) of chat utterances having 9-classes. Refer to dataset statistics in Table 2. We use Macro F1 as the standard multi-class classification metric for these tasks. To study the generalizability of our approach we also benchmark performance on MWP benchmarks of GSM8k [7] and ASDiv [24] using EM% as the evaluation metric. We provide some example prompts in Appendix A.3; however, due to space limitations, we could not include prompts for all datasets.

*Models:* To understand the efficacy of IclForge , we choose a collection of models from different model families and sizes. Under the umbrella of open-source models we choose Phi4 [1], Deepseek-Qwen-14B [10] and Mistral-7B [14]. From the list of proprietary models, we choose Claude-Haiku [3]. We provide the exact bedrock versions and huggingface model id in A.1.

*Baselines:* A simple approach for making selection of datapoints from a pool of unlabelled set is to just perform a random selection of $\mathcal{B}$ datapoints. We call this strategy as Random . For comparison against IclForge , we consider two active learning approaches from the literature. MaxHerding [5] is an active learning method of selecting datapoints by focusing on maximizing the diversity of the selected datapoints. Ideal [36] is an active learning strategy which is specifically tailored for selecting unlabelled datapoints and augmenting the ICL pool set for the purpose of optimizing the LLM's few-shot performance.

Note that in our results we use the ICL selection strategy, $\mathcal{V}_k$ fixed as the pre-trained retriever (unless specified otherwise). We use the MiniLM [35] model to compute the embeddings and perform the exemplar retrieval. In addition, the number of shots $k$ are fixed as 15.

## 5.2 Main Results

We provide the comparison of IclForge with other active learning strategies in Table 1 (refer to Appendix A.2 for hyperparameter details) for an annotation budget of 250 datapoints. For reference, we also compute results on two more setups where we assume annotation budget is zero (denoted as active learning strategy "_") or unlimited (Complete ). These provide an approximate lower and upper bound of the performance for any active learning approach. We can observe that IclForge is consistently the best performing active learning strategy or is comparable to Ideal across all models and datasets. For Trec , IclForge is consistently outperforming Ideal by 250 bps. Specifically for Mistral, we can see that the IclForge curated ICL set is close to the Complete performance across datasets, illustrating the effectiveness of our approach.

## 6 Analysis

In this section, we critically analyze the behavior of IclForge and conduct studies from various angles to understand the efficacy and robustness of our strategy. In Section 6.1, we study the effect of varying batch size, $b$ for fixed annotation budget $\mathcal{B}$ in IclForge . We also study the effect of scaling annotation budget on our method in Section 6.2. The selected batch of datapoints from IclForge is conditioned on a specific LLM and a specific ICL selection strategy. In Section 6.3, we study the consistency of IclForge across different ICL selection strategies. In Section 6.4, we try to understand the quality and diversity of the selected datapoints across IclForge iterations. Lastly, we show the generalizability of IclForge to other tasks in Sections 6.5.

## 6.1 Understanding effect of batch size in IclForge

In IclForge , batch size acts as a crucial parameter since the fitnesses are computed over the combination of seed set $\mathcal{D}_{seed}$ and

| ICL Selection Strategy, $\mathcal{V}_k$ | AgNews | Trec | DBPedia | Intent |
|---|---|---|---|---|
| Clustering | 0.7953 (0.00) | 0.7709 (0.01) | 0.9402 (0.00) | +0.2192 (0.01) |
| Label Balance | 0.7912 (0.01) | 0.7698 (0.00) | 0.9354 (0.01) | +0.2057 (0.01) |
| Pre-trained Retriever, MiniLM | 0.8123 (0.01) | 0.7903 (0.01) | 0.9423 (0.02) | +0.2226 (0.01) |
| Pre-trained Retriever, UDR | **0.8287 (0.02)** | **0.8031 (0.01)** | **0.9539 (0.01)** | **+0.2343 (0.01)** |

**Table 3: Effect on IclForge by varying ICL selection strategy across classification tasks for Phi4 . As one might expect, the performance of IclForge improves with selecting a better ICL selection strategy for the task.**

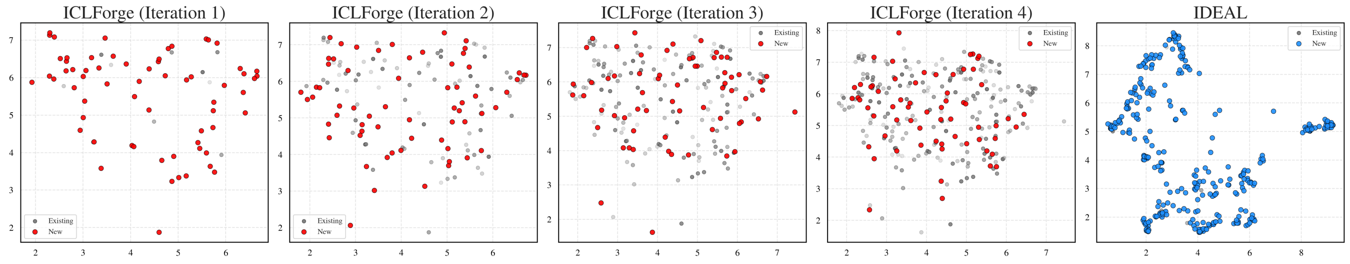

**Figure 4: UMAP representations of datapoints selected with IclForge vs Ideal . Every iteration, IclForge selects datapoints (Red) and adds them to the existing set of datapoints (Grey). We can observe that for Ideal , the selected set of datapoints (Blue), appear to be more clustered with each other.**

the examples annotated so far. If batch size is the same as total annotation budget, there is only a single iteration of IclForge to select all $\mathcal{B}$ examples. This is sub-optimal because the size of the seed set is very limited. Having a smaller batch size iteratively keeps augmenting the fitness evaluation set, giving better estimates of candidate fitnesses as iterations progress. To study the effect of batch size, we fix an annotation budget $\mathcal{B}$ as 256 and vary the batch sizes to obtain different ICL pool sets using IclForge . In Figure 3, we scale the batch size from 16 to 256 and observe that both very small and huge batch sizes adversely affect the performance of our method. Given the empirical results in the figure, we select the batch size to be 64.

## 6.2 Scaling the annotation budget $\mathcal{B}$ in IclForge

In another parallel study to probe IclForge , we scale the annotation budget. From Section 6.1, we noted that for annotation budget of 256, batch size 64 works well with IclForge , therefore, we set the batch size to be $\mathcal{B}/4$. We execute a single run of IclForge with annotation budget $\mathcal{B}$ in $[32, 64, 128, 256, 512]$. We plot the performance of IclForge on AgNews across different models for 3 active learning strategies. Interestingly, we note that Ideal underperforms when annotation budget is less than 64. IclForge consistently outperforms both Ideal and Random for Phi4 for any annotation budget, except for $\mathcal{B} = 256$ in case of Mistral-7B and for $\mathcal{B} = 64$ in Deepseek-Qwen-14B . Additionally, we note that in several places Ideal requires $2-4$ times more annotations for comparable performance with IclForge .

## 6.3 Evaluation on different ICL selection strategies, $\mathcal{V}_k$

IclForge involves the use of strategy $\mathcal{V}_k$ to obtain the fitness scores. In this study, we try to answer the question of whether the performance of IclForge improves with better ICL selection strategy. We benchmark the performance of IclForge with different $\mathcal{V}_k$ during both selection stage and evaluation stage. Specifically, we choose 4 ICL selection strategies of Clustering (computing $k$ clusters from the ICL set and choosing cluster means as the ICL examples), Label Balance (using label-stratified $k$ examples) and pre-trained retriever (online $k$-shot example retrieval with either MiniLM or Unified Demonstration Retriever, UDR [16]). We compute the results of IclForge with these selection strategies for Phi4 with $\mathcal{B} = 256$ in Table 3. From [16], we know that online ICL selection with UDR performs best on the downstream task, and we observe a similar pattern in the results in Table 3. As we improve the ICL selection strategy, the performance of IclForge automatically improves.

## 6.4 Diversity of datapoints selected by IclForge

To answer the question of: Why are datapoints selected using IclForge construct a better ICL pool set?, we aim to measure the quality of the selected datapoints through the lens of diversity. One metric of measuring the quality is the evaluation results post selection, however, the evaluation results appear as a black box. To understand the quality of the selected datapoints in a more deeper way, we plot the UMAP representations of the datapoints across iterations to confirm whether we are indeed covering a broader and more representative part of the data manifold. We conduct an experiment on the Trec dataset with Phi4 and plot their UMAP representations. In Figure 4, we can see that IclForge picks up a

| Model | AL Strategy | GSM8k | ASDiv |
|---|---|---|---|
| Phi4 | – | 0.3894 | 0.8064 |
| | Complete | 0.4289 | 0.8529 |
| | Random | 0.4167 | 0.826 |
| | Ideal | 0.4029 | 0.8578 |
| | IclForge | **0.4261** | **0.8662** |
| Deepseek-Qwen-14B | – | 0.8749 | 0.8922 |
| | Complete | 0.8856 | 0.902 |
| | Random | 0.8836 | 0.9118 |
| | Ideal | 0.8895 | **0.9190** |
| | IclForge | **0.8977** | 0.9173 |

**Table 4: Exact Match% comparison of AL strategies on MWP benchmarks of GSM8k and ASDiv .**

diverse selection of datapoints across the iterations, whereas, Ideal selects datapoints that appear to be more clustered. This re-affirms the observations in previous works that there is indeed a correlation between diversity of selected datapoints as ICL set and their contribution towards an improved performance on the task.

## 6.5 Extention to other tasks

To assess the generalizability of IclForge beyond classification tasks, we extended our evaluation to mathematical word problem (MWP) solving, which represents a more complex generative task. We conducted experiments on two standard MWP benchmarks: GSM8k and ASDiv . For these tasks, we employed the same active learning strategies as in our classification experiments but adapted the evaluation metric to Exact Match (EM%), which measures whether the model's final numerical answer exactly matches the ground truth. Notably, for these MWP tasks, we incorporated chain-of-thought reasoning in both the input and exemplars. To adapt IclForge for these generative tasks, we implemented two key modifications: (1) the fitness function now measures EM% instead of accuracy, and (2) we incorporated reasoning chains alongside input queries when encoding for retrieval with strategy $\mathcal{V}_k$.

Table 4 demonstrates that the ICL set constructed by ICLForge outperforms both Random and Ideal selection strategies on both benchmarks when used as an ICL pool set. Interestingly, in some cases, the ICL sets selected by our method deliver better performance than Complete sets, where we assume an unlimited annotation budget. This suggests that the Complete set does not represent a strict upper bound for exemplar performance, and that active learning approaches can identify smaller, more effective subsets. These experiments further validate the utility of IclForge as a general-purpose method for optimizing in-context learning across diverse NLP applications.

## 7 Conclusion and Future Work

We presented IclForge , a novel active learning framework for creating high-quality in-context learning exemplar set under constrained annotation budget. Our approach identifies the most informative examples for annotation, outperforming traditional active learning methods across multiple classification tasks and LLM architectures. IclForge achieves comparable performance with 50% fewer annotations, offering significant advantages in scenarios with limited labeled data availability. For future work, we plan to experiment with more open-ended generation tasks, extending capabilities of IclForge beyond classification and MWPs. Expectedly, this would require adapting our fitness function to evaluate generation quality and diversity, potentially incorporating metrics like ROUGE, BLEU, etc similar to our changes in 6.5. This could help identify exemplars that demonstrate diverse writing styles, reasoning patterns, or domain-specific knowledge, further expanding the utility of IclForge across a broader range of NLP applications.

## 8 GenAI Usage Disclosure

In preparing this manuscript, we utilized generative AI tools solely for editorial assistance to enhance the quality of existing text. Like the conventional typing assistants, we employed these tools to improve grammar, spelling, punctuation, and overall clarity. The content, ideas, and analysis presented in this paper were developed independently without AI generation. The AI tools served exclusively as editorial aids to refine the presentation of our work.

## A Appendix

### A.1 Models

**Hugging Face Models:**

- Phi4 : microsoft/Phi-4-mini-instruct
- Mistral-7B : mistralai/Mistral-7B-Instruct-v0.3
- Deepseek-Qwen-14B : deepseek-ai/DeepSeek-R1-Distill-Qwen-14B

**Amazon Bedrock Models:**

- Claude-Haiku : anthropic.claude-3-haiku-20240307-v1:0

### A.2 Hyperparameters

For MaxHerding, we experimented with $\gamma$ values of 0.1, 1.0, and 10.0, maintaining a batch size of 1. The Ideal method was implemented with 4 nearest neighbors (K=4). IclForge was configured with a population size of 128, batch size of 50, and executed 24 iterations per batch.

### A.3 Prompts

*A.3.1 AGnews.*

```
Classify the following article into one of:
- Business
- World
- Sports
- Science

Article: {icl_example_input}
Answer: {icl_example_label}
.
.
.
Article: {input}
Answer:
```

*A.3.2 Trec.*

```
Classify the questions based on whether their
answer type is a Number, Location, Person,
Description, Entity, or Abbreviation.

Question: {icl_example_input}
Answer Type: {icl_example_label}
.
.
.
Question: {input}
Answer Type:
```

## References

[1] Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J. Hewett, Mojan Javaheripi, Piero Kauffmann, James R. Lee, Yin Tat Lee, Yuanzhi Li, Weishung Liu, Caio C. T. Mendes, Anh Nguyen, Eric Price, Gustavo de Rosa, Olli Saarikivi, Adil Salim, Shital Shah, Xin Wang, Rachel Ward, Yue Wu, Dingli Yu, Cyril Zhang, and Yi Zhang. 2024. Phi-4 Technical Report. arXiv:2412.08905 [cs.CL] https://arxiv.org/abs/2412.08905

[2] Sweta Agrawal, Chunting Zhou, Mike Lewis, Luke Zettlemoyer, and Marjan Ghazvininejad. 2022. In-context Examples Selection for Machine Translation. arXiv:2212.02437 [cs.CL] https://arxiv.org/abs/2212.02437

[3] Anthropic. 2024. The Claude 3 Model Family: Opus, Sonnet, Haiku. https://www-cdn.anthropic.com/de8ba9b01c9ab7cbabf5c33b80b7bbc618857627/Model_Card_Claude_3.pdf

[4] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. DBpedia: a nucleus for a web of open data. In *Proceedings of the 6th International The Semantic Web and 2nd Asian Conference on Asian Semantic Web Conference* (Busan, Korea) *(ISWC'07/ASWC'07)*. Springer-Verlag, Berlin, Heidelberg, 722–735.

[5] Wonho Bae, Junhyug Noh, and Danica J. Sutherland. 2024. Generalized Coverage fornbsp;More Robust Low-Budget Active Learning. In *Computer Vision – ECCV 2024: 18th European Conference, Milan, Italy, September 29–October 4, 2024, Proceedings, Part LXXXIII* (Milan, Italy). Springer-Verlag, Berlin, Heidelberg, 318–334. doi:10.1007/978-3-031-73010-8_19

[6] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared 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 M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. arXiv:2005.14165 [cs.CL] https://arxiv.org/abs/2005.14165

[7] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training Verifiers to Solve Math Word Problems. *arXiv preprint arXiv:2110.14168* (2021).

[8] D. A. Cohn, Z. Ghahramani, and M. I. Jordan. 1996. Active Learning with Statistical Models. arXiv:cs/9603104 [cs.AI] https://arxiv.org/abs/cs/9603104

[9] Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Jingyuan Ma, Rui Li, Heming Xia, Jingjing Xu, Zhiyong Wu, Tianyu Liu, Baobao Chang, Xu Sun, Lei Li, and Zhifang Sui. 2024. A Survey on In-context Learning. arXiv:2301.00234 [cs.CL] https://arxiv.org/abs/2301.00234

[10] DeepSeek-AI et al. 2025. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. arXiv:2501.12948 [cs.CL] https://arxiv.org/abs/2501.12948

[11] OpenAI et al. 2024. GPT-4 Technical Report. arXiv:2303.08774 [cs.CL] https://arxiv.org/abs/2303.08774

[12] Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making Pre-trained Language Models Better Few-shot Learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli (Eds.). Association for Computational Linguistics, Online, 3816–3830. doi:10.18653/v1/2021.acl-long.295

[13] Shivanshu Gupta, Matt Gardner, and Sameer Singh. 2023. Coverage-based Example Selection for In-Context Learning. arXiv:2305.14907 [cs.CL] https://arxiv.org/abs/2305.14907

[14] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7B. arXiv:2310.06825 [cs.CL] https://arxiv.org/abs/2310.06825

[15] Ksenia Konyushkova, Raphael Sznitman, and Pascal Fua. 2017. Learning Active Learning from Data. arXiv:1703.03365 [cs.LG] https://arxiv.org/abs/1703.03365

[16] Xiaonan Li, Kai Lv, Hang Yan, Tianyang Lin, Wei Zhu, Yuan Ni, Guotong Xie, Xiaoling Wang, and Xipeng Qiu. 2023. Unified Demonstration Retriever for In-Context Learning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (Eds.). Association for Computational Linguistics, Toronto, Canada, 4644–4668. doi:10.18653/v1/2023.acl-long.256

[17] Xiaonan Li and Xipeng Qiu. 2023. Finding Support Examples for In-Context Learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, Houda Bouamor, Juan Pino, and Kalika Bali (Eds.). Association for Computational Linguistics, Singapore, 6219–6235. doi:10.18653/v1/2023.findings-emnlp.411

[18] Xin Li and Dan Roth. 2002. Learning Question Classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*. https://aclanthology.org/C02-1150/

[19] Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2021. What Makes Good In-Context Examples for GPT-3? arXiv:2101.06804 [cs.CL] https://arxiv.org/abs/2101.06804

[20] Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically Ordered Prompts and Where to Find Them: Overcoming Few-Shot Prompt Order Sensitivity. arXiv:2104.08786 [cs.CL] https://arxiv.org/abs/2104.08786

[21] Katerina Margatina, Loïc Barrault, and Nikolaos Aletras. 2022. On the Importance of Effectively Adapting Pretrained Language Models for Active Learning. arXiv:2104.08320 [cs.CL] https://arxiv.org/abs/2104.08320

[22] Katerina Margatina, Timo Schick, Nikolaos Aletras, and Jane Dwivedi-Yu. 2023. Active Learning Principles for In-Context Learning with Large Language Models. arXiv:2305.14264 [cs.CL] https://arxiv.org/abs/2305.14264

[23] Costas Mavromatis, Balasubramaniam Srinivasan, Zhengyuan Shen, Jiani Zhang, Huzefa Rangwala, Christos Faloutsos, and George Karypis. 2023. Which Examples to Annotate for In-Context Learning? Towards Effective and Efficient Selection. arXiv:2310.20046 [cs.CL] https://arxiv.org/abs/2310.20046

[24] Shen-yun Miao, Chao-Chun Liang, and Keh-Yih Su. 2020. A Diverse Corpus for Evaluating and Developing English Math Word Problem Solvers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 975–984.

[25] Tai Nguyen and Eric Wong. 2023. In-context Example Selection with Influences. arXiv:2302.11042 [cs.CL] https://arxiv.org/abs/2302.11042

[26] Kiran Purohit, Venktesh V, Raghuram Devalla, Krishna Mohan Yerragorla, Sourangshu Bhattacharya, and Avishek Anand. 2024. EXPLORA: Efficient Exemplar Subset Selection for Complex Reasoning. arXiv:2411.03877 [cs.LG] https://arxiv.org/abs/2411.03877

[27] Jian Qian, Miao Sun, Sifan Zhou, Ziyu Zhao, Ruizhi Hun, and Patrick Chiang. 2024. Sub-SA: Strengthen In-context Learning via Submodular Selective Annotation. arXiv:2407.05693 [cs.LG] https://arxiv.org/abs/2407.05693

[28] Stephen Robertson and Hugo Zaragoza. 2009. The Probabilistic Relevance Framework: BM25 and Beyond. *Found. Trends Inf. Retr.* 3, 4 (April 2009), 333–389. doi:10.1561/1500000019

[29] Alexander Scarlatos and Andrew Lan. 2024. RetICL: Sequential Retrieval of In-Context Examples with Reinforcement Learning. arXiv:2305.14502 [cs.CL] https://arxiv.org/abs/2305.14502

[30] Christopher Schröder, Lydia Müller, Andreas Niekler, and Martin Potthast. 2023. Small-Text: Active Learning for Text Classification in Python. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*. Association for Computational Linguistics, 84–95. doi:10.18653/v1/2023.eacl-demo.11

[31] Burr Settles. 2009. Active Learning Literature Survey. https://api.semanticscholar.org/CorpusID:324600

[32] Ard Snijders, Douwe Kiela, and Katerina Margatina. 2023. Investigating Multi-source Active Learning for Natural Language Inference. arXiv:2302.06976 [cs.CL] https://arxiv.org/abs/2302.06976

[33] Hongjin Su, Jungo Kasai, Chen Henry Wu, Weijia Shi, Tianlu Wang, Jiayi Xin, Rui Zhang, Mari Ostendorf, Luke Zettlemoyer, Noah A. Smith, and Tao Yu. 2022. Selective Annotation Makes Language Models Better Few-Shot Learners. arXiv:2209.01975 [cs.CL] https://arxiv.org/abs/2209.01975

[34] Liang Wang, Nan Yang, and Furu Wei. 2024. Learning to Retrieve In-Context Examples for Large Language Models. arXiv:2307.07164 [cs.CL] https://arxiv.org/abs/2307.07164

[35] Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. MINILM: deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Proceedings of the 34th International Conference on Neural Information Processing Systems* (Vancouver, BC, Canada) *(NIPS '20)*. Curran Associates Inc., Red Hook, NY, USA, Article 485, 13 pages.

[36] Shaokun Zhang, Xiaobo Xia, Zhaoqing Wang, Ling-Hao Chen, Jiale Liu, Qingyun Wu, and Tongliang Liu. 2024. IDEAL: Influence-Driven Selective Annotations Empower In-Context Learners in Large Language Models. arXiv:2310.10873 [cs.CL] https://arxiv.org/abs/2310.10873

[37] Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Proceedings of the 29th International Conference on Neural Information Processing Systems - Volume 1* (Montreal, Canada) *(NIPS'15)*. MIT Press, Cambridge, MA, USA, 649–657.

[38] Tony Z. Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate Before Use: Improving Few-Shot Performance of Language Models. arXiv:2102.09690 [cs.CL] https://arxiv.org/abs/2102.09690

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009