CoverICL: Selective Annotation for In-Context Learning via Active Graph Coverage

Anonymous ACL submission

Abstract

In-context learning (ICL) adapts Large Language Models (LLMs) to new tasks, without requiring any parameter updates, but few annotated examples as input. In this work, we investigate selective annotation for ICL, where there is a limited budget for annotating examples, similar to low-budget active learning (AL). Although uncertainty-based selection is unreliable with few annotated data, we present COVERICL, an adaptive graph-based selection algorithm, that effectively incorporates uncertainty sampling into selective annotation for ICL. First, COVERICL builds a nearest-neighbor graph based on the semantic similarity between candidate ICL examples. Then, COVERICL employs uncertainty estimation by the LLM to identify hard examples for the task. Selective annotation is performed over the *active graph* of the hard examples, adapting the process to the particular LLM used and the task tackled. COVERICL selects the most representative examples by solving a Maximum Coverage problem, approximating diversity-based sampling. Extensive experiments on nine datasets and six LLMs show that, by incorporating uncertainty via coverage on the active graph, COVERICL (1) outperforms existing AL methods for ICL by 2–4.6% accuracy points, (2) is up to 2× more budget-efficient than SOTA methods for low-budget AL, and (3) generalizes better across tasks compared to non-graph alternatives.

1 Introduction

Large Language Models (LLMs) have shown remarkable performance in various natural language tasks. One of the LLMs' advantages is their ability to perform few-shot learning (Brown et al., 2020), where they can adapt to new tasks, e.g., topic classification or sentiment prediction, via in-context learning (ICL). ICL uses few-shot labeled examples in the form `(input, label)`, e.g., `(“Amazing movie!”, positive)`, to construct a prompt P.

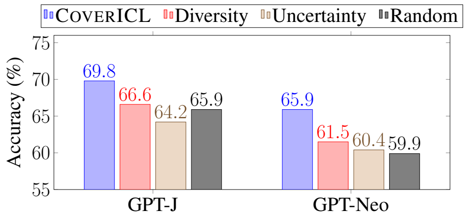


Figure 1: COVERICL effectively combines diversity and uncertainty sampling for low-budgeted ICL, outperforming their counterparts. Results are averaged over seven tasks for GPT-J (6B) and GPT-Neo (1.3B) models with budget $B = 20$ and 5-shot ICL inference.

Prompt P is used as a new input to the LLM, e.g., “`Amazing movie!: positive \n Awful acting: negative \n Terrible movie:`”, before making predictions for the query (“`Terrible movie`”, `?`). The new input enables the LLM to infer the missing label by conditioning the generation on the few-shot examples.

ICL is efficient as it does not require any parameter updates or fine-tuning, wherein users can leverage ICL to generate task-adaptive responses from black-box LLMs. However, ICL is sensitive to the input prompt (Lu et al., 2022) as careful prompt engineering and ground-truth labeling are crucial for good ICL performance (Yoo et al., 2022). Ground-truth labeling requires expert annotators and can be costly, especially for tasks in which the annotators need to provide elaborate responses (Wei et al., 2022). Apart from lowering the labeling cost, carefully reducing the number of the ICL examples can benefit inference costs and the LLM's input context length requirements. Consequently, we study the following active learning (AL) problem: *Given a budget B, which examples do we select to annotate and include in the prompt of ICL?*

Selecting examples via semantic diversity (Zhang et al., 2023) offers better generalization while uncertainty sampling (Lewis and Gale, 1994) captures how well the LLM understands the task. However, in ICL, the LLM is not fine-tuned and the annotated data are used as few-shot input examples. Having few labeled examples at inference (as in few-shot ICL) results in a low-budget AL setting. It has been shown (Zhu et al., 2019; Hacohen et al., 2022; Yehuda et al., 2022; Rittler and Chaudhuri, 2023) that semantic diversity is is crucial in the low-budget AL as uncertainty estimation with few annotated data is unreliable. As a result, current selective annotation methods for ICL rely on semantic diversity (Zhang et al., 2023; Su et al., 2023; Zhang et al., 2024).

To effectively utilize uncertainty sampling for ICL, we propose an adaptive graph-based algorithm, termed CoverICL (Section 4). Motivated by recent theoretical works (Han et al., 2023; Bai et al., 2023) that relate ICL with nearest-neighbor classifiers, CoverICL builds a nearest-neighbor graph that captures the semantic similarities between candidate examples. Then, CoverICL identifies the examples that the LLM is uncertain about (*hard examples*) and creates the *active* subgraph, which consists of the hard examples of interest. The active graph is task and model-aware, as uncertainty estimation depends on the LLM used and how well it understands the task. Having the active graph, CoverICL performs diversity-based sampling by formulating the well-studied Maximum Coverage problem (MaxCover) over the graph. MaxCover selects the examples that best represent the task's difficulty, captures interactions between hard examples, and can be approximately solved via greedy algorithms. Furthermore, (i) we extend CoverICL to an iterative approach that gradually selects harder examples, (ii) we prove that CoverICL approximates diversity sampling, and (iii) we propose a heuristic rule to initialize CoverICL's hyperparameters.

We conduct experiments on nine datasets across five NLP tasks (topic classification, sentiment analysis, natural language inference, summarization, and math reasoning) with six LLMs of varying sizes (1.3B to 65B parameters). As shown in Figure 1, CoverICL boosts ICL performance, improving performance by up to 4.4% accuracy points over diversity and uncertainty sampling. Our **key contributions** are the following:

- CoverICL incorporates the LLM's uncertainty by constructing the active graph of hard examples. The most representative and diverse examples are selected via MaxCover to be annotated for ICL.
- CoverICL is extended to an iterative approach that gradually selects harder examples (CoverICL+). Moreover, CoverICL has theoretical guarantees that it approximates diversity sampling, while CoverICL's hyperparameters can be determined via a heuristic rule.
- CoverICL outperforms competing ICL methods for selective annotation by up to 4.4% points. By incorporating uncertainty via the active graph, CoverICL is up to 2× more budget-efficient than SOTA methods for low-budget AL.

2 Related Work

Active Learning for NLP. Active learning (Settles, 2009) for NLP has been well-studied (Zhang et al., 2022b) with applications to text classification (Schröder and Niekler, 2020), machine translation (Haffari et al., 2009), and name entity recognition Erdmann et al. (2019), among others. Ein-Dor et al. (2020) studied the application of traditional active learning techniques (Lewis and Gale, 1994; Sener and Savarese, 2018) for BERT pretrained models (Devlin et al., 2019), with many works following up (Margatina et al., 2021; Schröder et al., 2022) and (Yu et al., 2022, 2023). These approaches fine-tune the model during different active learning rounds, which allows the model to incorporate information from the newly labeled examples into its parameters to gradually improve its predictions. However, LLMs with billions of parameters are used for ICL. In this case, computing gradient updates is costly and requires additional fine-tuning for every new task. Furthermore, ICL acts as a nonparametric kernel regression (Han et al., 2023; Bai et al., 2023). Designing active learning for non-parametric classifiers has been recently highlighted to be challenging (Rittler and Chaudhuri, 2023), as the assumption that new information is incorporated into the model's parameters does not hold.

Selective Annotation for ICL. In this work, we focus on the low-budget setting, similar to (Su et al., 2023; Zhang et al., 2024), where we are given an unlabeled set to select examples from. As there are no to few annotated examples, it is challenging for the LLM to understand the ICL task. Most of the current approaches of annotating new examples for ICL (Zhang et al., 2022a; Li and Qiu, 2023; Nguyen and Wong, 2023; Shum et al., 2023; Ma et al., 2023) assume a high-resource setting,

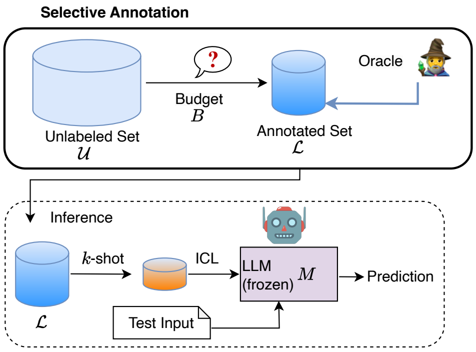


Figure 2: Our studied problem setting: *How to select $\mathcal{L}$ for ICL inference?* Given an unlabeled set $\mathcal{U}$ and a fixed budget B, the goal is to select the B most informative examples for annotation (set $\mathcal{L}$) by an oracle. Examples in $\mathcal{L}$ are used to for k-shot ICL inference with an LLM M.

where a large set of ICL examples is already annotated (validation set). The validation set is leveraged for measuring the informativeness of each individual example as well as for hyperparameter tuning. For example, Zhang et al. (2022a) employ reinforcement learning, which requires one set of labeled examples for policy training and another set of labeled examples for reward estimation. This limits the applicability in practical low-resource scenarios (Perez et al., 2021), where annotations are costly to obtain.

3 Problem Statement & Background

We illustrate the overall problem setting in Figure 2. Given an unlabeled set $\mathcal{U} = \{x_i\}_{i=1}^{N}$ and a fixed budget $B \in \mathbb{Z}^+$, the goal is to select a subset that contains B selected examples. The B selected examples $\{x_i\}_{i=1}^{B}$ are queried to an oracle (i.e., human annotators) for their ground-truth annotations $\{y_i\}_{i=1}^{B}$, forming the annotated set $\mathcal{L} = \{(x_i, y_i)\}_{i=1}^{B}$. During inference with a target LLM M, set $\mathcal{L}$ provides ICL examples to construct a new prompt P for the LLM. Due to context-length limits or inference cost considerations, we consider a k-shot ICL inference, where $k < B$. The k-shot examples are used to construct a new prompt P as input to the LLM by

$$P = \pi(x_1, y_1) \oplus \cdots \oplus \pi(x_k, y_k) \oplus \pi(x_{\text{test}}, *). \quad (1)$$

Template π denotes a natural language verbalization for each demonstration (x, y) and it also expresses how the labels y map to the target tokens.

We elaborate on selective annotation in practice:

1. **Selective annotation** methods identify the B examples to be annotated.
2. **Human experts (oracle)** are employed to annotate the examples; this can be a time-consuming process depending on the task (e.g., math tasks require writing elaborate arithmetic steps).
3. LLMs performs **ICL inference** using the annotated examples. Inference is the same regardless of the selective annotation method used.

Selective Annotation. Selection algorithms differ at the way to choose the examples to be annotated in $\mathcal{L}$. For instance, random selection selects B random examples to be annotated in $\mathcal{L}$, while diversity-based sampling, such as $k$means (MacQueen et al., 1967), select the B most representative examples in the embedding space. Uncertainty-based sampling (Lewis and Gale, 1994) selects B examples the LLM is the most uncertain about to be annotated by the oracle. While uncertainty-based methods require more resources for Step (1) above, it is a one-time cost before human annotation and inference.

Inference. After the B selected ICL examples are annotated by the oracle, inference is the same for all selection algorithms (random, diversity-based, etc.), using the target LLM. To determine which k-shot ICL examples to use for a test instance x_{test}, most approaches (Liu et al., 2021; Rubin et al., 2022; Margatina et al., 2023) employ a k-NN retriever that selects the top-k examples from $\mathcal{L}$, e.g., (x_k, y_k), for x_{test} based on their semantic similarity using models such as SBERT (Reimers and Gurevych, 2019).

3.1 ICL as Low-Budget AL

To understand the impact of the ICL examples on model predictions, we express ICL inference as a non-parametric kernel regression, following the theoretical works from Han et al. (2023); Bai et al. (2023). The prediction for the test instance x_{test} is related to

$$\tilde{y}_{\text{test}} = \frac{\sum_{i=1}^{k} y_i K_{\mathcal{D}}(x_{\text{test}}, x_i)}{\sum_{i=1}^{k} K_{\mathcal{D}}(x_{\text{test}}, x_i)}, \quad (2)$$

where $K_{\mathcal{D}}(x_{\text{test}}, x_i)$ is a kernel that measures the similarity between x_{test} with each of the k-shot retrieved instance x_i, which depends on the pre-training data distribution $\mathcal{D}$.

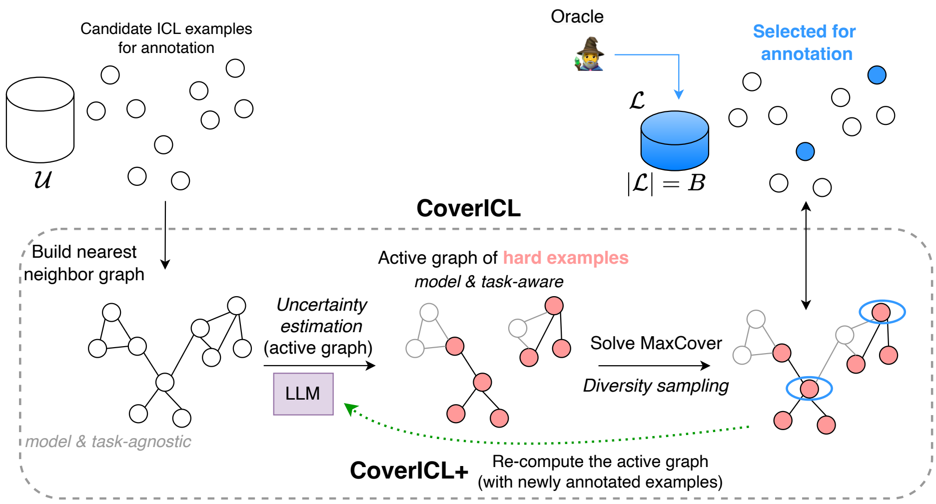


Figure 3: COVERICL algorithm for selective ICL annotation. CoverICL leverages the LLM's uncertainty to construct the active nearest-neighbor graph, which is model and task-aware. Then, COVERICL performs diversity-based sampling over the active graph by solving a MAXCOVER problem. COVERICL+ performs the selection process iteratively, where the LLM's uncertainty is re-estimated.

ICL acts similar to non-parametric kNN classifiers (Equation 2) and designing active learning strategies for such classifiers has been recently highlighted to be challenging (Rittler and Chaudhuri, 2023). New information cannot be directly incorporated into the model's parameters, but can only be provided as few-shot input examples, resulting in a **low-budget AL** setting. It has been shown (Zhu et al., 2019; Hacohen et al., 2022; Yehuda et al., 2022; Rittler and Chaudhuri, 2023) that diversity-based sampling is crucial in the low-budget AL as uncertainty estimation with few annotated data is unreliable.

4 COVERICL: Improving Selective Annotation for ICL

Using the LLM's feedback, e.g., via uncertainty sampling, adapts the selective annotation process to the underlying model and task. However, uncertainty estimation with few annotated examples, as in the low-budget AL setting, is unreliable.

To effectively utilize uncertainty sampling for ICL, we propose an adaptive graph-based algorithm, termed COVERICL. The overall framework is presented in Figure 3. Motivated by recent works that relate ICL with nearest-neighbor classifiers (Section 3.1), COVERICL builds a nearest-neighbor graph that captures the semantic similarities between candidate examples. Then, COVERICL identifies the examples that the LLM is uncertain about (*hard examples*) and creates the *active* subgraph, which consists of the hard examples of interest. The active graph is task and model-aware, as uncertainty estimation depends on the LLM used and how well it understands the task. Having the active graph, COVERICL performs diversity-based sampling by formulating the well-studied Maximum Coverage problem (MAXCOVER). Additionally, our COVERICL+ variant (Section 4.4) seeks to further improve the LLM's uncertainty estimations and predictions via an iterative framework, similar to having multiple AL iterations.

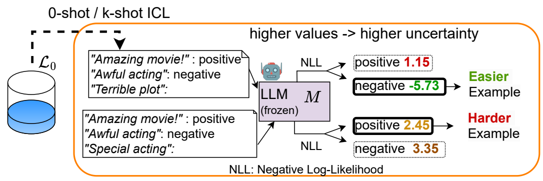


Figure 4: Uncertainty estimation by LLM M with ICL.

4.1 Graph Construction

We build the m-nearest neighbors graph $\mathcal{G}_m$, where the nearest neighbors are determined based on a semantic similarity, e.g., via SBERT embeddings. We compute the embedding of each example x_i and determine its m closest neighbors based on cosine similarity of the embeddings. Graph $\mathcal{G}_m$ does not depend on the LLM used for ICL.

4.2 Active Graph via Uncertainty

Hard Examples. First, we describe how we use uncertainty estimation from the LLM to identify the hard examples, providing an example in Figure 4. We assume we are given an initially small annotated pool $\mathcal{L}_0$ to construct k-shot ICL prompts (if $\mathcal{L}_0 = \emptyset$, it is zero-shot ICL) for each $x_i \in \mathcal{U}$. The k-shot input is given to the LLM along with a query x_i and the LLM makes predictions or generates outputs. Based on the negative loglikehood of the predicted label (for classification tasks) or the average logprobabilites of the generated tokens (for generation tasks), we compute an uncertainty score u_i for each unlabeled example $x_i \in \mathcal{U}$. We sort the examples $x_i \in \mathcal{U}$ based on their uncertainty scores u_i, and mark the top-N_θ out of N total examples as hard examples, which are collected in $\mathcal{U}_h$. Here, $N_\theta = \lfloor \theta N \rfloor$ and $\theta \in [0, 1]$ is a hyperparameter with default value $\theta = 0.5$, which denotes the portion of the examples that we consider as hard ones.

Active Graph. We are interested in hard examples for the LLM, which are collected in set $\mathcal{U}_h$, as explained above. For each $x_i \in \mathcal{U}$, we construct its egonet S_i (1-hop or 2-hop neighbors), where we consider edges of $\mathcal{G}_m$ that direct towards x_i from other hard examples $x_j \in \mathcal{U}_h$. This captures the dependence of other hard examples on x_i. As a result, the active graph is the subgraph that consists of hard examples and their semantic dependencies. Because the active graph is constructed via uncertainty, it captures how well the LLM understands the task (model-aware) as well as the task's difficulty (task-aware).

4.3 Selection via Active Graph MaxCover

Having employed uncertainty for the construction of the active graph, CoverICL performs diversity-based sampling over the active graph. CoverICL solves the Maximum Coverage (MaxCover) problem (Khuller et al., 1999) over the constructed graph, which selects the most representative and diverse examples.

Formally, MaxCover takes N sets $\{S_1, \dots, S_N\}$ and a number B as input. Each set includes some examples, e.g., $S_i = \{x_1, x_2, \dots, x_n\}$ and the intersection of two sets is not necessarily empty, while the goal is to select the B most representative sets that include (cover) as many examples as possible. We assume that if an example is marked as covered by another selected set, it conveys little new information to the LLM. Given the hard examples of $\mathcal{U}_h$ and the egonet S_i of each example (Section 4.2), the MaxCover problem is expressed as

$$\text{maximize} \sum_{x_j \in \mathcal{U}_h} c_j, \tag{3}$$

$$\text{where } c_j \in \{0,1\}, s_i \in \{0,1\}, \tag{4}$$

$$\sum_{i=1}^{N} s_i \leq B, \sum_{x_j \in S_i} s_i \geq c_j. \tag{5}$$

Equation 3 performs diversity-based selection by maximizing the coverage of the examples in $\mathcal{U}_h$. The indicator variable $c_j \in \{0, 1\}$ denotes if example x_j is covered or not. Variable s_i denotes if set S_i is selected. Selecting set S_i, i.e., MaxCover marks $s_i = 1$, means that we select example x_i to be annotated in $\mathcal{L}$. Then, all examples in the egonet of x_i are marked as covered, assuming they convey little new information to the model for the task. Equation 5 ensures that we select at most B sets (first part) and covered examples belong to at least one selected set (second part).

Algorithm 1 Greedy approximation for MaxCover.

Input: Examples $\mathcal{U}_h$, Sets $\{S_1, \dots, S_N\}$, Budget B, $\mathcal{L} = \emptyset$.
while B not exhausted **do**
  Pick the set S_i that covers the most uncovered examples in $\mathcal{U}_h$. Example x_i is selected for annotation, $\mathcal{L} = \mathcal{L} \cup \{x_i\}$.
  Mark examples in $\mathcal{U}_h$ of the chosen set S_i as covered.
end while
Output: Return $\mathcal{L}$.

Greedy Solution. The MaxCover problem is known to be NP-hard (Vazirani, 2001). A natural greedy solution for the MaxCover chooses sets according to one rule: at each stage, choose a set that contains the largest number of uncovered elements. This approximation algorithm is summarized in Algorithm 1, and is well-known to approximately solve MaxCover and can be further improved due to its submodularity (Krause and Guestrin, 2005).

4.4 Further Discussions

CoverICL+. CoverICL performs uncertainty-guided diversity sampling over the active graph. Our variant CoverICL+ considers uncertainty estimation more important for the task and encour-

Table 1: Performance comparison (accuracy in %) of different selective annotation methods for ICL. The budget is $B = 20$ and we perform 5-shot ICL inference with GPT-J (6B) and GPT-Neo (1.3B), averaging the results.

Type	**Method**	**Topic Classification**		**Sentiment Analysis**		**Natural Language Inference**			**Avg.**
		AGNews	TREC	SST2	Amazon	RTE	MRPC	MNLI	
Random	Random	64.17	52.01	75.06	80.92	50.58	66.75	40.29	61.40
Uncertainty	*Hardest (Lewis and Gale, 1994)	68.62	42.64	77.08	81.18	54.10	64.30	38.15	60.87
Diversity	Fast-Vote$k$ (Su et al., 2023)	67.96	48.05	74.39	79.48	51.82	66.21	39.19	61.01
	IDEAL (Zhang et al., 2024)	71.78	42.12	75.78	78.01	54.62	64.77	38.47	60.79
	*Vote$k$ (Su et al., 2023)	67.52	49.48	77.07	80.47	52.08	67.77	39.45	61.98
Diversity+Uncertainty	*Patron (Yu et al., 2023)	72.65	46.74	76.69	83.33	54.03	64.12	38.01	62.22
	*Active-Kmeans	72.26	49.44	80.99	83.39	53.25	65.62	39.19	63.45
	*CoverICL (**Ours**)	**73.92**	**53.64**	**81.23**	**84.11**	**55.01**	**68.61**	**41.66**	**65.45**

*Denotes model-aware methods that consider the target LLM.
Full results are provided in Appendix D.1 and show that CoverICL is significantly better (Wilcoxon signed-rank test) than Patron at p-value < 0.05.

ages the LLM to give new predictions when a certain number of hard examples are covered. We introduce a new hyperparameter T, which denotes the desired number of iterations until we exhaust the budget. At each iteration, we select $\lfloor B/T \rfloor$ new examples that are annotated by the oracle and that are used by the LLM to gradually identify harder examples. We present CoverICL+ in detail in Appendix A.1.

Theoretical Analysis. We provide a theoretical analysis that CoverICL approximates diversity-based sampling over a subsampled graph (the active graph). Our theorem and its proof are provided in Appendix A.2. The theorem suggests that CoverICL can approximate diversity-based selection when the most representative examples are well-separated, even when uncertainty sampling is not helpful.

Heuristic Rule. As there is no validation set for hyperparameter tuning, we propose a heuristic rule to automatically adjust the hyperparameter m, that is used to create the m-nn graph $\mathcal{G}_m$. The heuristic rule (see Appendix A.3) takes advantage of the active graph and the minimum number of hard examples that need to be covered. The number of neighbors m is adjusted so that MaxCover covers at least $\hat{N}_\theta$ hard examples, $\hat{N}_\theta < N_\theta$, before we exhaust the budget B. This ensures that the selected examples are representative enough of the hard examples.

5 Experimental Setting

With our experimental analysis, we address the following research questions (RQs):

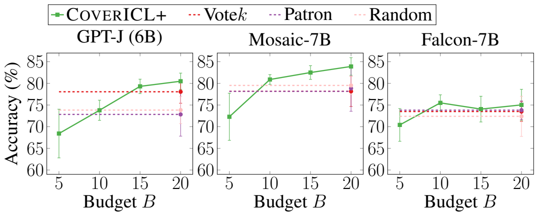
RQ1. How does CoverICL compare with other ICL selective annotation methods across diverse tasks?

RQ2. How effective is CoverICL's active graph coverage for low-budget AL?

RQ3. How sensitive is CoverICL to the graph construction?

Datasets. We perform empirical evaluation with nine NLP datasets that cover well-studied tasks, such as topic classification (Zhang et al., 2015; Hovy et al., 2001), sentiment analysis (Socher et al., 2013; McAuley and Leskovec, 2013), natural language inference (Bentivogli et al., 2009; Dolan et al., 2004; Williams et al., 2018), text summarization (Narayan et al.) and math reasoning (Cobbe et al., 2021). We provide additional details of these datasets in Appendix C.

Competing Methods. All compared methods differ only on the "Selective Annotation" phase (Figure 2), while inference is the same for all (see also Appendix B). We use the following approaches as baselines for comparison: (i) **Random** performs random example selection for annotation. (ii) **Pseudo-labeling** uses the LLM to generate pseudo-labels for the unlabeled instances as additional annotated data. (iii) **IDEAL** (Zhang et al., 2024) is a diversity-based sampling strategy that selects representative examples in the similarity space. (iv) **Vote$k$** (Su et al., 2023) accounts for the model's feedback. It sorts the examples based on the model's confidence scores and stratifies them into B equally-sized buckets. It selects the most representative example from each bucket. (v) **Fast-Vote$k$** (Su et al., 2023) is Vote$k$ but without accounting for the target LLM. (vi) **Hardest** (Lewis and Gale, 1994) resembles the uncertainty sampling strategy, where the examples that the model is the most uncertain about are selected. (vii) **Patron** (Yu et al., 2023) is the SOTA method that combines uncertainty and diversity sampling, but is designed for finetuned-based NLP. Additionally, we include (viii) **Active-Kmeans** method (Appendix A.4) as further ablations, which employs



(a) Average results at AGNews, TREC, SST2, and Amazon datasets with three LLMs of similar size.

	Falcon-40B	LLaMa-65B
	Summarization (RougeL)	
Zero-shot	$18.50_{\pm 0.61}$	$15.26_{\pm 0.69}$
Vote$k$	$20.83_{\pm 0.05}$	$23.38_{\pm 0.84}$
CoverICL	$\mathbf{21.42}_{\pm 0.68}$	$\mathbf{24.67}_{\pm 0.45}$
	Math Reasoning (Accuracy)	
Zero-shot	$36.58_{\pm 3.14}$	$32.54_{\pm 1.86}$
Vote$k$	$37.23_{\pm 1.75}$	$45.04_{\pm 1.47}$
CoverICL	$\mathbf{39.58}_{\pm 3.01}$	$\mathbf{49.08}_{\pm 2.89}$

(b) Generation tasks (XSUM, GSM8K).

Figure 5: Performance comparison across different LLMs and tasks.

Kmeans instead of CoverICL's graph.

Implementation. We experiment with six LLMs of varying sizes (1.3B to 65B parameters), including GPT-J (Wang and Komatsuzaki, 2021), Mosaic (MosaicML, 2023), Falcon (Penedo et al., 2023), and LLaMa (Touvron et al., 2023) models, all of which are open-source and allow the reproducibility of our research. Unless otherwise stated, we set $k = 5$, $B = 20$ and we obtain embeddings for semantic similarity via SBERT (Reimers and Gurevych, 2019). Please refer to Appendix C.2 for more specifics.

Regarding CoverICL's implementation, we construct $m = 5$ nearest-neighbor graphs for CoverICL, and $m = 15$ for CoverICL+. The egonet S_i of each candidate example x_i, which is used as input to the MaxCover problem, includes 1-hop neighbors for CoverICL+ and 2-hop CoverICL. The default number of iterations T for CoverICL+ is $T = 2$. As the threshold hyper-parameter θ, we have $\theta = 0.5$, i.e., 50% of the examples are considered as hard.

6 Results & Studies

6.1 RQ1: CoverICL's Performance across Tasks

Table 1 presents performance results of different selective annotation methods for classification tasks. We include tasks ranging from topic classification, sentiment analysis, and natural language inference. We average the results over two LLMs of 1.3B and 6B sizes. As Table 1 shows, CoverICL is the method that achieves the best performance, with an improvement of 2.00–4.66% accuracy points over competing methods. Methods that give more importance to uncertainty sampling (Patron, Active-Kmeans, CoverICL) perform better on topic classification and sentiment analysis tasks, showing the importance of combining diversity and uncertainty-based selection for ICL. For natural language inference tasks, diversity-based selection is more

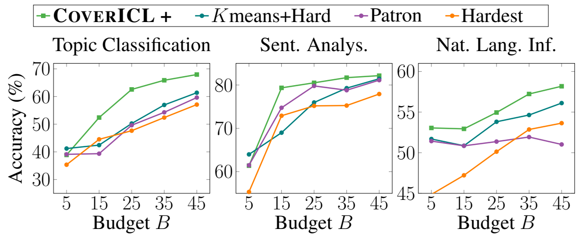


Figure 6: ICL inference results (GPT-Neo) with different uncertainty-based selective annotation methods. CoverICL+ performs the best over all tasks.

important, where methods such as Vote$k$ outperform other uncertainty-based baselines. Overall, CoverICL and Active-Kmeans are the best performing methods, but selection via graph coverage (CoverICL) instead of $k$means (CoverICL) improves accuracy by 0.24–4.20% in all tasks.

Figure 5 compares selective annotation methods across different LLMs and tasks. Figure 5a shows that CoverICL+ generalizes well across different target LLMs. The best performance is achieved for the Mosaic and GPT-J models, where CoverICL+ outperforms Vote$k$ by 4.09% accuracy points, when $B = 20$. In addition, CoverICL+ can considerably reduce the annotation and inference costs. In all cases, CoverICL+ needs only $B = 10$ annotated examples to outperform Patron and Random, which use $B = 20$ annotated examples.

Figure 5b provides results for generation tasks with larger LMs (40B and 65B parameters). On the challenging reasoning tasks, CoverICL outperforms Vote$k$ and zero-shot ICL by 4.04% and 16.54% in accuracy, respectively. Vote$k$ selects examples that are both easy and hard for the model, which do not always provide new information to the model. On the other hand, CoverICL selects representative examples of difficult cases, which help the LLM to better understand the task.

Results with additional LLMs and tasks are provided in Appendix D.2.

Table 2: Performance comparison across different semantic similarity embedding models. Semantic similarity facilitates diversity sampling as well as retrieval-based ICL inference.

Semantic Similarity →	SBERT-all-mpnet-base			RoBERTa-nli-large-mean-tokens			BERT-nli-large-cls-pool			Avg.
	TREC	SST2	Amazon	TREC	SST2	Amazon	TREC	SST2	Amazon	
Pseudo-labeling	$48.56_{\pm 6.33}$	$69.13_{\pm 3.87}$	$70.96_{\pm 3.35}$	$33.98_{\pm 3.68}$	$74.08_{\pm 4.40}$	$81.11_{\pm 4.14}$	$41.27_{\pm 4.24}$	$77.47_{\pm 1.60}$	$81.63_{\pm 2.49}$	64.24
Random	$54.68_{\pm 1.68}$	$68.48_{\pm 1.87}$	$73.95_{\pm 2.03}$	$37.23_{\pm 2.30}$	$74.21_{\pm 3.50}$	$84.46_{\pm 3.21}$	$34.75_{\pm 2.41}$	$72.65_{\pm 5.82}$	$80.20_{\pm 3.34}$	64.51
Vote$k$	$54.81_{\pm 0.49}$	$73.69_{\pm 9.05}$	$75.13_{\pm 0.98}$	$37.77_{\pm 4.65}$	$76.16_{\pm 2.23}$	$84.11_{\pm 1.28}$	$42.43_{\pm 3.34}$	$80.85_{\pm 2.09}$	$83.59_{\pm 1.77}$	67.61
Active-Kmeans	$48.24_{\pm 0.98}$	$77.86_{\pm 1.02}$	$75.77_{\pm 3.63}$	$38.12_{\pm 5.74}$	$78.12_{\pm 5.30}$	$\underline{85.93}_{\pm 2.30}$	$38.15_{\pm 3.10}$	$78.64_{\pm 2.78}$	$\underline{85.80}_{\pm 1.75}$	67.40
COVERICL	$\underline{55.33}_{\pm 2.57}$	$\underline{79.68}_{\pm 2.47}$	$\underline{77.73}_{\pm 2.23}$	$\underline{39.06}_{\pm 3.37}$	$\underline{81.11}_{\pm 1.50}$	$85.15_{\pm 0.55}$	$\underline{44.06}_{\pm 2.49}$	$\underline{80.85}_{\pm 2.83}$	$84.65_{\pm 3.52}$	**69.74**

Table 3: Graph ablation study on hyper-parameter m, which controls the number of graph neighbors, considering 1-hop or 2-hop sets. The values of m are adjusted via our heuristic rule (Appendix A.3).

	AGNews	SST2	Amazon
Vote$k$	$62.77_{\pm 4.82}$	$73.69_{\pm 9.05}$	$75.13_{\pm 0.98}$
COVERICL			
$m = 15$ (1-hop)	$68.61_{\pm 1.02}$	$79.42_{\pm 1.28}$	$77.34_{\pm 2.73}$
*$m = 5$ (2-hop)	$70.95_{\pm 1.87}$	$79.68_{\pm 1.77}$	$77.73_{\pm 2.23}$
COVERICL+ ($T = 2$)			
*$m = 15$ (1-hop)	$69.39_{\pm 1.35}$	$79.03_{\pm 2.47}$	$77.08_{\pm 1.50}$
$m = 5$ (2-hop)	$70.43_{\pm 1.60}$	$77.73_{\pm 1.15}$	$76.43_{\pm 2.55}$

*Denotes the default value.

6.2 RQ2: Active Graph's Impact on Low-Budget AL

In this section, we experiment with different uncertainty-based methods on the low-budget AL. We employ a small GPT-Neo (1.3B) model, which is sensitive to the number of ICL examples annotated. We range the budget size from 5 to 45, incrementing the budget with 10 more annotations for 4 steps. During inference, we use as many ICL annotated examples as the context-length limit of GPT-Neo allows. Figure 6 presents the results. COVERICL+ performs the best in all cases, where the average accuracy improvement over the best baseline is 7.09% for topic classification, 1.86% for sentiment analysis, and 2.36% for natural language inference. It is noteworthy that Active-Kmeans is the best-performing baseline when $B = 45$, showing the benefits of combining diversity and uncertainty-based selection. When the budget is limited, e.g., $B = 15$, COVERICL+ outperforms Active-Kmeans significantly, which shows the benefit of COVERICL's active graph over non-graph baselines, such as $k$means.

6.3 RQ3: Ablation Studies on Graph Sensitivity

In the previous experiments, we use SBERT embedding to calculate semantic similarity between examples during the graph construction. In the following experiment, we use different models for calculating semantic embeddings. Table 2 shows results when we experiment with SBERT, BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) encoders. The target LLM is the GPT-Neo (1.3B) model. Using different encoder models affects the prompt for each test query and thus, ICL performance varies. For instance, SBERT achieves a maximum average performance of 55.33% and 77.73% for TREC and Amazon, respectively, while BERT achieves 44.06% and 85.80%. Despite the encoder choice, COVERICL performs overall the best, outperforming Vote$k$, the second-best method, by 2.13% accuracy points.

Table 3 shows an ablation study on the hyperparameters that control the nearest-neighbor graph construction. We experiment with the values obtained by our proposed heuristic rule (Appendix A.3). As Table 3 shows, different hyperparameter values achieve overall good performance for both COVERICL and COVERICL+. In some cases, there is no performance drop, while COVERICL+ works better with 1-hop egonets.
Further graph ablations are provided in Appendices D.3, D.4.

7 Conclusions

In this work, we investigate selective annotation for ICL and we introduce COVERICL that combines diversity and uncertainty-based selection. Our **key contributions** are highlighted as follows: (1) COVERICL incorporates the LLM's uncertainty by constructing the active graph of hard examples. The most representative and diverse examples are selected via MAXCOVER to be annotated for ICL. (2) COVERICL is extended to an iterative approach that gradually selects harder examples (COVERICL+). Moreover, COVERICL has theoretical guarantees that it approximates diversity sampling, while COVERICL's hyperparameters can be determined via a heuristic rule. (3) COVERICL outperforms competing ICL methods for selective annotation by up to 4.4% points. Incorporating uncertainty via COVERICL's active graph is shown to is up to 2× more budget-efficient than SOTA methods for low-budget AL.

8 CoverICL Limitations

We list some of our assumptions that may limit CoverICL if they are not satisfied. CoverICL relies on embedding methods to determine semantic diversity, similar to many competing methods (except for Random and Hardest). While CoverICL is shown to be robust to different embedding models (Section 6.3), it can still suffer if the semantic space of the test is wildly different from the annotation pool space. Moreover, the graph/set construction is a heuristic approach and does not account for cases where adversarial examples are injected into the pool in order to degrade performance.

References

Yu Bai, Fan Chen, Huan Wang, Caiming Xiong, and Song Mei. 2023. Transformers as statisticians: Provable in-context learning with in-context algorithm selection. *arXiv preprint arXiv:2306.04637*.

Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. 2009. The fifth pascal recognizing textual entailment challenge. *TAC*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *NeurIPS*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Janez Demšar. 2006. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine learning research*, 7:1–30.

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

Bill Dolan, Chris Quirk, and Chris Brockett. 2004. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*.

Liat Ein-Dor, Alon Halfon, Ariel Gera, Eyal Shnarch, Lena Dankin, Leshem Choshen, Marina Danilevsky, Ranit Aharonov, Yoav Katz, and Noam Slonim. 2020. Active Learning for BERT: An Empirical Study. In *EMNLP*.

Alexander Erdmann, David Joseph Wrisley, Benjamin Allen, Christopher Brown, Sophie Cohen-Bodénès, Micha Elsner, Yukun Feng, Brian Joseph, Béatrice Joyeux-Prunel, and Marie-Catherine de Marneffe. 2019. Practical, efficient, and customizable active learning for named entity recognition in the digital humanities. In *EMNLP*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *ACL*.

Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, et al. 2023. Textbooks are all you need. *arXiv preprint arXiv:2306.11644*.

Guy Hacohen, Avihu Dekel, and Daphna Weinshall. 2022. Active learning on a budget: Opposite strategies suit high and low budgets. In *ICML*.

Gholamreza Haffari, Maxim Roy, and Anoop Sarkar. 2009. Active learning for statistical phrase-based machine translation. In *NAACL*.

Chi Han, Ziqi Wang, Han Zhao, and Heng Ji. 2023. In-context learning of large language models explained as kernel regression. *arXiv preprint arXiv:2305.12766*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Eduard Hovy, Laurie Gerber, Ulf Hermjakob, Chin-Yew Lin, and Deepak Ravichandran. 2001. Toward semantics-based answer pinpointing. In *Proceedings of the First International Conference on Human Language Technology Research*.

Samir Khuller, Anna Moss, and Joseph Seffi Naor. 1999. The budgeted maximum coverage problem. *Information processing letters*, 70(1):39–45.

Andreas Krause and Carlos Guestrin. 2005. *A note on the budgeted maximization of submodular functions*. Citeseer.

David D. Lewis and William A. Gale. 1994. A sequential algorithm for training text classifiers. In *SIGIR*.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Šaško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander Rush, and Thomas Wolf. 2021. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*.

Xiaonan Li and Xipeng Qiu. 2023. Finding supporting examples for in-context learning. *arXiv preprint arXiv:2302.13539*.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2021. What makes good in-context examples for gpt-3? *arXiv preprint arXiv:2101.06804*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

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

Huan Ma, Changqing Zhang, Yatao Bian, Lemao Liu, Zhirui Zhang, Peilin Zhao, Shu Zhang, Huazhu Fu, Qinghua Hu, and Bingzhe Wu. 2023. Fairness-guided few-shot prompting for large language models. *arXiv preprint arXiv:2303.13217*.

James MacQueen et al. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA.

Katerina Margatina, Timo Schick, Nikolaos Aletras, and Jane Dwivedi-Yu. 2023. Active learning principles for in-context learning with large language models.

Katerina Margatina, Giorgos Vernikos, Loïc Barrault, and Nikolaos Aletras. 2021. Active learning by acquiring contrastive examples. In *EMNLP*.

Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: Understanding rating dimensions with review text. In *Proceedings of the 7th ACM Conference on Recommender Systems (RecSys)*.

MosaicML. 2023. Introducing mpt-7b: A new standard for open-source, commercially usable llms.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

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

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. 2023. The RefinedWeb dataset for Falcon LLM: outperforming curated corpora with web data, and web data only. *arXiv preprint arXiv:2306.01116*.

Ethan Perez, Douwe Kiela, and Kyunghyun Cho. 2021. True few-shot learning with language models. In *NeurIPS*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *EMNLP*.

Nicholas Rittler and Kamalika Chaudhuri. 2023. A two-stage active learning algorithm for k-nearest neighbors. In *International Conference on Machine Learning*, pages 29103–29129. PMLR.

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2022. Learning to retrieve prompts for in-context learning. In *NAACL*.

Christopher Schröder, Andreas Niekler, and Martin Potthast. 2022. Revisiting uncertainty-based query strategies for active learning with transformers. In *ACL Findings*.

Christopher Schröder and Andreas Niekler. 2020. A survey of active learning for text classification using deep neural networks.

Ozan Sener and Silvio Savarese. 2018. Active learning for convolutional neural networks: A core-set approach. In *ICLR*.

Burr Settles. 2009. Active learning literature survey.

KaShun Shum, Shizhe Diao, and Tong Zhang. 2023. Automatic prompt augmentation and selection with chain-of-thought from labeled data. *arXiv preprint arXiv:2302.12822*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Hongjin Su, Jungo Kasai, Chen Henry Wu, Weijia Shi, Tianlu Wang, Jiayi Xin, Rui Zhang, Mari Ostendorf, Luke Zettlemoyer, Noah A Smith, et al. 2023. Selective annotation makes language models better few-shot learners. In *ICLR*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Vijay V Vazirani. 2001. *Approximation algorithms*, volume 1. Springer.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. `https://github.com/kingoflolz/mesh-transformer-jax`.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *ACL (Demos)*.

Ofer Yehuda, Avihu Dekel, Guy Hacohen, and Daphna Weinshall. 2022. Active learning through a covering lens. *Advances in Neural Information Processing Systems*, 35:22354–22367.

Kang Min Yoo, Junyeob Kim, Hyuhng Joon Kim, Hyunsoo Cho, Hwiyeol Jo, Sang-Woo Lee, Sang-goo Lee, and Taeuk Kim. 2022. Ground-truth labels matter: A deeper look into input-label demonstrations. In *EMNLP*.

Yue Yu, Lingkai Kong, Jieyu Zhang, Rongzhi Zhang, and Chao Zhang. 2022. Actune: Uncertainty-based active self-training for active fine-tuning of pretrained language models. In *NAACL*.

Yue Yu, Rongzhi Zhang, Ran Xu, Jieyu Zhang, Jiaming Shen, and Chao Zhang. 2023. Cold-start data selection for few-shot language model fine-tuning: A prompt-based uncertainty propagation approach. In *ACL*.

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. In *ICLR*.

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

Yiming Zhang, Shi Feng, and Chenhao Tan. 2022a. Active example selection for in-context learning. In *EMNLP*.

Zhisong Zhang, Emma Strubell, and Eduard Hovy. 2022b. A survey of active learning for natural language processing. In *EMNLP*.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. 2023. Automatic chain of thought prompting in large language models. In *ICLR*.

Yu Zhu, Jinghao Lin, Shibi He, Beidou Wang, Ziyu Guan, Haifeng Liu, and Deng Cai. 2019. Addressing the item cold-start problem by attribute-driven active learning. *IEEE Transactions on Knowledge and Data Engineering*, 32(4):631–644.

A CoverICL

Algorithm 2 summarizes the overall CoverICL algorithm. The greedy solution for MaxCover may be terminated when every hard example is covered, regardless of whether the budget B is exhausted. In this case, diversity selection captures the difficulty of the task, and not all hard examples are equally useful. Thus, we add the selected examples to the current annotation set $\mathcal{L}'$, and re-evaluate the model's feedback to define the new hard set $\mathcal{U}'_h$. Algorithm 2 is terminated when the total budget B is exhausted.

A.1 CoverICL+: Iterative Selection

CoverICL performs uncertainty-guided diversity sampling over the active graph. Our variant CoverICL+ considers uncertainty estimation more important for the task and encourages the LLM to give new predictions when a certain number of hard examples are covered. We introduce a new hyperparameter T, which denotes the desired number of iterations until we exhaust the budget. At each iteration, we select $\lfloor B/T \rfloor$ new examples that are annotated by the oracle and that are used by the LLM to gradually identify harder examples. Furthermore, CoverICL+ avoids selecting examples from sets that contain few hard examples, e.g., outliers, or sets that belong to isolated sparse regions by a re-weighting schema for its MaxCover problem. Whenever a hard example is covered, instead of being marked as covered, CoverICL+ reduces its weight.

Dynamically updating the weights of each example does not satisfy the submodularity property of MaxCover, which is satisfied for fixed weights. Nevertheless, such that we can use the greedy algorithm to approximate the optimal solution, we propose a re-weighting trick by reusing $\mathcal{U}_h$ multiple times. Specifically, we copy the set $\mathcal{U}_h$ multiple times, i.e., to $\mathcal{U}_h^0, \mathcal{U}_h^1, \ldots, \mathcal{U}_h^t$, etc., where different sets have different weights for their elements. If hard example x_j^t is covered in $\mathcal{U}_h^t$, then we use its weights from the other sets. Formally, we optimize

$$\text{maximize} \sum_t \sum_{x_j^t \in \mathcal{U}_h^t} w^t c_j^t, \qquad (6)$$

where we set the weights $w^t = 10^{-t}$, so that $w^t \approx w^t + w^{t+1} + \cdots$. In the beginning, every hard example of $\mathcal{U}_h$ has weight $w^0 = 1$. If one example is covered in $\mathcal{U}_h$, i.e., $c_j = 1$, then its new weight is set to $w^1 = 0.1$. The constraint at each iteration for solving Equation 6 is $\sum_{j=1}^{N_\theta} s_j \leq \lfloor B/T \rfloor$. Then, we perform uncertainty estimation with the model M based on the newly annotated examples, before we solve MaxCover with the remaining budget.

Algorithm 2 CoverICL Algorithm.

Input: Model M, Unlabeled Set $\mathcal{U}$, Budget B, Similarity Space $\mathcal{S}$ for k-NN Retriever.
Optional: Initial set $\mathcal{L}_0$, else $\mathcal{L}_0 = \emptyset$.
Hyperparameters: threshold θ, number of neighbors m.
Output: Annotated Set $\mathcal{L}$.

$B_{cur} = 0, \mathcal{L} = \mathcal{L}_0$.
Create global graph $\mathcal{G}_m$.
while $B_{cur} < B$ **do**
  for $x_i \in \mathcal{U}$ **do**
    Retrieve (at most) k examples from $\mathcal{L}$ based on similarity $\mathcal{S}$.
    Use model M to obtain an uncertainty score u_i for x_i with k-shot ICL.
  end for
  Determine hard set $\mathcal{U}_h$ given scores $\{u_i\}_{i=1}^N$ and threshold θ.
  Create sets S_i given $\mathcal{U}_h$ and $\mathcal{G}_m$.
  $\{x_i^*\}_{i=1}^{B^*}$ = Greedy-MaxCover $\Big(\mathcal{U}, \{S_i\}, B - B_{cur}\Big)$.
  Add the selected $\{x_i^*\}_{i=1}^{B^*}$ to $\mathcal{L} = \mathcal{L} \cup \{x_i^*\}_{i=1}^{B^*}$ (querying the oracle) and remove them from $\mathcal{U} = \mathcal{U} \setminus \{x_i^*\}_{i=1}^{B^*}$.
  $B_{cur} = B_{cur} + B^*$.
end while

A.2 Theoretical Analysis

CoverICL constructs a m-nearest neighbor graph $\mathcal{G}_m$. Let $\mathcal{R}_i$ denote the set of neighbors of each node $i \in N$. CoverICL creates sets S_i by excluding the neighbors nodes $v \notin \mathcal{U}_h$ that do not correspond to hard examples. The coverage of sets $\{S_1, \ldots, S_N\}$ is optimized by the MaxCover problem in Algorithm 1. Let a vanilla MaxCover solve the coverage of the original sets $\{\mathcal{R}_1, \ldots, \mathcal{R}_N\}$ with $\mathcal{U}_h = \mathcal{U}$.

Theorem 1. *If the B selected sets by solving a vanilla* MaxCover *on sets $\{\mathcal{R}_i\}_{i=1}^N$ are non-overlapping i.e., $\mathcal{R}_i \cap \mathcal{R}_j = \emptyset$ with $i \neq j$, then there $\exists\, \mathcal{U}_h$ such that* CoverICL*'s* MaxCover *problem has the same solution.*

Proof. If the vanilla MaxCover problem and CoverICL's MaxCover have the same solution, it means that they select the same examples $\{x^{(b)}\}_{b=1}^{B}$ for annotation. We prove the theorem by induction.

Base Case: At the first iteration of the MaxCover, we have budget $B = 1$. As there are no covered elements, the vanilla MaxCover selects the set R_i with the most elements, i.e.,

$$x^{(1)} := x_i = \arg\max_{i \in N} |R_i|.$$

Now, solving MaxCover over the sets S_i requires

$$x_i = \arg\max_{i \in N} |R_i| = \arg\max_{i \in N} |S_i|$$

in order to have the same selected example $x^{(1)}$. This holds, for instance, if $\mathcal{U}_h$ removes a portion of node neighbors from $\mathcal{R}_i$ (randomly or selectively) such that the ordering of sets by their number of elements remains the same. In that case since the relative order by number of elements is preserved, we have

$$\arg\max_{i \in N} |R_i| = \arg\max_{i \in N} |S_i|.$$

Thus, $\exists\, \mathcal{U}_h$ that satisfies the condition of Theorem 1 when $B = 1$.

Induction Hypothesis: When the budget is $B-1$, we assume that solving MaxCover over sets $\{\mathcal{R}_i\}_{i=1}^{N}$ and sets $\{S_i\}_{i=1}^{N}$ have the same solution $\{x^{(b)}\}_{b=1}^{B-1}$.

Induction Step: After selecting $B-1$ sets $\{\mathcal{R}^{(b)}\}_{b=1}^{B-1}$, the vanilla MaxCover optimization chooses the B-the set $\mathcal{R}^{(B)}$. As the B selected sets are non-overlapping (condition in Theorem 1), it means that the B-th selected set $\mathcal{R}^{(B)}$ does not contain any elements that are covered by the previously selected sets $\{\mathcal{R}^{(b)}\}_{b=1}^{B-1}$. Similarly, due to the induction hypothesis, CoverICL selects the same examples and because $S_i \subset \mathcal{R}_i\ \forall i$, the selected sets $\{S^{(b)}\}_{b=1}^{B}$ by CoverICL are also non-overlapping. As the B-th selected example is the solution to the B-th MaxCover iteration, it must have the largest number of elements, i.e.,

$$x^{(B)} := x_i = \arg\max |\mathcal{R}_i|$$

and

$$\hat{x}^{(B)} := \hat{x}_k = \arg\max |S_k|,$$

where $\hat{x}^{(B)}$ is the example selected by CoverICL. If $\hat{x}^{(B)} \neq x^{(B)}$, that means that there is a set S_k that has more elements $|S_k|$ than $|S_i|$ that corresponds to vanilla MaxCover selection x_i, i.e., $|S_k| > |S_i|$.

However, since $\mathcal{U}_h$ preserves the order by number of elements (Base Case) and the selected sets by CoverICL do not overlap (Induction Hypothesis), $|S_k| \not> |S_i|$ and leads to a contradiction. Thus, $\hat{x}^{(B)} = x^{(B)}$, and we have the same solution for the vanilla MaxCover and CoverICL's MaxCover problems. □

Table 4: Ablation study using (i) the estimated uncertainty scores by the LLM and (ii) random uncertainty scores for uncertainty-based methods at AGNews and SST2 datasets.

uncertainty estimation →	AGNews LLM scores	AGNews random scores	SST2 LLM scores	SST2 random scores
Patron	$69.39_{\pm 1.76}$	$64.18_{\pm 1.47}$	$78.64_{\pm 4.16}$	$74.86_{\pm 4.58}$
CoverICL	$70.95_{\pm 1.87}$	$67.70_{\pm 1.80}$	$79.03_{\pm 2.47}$	$78.77_{\pm 5.11}$

Theorem 1 suggests that CoverICL can approximate diversity-based selection when the most representative examples are well-separated. This benefits cases where the LLM's uncertainty scores are not indicative for the task (similarly to considering all examples in $\mathcal{U}$ as hard ones) and cases where diversity sampling is crucial for good ICL performance.

To empirically verify Theorem 1, we experiment with a target GPT-Neo (1.3B) LLM where uncertainty scores are generated (i) by the LLM itself and (ii) randomly. As a baseline, we use Patron (Yu et al., 2023), which is designed for fine-tuned based NLP and assumes the uncertainty scores are indicative for the task. As Table 4 shows, CoverICL is robust due to its core diversity-based selection and shows minor performance degradation when using random uncertainty scores. On the other hand, Patron underperforms CoverICL by up to 3.91% accuracy points as it does not adapt its selection process when diversity sampling is more important.

A.3 Heuristic Rule

As there is no validation set for hyperparameter tuning, we propose a heuristic rule to automatically adjust the hyperparameter m, that is used to create the m-nn graph $\mathcal{G}_m$. Given the number of hard examples $N_\theta = \lfloor \theta N \rfloor$ (Section 4.2), where $\theta \in [0, 1]$, the number of neighbors m is adjusted so that MaxCover covers at least $\hat{N}_\theta$ hard examples, $\hat{N}_\theta < N_\theta$, before we exhaust the budget B. This ensures that the selected examples are representative enough of the hard examples.

Assuming the graph has reciprocal edges, each node has approximately $\lceil \theta m \rceil$ and $\lceil \theta^2 m^2 \rceil$ hard examples as neighbors for 1-hop and 2-hop sets, respectively. Thus, we can cover approximately $\lceil B\theta m \rceil$ and $\lceil B\theta^2 m^2 \rceil$ hard examples if MAXCOVER has budget B. If MAXCOVER needs to cover at least $\hat{N}_\theta$ hard examples before terminated, we need to satisfy $\hat{N}_\theta \approx \lceil B\theta m \rceil$ (for 1-hop sets) and $\hat{N}_\theta \approx \lceil B\theta^2 m^2 \rceil$ (for 2-hop sets). Thus, the heuristic-based rule is given by

$$\begin{cases} m = \left\lceil \frac{\hat{N}_\theta}{\theta B} \right\rceil & \text{for 1-hop sets (COVERICL+)}, \\ m^2 = \left\lceil \frac{\hat{N}_\theta}{\theta^2 B} \right\rceil & \text{for 2-hop sets (COVERICL)}. \end{cases} \tag{7}$$

For COVERICL, we consider 2-hop neighbor sets, which are dense, and can improve its density-based selection. For COVERICL+, we consider 1-hop sets as the model re-evaluates its predictions to gradually identify harder examples. The heuristic rule is adjusted to the number of the examples N_θ that we account as hard ones, and we find that $\hat{N}_\theta = N_\theta/2$ works well across datasets. When we have iterations $T > 1$ for COVERICL+, the budget for the MAXCOVER becomes $B := B/T$ in Equation 7.

A.4 Active-Kmeans: A $k$means Approach

COVERICL performs diversity sampling over the active graph. Another solution to combine uncertainty and diversity sampling is to perform $k$means clustering (MacQueen et al., 1967) over the set of hard examples $\mathcal{U}_h$. Then, we can select representative examples for each cluster by sampling the example closest to its centroid. Here, the number of clusters for $k$means is B, so that we sample as many examples as the budget B allows. We refer to that approach as Active-Kmeans.

Yet, Active-Kmeans suffers from certain limitations: It is sensitive to outlier examples, such as out-of-distribution examples or examples with mispredicted uncertainty scores. In addition, it assumes that the B formed clusters are equally important, which may not always be the case.

COVERICL constructs the active graph and is a more dynamic approach than the Active-Kmeans baseline due to the MAXCOVER problem it solves. MAXCOVER computes an "influence region" around each example. Outliers have small influence regions, while examples that have the same influence regions are not evenly helpful. That way, MAXCOVER selects examples that interact with the most of the hard examples, but also capture distinct influence patterns, utilizing the limited budget better.

B Pipeline of Selective Annotation Methods

Table 5: Pipeline and time cost of compared methods.

Method	Selective Annotation		Inference
	Method	Time Cost	(same for all)
Random	Random	Zero-cost	k-shot ($k \ll B$)
Kmeans	Clustering	Independent of the LLM	k-shot ($k \ll B$)
Hardest	Uncertainty	Depends on LLM	k-shot ($k \ll B$)
Vote$k$	Vote$k$	Depends on LLM	k-shot ($k \ll B$)
COVERICL	COVERICL	Depends on LLM	k-shot ($k \ll B$)

We elaborate on selective annotation in practice:

1. Selective annotation methods, such as COVERICL, identify the examples to be annotated.
2. Human experts are employed to annotate the examples; this can be a time-consuming process depending on the task (e.g., GSM8K requires writing elaborate arithmetic steps).
3. LLMs performs ICL inference with the annotated examples. Inference is the same regardless of the selective annotation method used.

While LLM-based methods, such as COVERICL, Votek, and Patron, require more resources for Step 1, it is a one-time cost before human annotation and inference. Thus, we believe that COVERICL is suitable for practical settings. We will add this discussion in the final version.

We provide the comparison Table 5, where compared methods differ during the "Selection Phase". As it is shown, all methods have the same computation cost during inference. During selection, model-based methods (Votek, COVERICL) have a higher cost, but this cost is only needed before inference/deployment.

B.1 Selection Time Cost

In Table 6, we compare competing approaches based on their computation time during their selection process (during downstream inference, their time cost is the same). Random selection has zero cost. Vote$k$ and COVERICL ($T = 1$) have the same cost, while the cost doubles for COVERICL ($T = 2$). Nevertheless, hyper-parameter T for COVERICL can be tuned depending on the desired runtime of the selection process.

Table 6: Time complexity analysis with 5-shot ICL for different selection processes over 300 examples on a GeForce RTX 3090 (24GB GPU).

	Embedding (SBERT)	Uncertainty (GPT-Neo)
Amazon		
Random	0 secs	0 secs
Vote$k$	≈1 secs	1 min & 51 secs
CoverICL ($T = 1$)	≈1 secs	1 min & 51 secs
CoverICL ($T = 2$)	≈1 secs	≈ 3 mins & 42 secs
AGNews		
Random	0 secs	0 secs
Vote$k$	≈0.5 secs	3 mins & 48 secs
CoverICL ($T = 1$)	≈0.5 secs	3 mins & 48 secs
CoverICL ($T = 2$)	≈0.5 secs	≈ 7 mins & 36 secs

C Experimental Setting Details

C.1 Datasets

We performed empirical evaluation with nine NLP datasets that cover well-studied tasks, such as topic classification (AGNews (Zhang et al., 2015), TREC (Hovy et al., 2001)), sentiment analysis (SST2 (Socher et al., 2013), Amazon (McAuley and Leskovec, 2013)), natural language inference (RTE (Bentivogli et al., 2009), MRPC (Dolan et al., 2004), MNLI (Williams et al., 2018)), text summarization (XSUM (Narayan et al.)) and math reasoning (GSM8K (Cobbe et al., 2021)).

Each dataset contains official train/dev/test splits. We follow Vote$k$ and sample 256 examples randomly from the test set (if it is publicly available, otherwise from the dev set) as test data. For the train data, we remove the annotations before our setup. As it is infeasible to evaluate the LLM's feedback on all instances due to computational constraints, e.g., Amazon dataset has more than 1 million instances, we randomly subsample 3,000 instances, which we cluster into 310 groups, and we select the 310 examples closest to the centroids as candidate examples for annotation. We repeat the above processes for both the train and test sets three times with different seeds and report mean and standard deviation results. In transductive settings, we evaluate performance on the unlabeled examples, but we also exclude retrieving examples that lead to self-label leakage issues.

C.2 Configurations

As summarized in Figure 2, the design space includes the unlabeled set $\mathcal{U}$, the number of ICL examples k, the similarity space, the budget B, and the LLM M. We experiment with six LLMs of varying sizes (1.3B to 65B parameters), including GPT-J (Wang and Komatsuzaki, 2021), Mosaic (MosaicML, 2023), Falcon (Penedo et al., 2023), and LLaMa (Touvron et al., 2023) models, all of which are open-source and allow the reproducibility of our research. We use the default hyper-parameters of the Transformers library (Wolf et al., 2020) for each LLM. We experiment with inductive settings, where test instances come from an *unseen* set $\mathcal{U}_{\text{test}}$, but also for transductive settings, where test instances come from $\mathcal{U}$. We obtain the initial pool of annotated examples $\mathcal{L}_0$ via $k$means so that we reduce randomness. We summarize the experimental configurations in Table 7.

D Further Experiments

D.1 Full Results & Significance Test

We present the full results of Table 1 in Table 8 (GPT-J-6B) and in Table 9 (GPT-Neo-1.3B).

We also run a significance test based on the Wilcoxon Signed-Rank (Demšar, 2006), which is a ranking-based metric that accounts for performance differences. We provide the results comparing CoverICL with Patron, the SOTA method for low-budget AL, using GPT-J. According to the results in Figure 7, CoverICL's performance **is significantly better** than Patron's performance at $p < .05$ (see the last line of the appended results, highlighted in blue color).

D.2 Other LLMs

CoverICL takes advantage of how well the LLM understands the underlying task. If the LLM does not understand the task, then CoverICL approximates diversity sampling, which is less affected by the LLM (Theorem 1).

Following, we use Falcon-40B and phi-2 (2.7B) (Gunasekar et al., 2023) on college exam questions (MMLU Bio/Math) (Hendrycks et al., 2020). Falcon-40B has a capacity of 40B parameters and thus broader knowledge to understand the task. Phi-2 has been pretrained on textbooks and science texts, having task-specific knowledge. Table 10 shows that CoverICL improves both of the LLMs compared to IDEAL, although these models have different sizes.

D.3 Uncertainty Threshold

By default, we consider 50% ($\theta = 0.5$) of the examples with the lowest confidence as hard examples. Table 11 shows results when we focus on harder examples by setting $\theta = 0.33$ for CoverICL+. Interestingly, CoverICL+'s performance can be

Table 7: Experimental setting configurations.

Setting	Models M	Train/Test $\mathcal{U}$	Budget B	k-shot	Retriever, $\mathcal{S}$	Init.
Table 8	GPTJ	Inductive	20	5	SBERT	$\|\mathcal{L}_0\| = 10$
Tables 3, 11	GPT-Neo	Inductive	20	5	SBERT	$\|\mathcal{L}_0\| = 10$
XSUM	Falcon-40B, LLaMa-65B	Transductive	10	Context-limit	SBERT	Zero-shot
GSM8K	Falcon-40B, LLaMa-65B	Transductive	20	5	BERT,SBERT	Zero-shot
Table 2	GPT-Neo	Inductive	20	5	SBERT, RoBERTa, BERT	$\|\mathcal{L}_0\| = 10$
Figure 6	GPT-Neo	Transductive	0-45	Context-limit	SBERT	Zero-shot
Figure 5a	GPT-J, MPT, Falcon, LLaMa (6-7B)	Transductive	0-20	Context-limit	SBERT	Zero-shot
Appendix D	GPT-J, GPT-Neo	Inductive	20	5	SBERT	$\|\mathcal{L}_0\| = 10$

Context-limit means that we retrieve as many few-shot examples as the input token-length limit allows. For example, XSUM has long sequences, where we usually have 3-shot examples, while for TREC we can use as many as 80-shot examples.

Table 8: Performance comparison for GPT-J (6B).

	Topic Classification		**Sentiment Analysis**		**Natural Language Inference**		
	AGNews	TREC	SST2	Amazon	RTE	MRPC	MNLI
Random	$68.87_{\pm 5.39}$	$49.34_{\pm 3.19}$	$81.63_{\pm 0.30}$	$87.89_{\pm 1.77}$	$52.86_{\pm 2.41}$	$69.01_{\pm 4.61}$	$39.58_{\pm 3.98}$
Hardest	$72.13_{\pm 2.12}$	$35.93_{\pm 5.53}$	$82.67_{\pm 1.64}$	$87.36_{\pm 0.66}$	$55.33_{\pm 2.25}$	$66.80_{\pm 5.25}$	$38.80_{\pm 1.11}$
Fast-vote$k$	$73.69_{\pm 2.39}$	$49.61_{\pm 4.43}$	$78.99_{\pm 4.53}$	$89.58_{\pm 0.80}$	$53.00_{\pm 0.49}$	$68.23_{\pm 2.89}$	$39.97_{\pm 3.98}$
IDEAL	$74.73_{\pm 2.43}$	$39.57_{\pm 6.33}$	$78.38_{\pm 8.14}$	$88.41_{\pm 0.80}$	$56.77_{\pm 0.48}$	$66.27_{\pm 3.73}$	$40.23_{\pm 1.59}$
Vote$k$	$72.26_{\pm 1.27}$	$45.83_{\pm 1.75}$	$80.45_{\pm 1.47}$	$85.80_{\pm 3.80}$	$54.16_{\pm 2.30}$	$68.10_{\pm 2.75}$	$39.72_{\pm 2.07}$
Patron	$75.90_{\pm 1.81}$	$44.13_{\pm 6.92}$	$81.89_{\pm 6.39}$	$90.88_{\pm 2.57}$	$55.20_{\pm 1.27}$	$66.40_{\pm 5.25}$	$38.53_{\pm 2.57}$
Active-Kmeans	$73.56_{\pm 2.96}$	$50.64_{\pm 9.11}$	$84.11_{\pm 3.25}$	$91.01_{\pm 1.77}$	$52.73_{\pm 2.21}$	$66.53_{\pm 4.78}$	$38.66_{\pm 4.50}$
Best (Avg.)	Active-Kmeans (62.10)		Active-Kmeans (87.56)		IDEAL (54.42)		
CoverICL	$76.89_{\pm 3.01}$	$51.95_{\pm 8.43}$	$82.81_{\pm 1.39}$	$90.49_{\pm 1.57}$	$56.90_{\pm 1.75}$	$70.17_{\pm 1.72}$	$40.36_{\pm 1.75}$
CoverICL+	$77.08_{\pm 1.11}$	$53.38_{\pm 5.10}$	$84.24_{\pm 1.32}$	$92.45_{\pm 1.50}$	$55.07_{\pm 0.85}$	$68.49_{\pm 0.97}$	$36.58_{\pm 1.12}$
Best (Avg.)	CoverICL+ (65.23)		CoverICL+ (88.35)		CoverICL (55.81)		
Δ-Gain (Absolute)	+3.13		+0.79		+1.39		

further boosted with careful tuning of the uncertainty threshold. Thus, *automatically* determining which examples are considered as hard examples for the models seems a promising research direction.

D.4 Graph Ablation

We experiment on the importance of graph-based algorithms, such as CoverICL. We implement different selection algorithms, such as clustering methods, that rely on (i) distances between points (Kmeans, Hierarchical Clustering), (ii) distances between graph-nearest points (Max Degree, Graph Clustering, Graph Propagation, Graph MaxCover), or (iii) none of the previous (Max Uncertainty). Methods in '(ii)' are graph-based.

As Table 12 shows, our proposed Graph-based MaxCover (CoverICL) algorithm outperforms competing alternatives. Overall, graph-based algorithms outperform non-graph methods, showing the importance graph-based solutions for ICL.

Furthermore, we experiment using a threshold-based graph (δ-graph) instead of the m-nn graph. To determine threshold δ, we compute the cosine similarity between all nodes and set δ such as each node has m neighbors *on average* (at the m-nn graph each nodes has exactly m neighbors). As Table 13 shows, using the δ-graph performs slightly worse than the m-nn graph. We hypothesize that using the δ-graph gives more importance on the semantics of the train distribution (as δ value is computed based on the similarity scores between all train examples), which may not always generalize well to the test distribution.

E Task Prompts

As a design choice of the input prompts, we slightly modify the templates proposed by Gao et al. (2021) to transform them as a continuation task. We find that these are more challenging prompts for the large LMs, which we present in Table 14 (top). Next, we experiment with alternative prompt templates similar to Su et al. (2023), as shown in Table 14 (bottom).

Table 15 reports results when we use alternative ICL prompt templates (Table 14) for the input examples. CoverICL is robust to the design of the prompt templates, where it outperforms other baselines in most datasets.

F Dataset Examples

We provide examples of these datasets in Table 16, which we access via Hugging Face package (Lhoest et al., 2021).

Table 9: Performance comparison for GPT-Neo (1.3B).

	Topic Classification		Sentiment Analysis		Natural Language Inference		
	AGNews	TREC	SST2	Amazon	RTE	MRPC	MNLI
Random	$59.47_{\pm 8.54}$	$54.68_{\pm 1.68}$	$68.48_{\pm 1.87}$	$73.95_{\pm 2.03}$	$48.30_{\pm 1.30}$	$64.48_{\pm 7.67}$	$40.99_{\pm 0.97}$
Fast-vote$k$	$62.23_{\pm 3.89}$	$46.48_{\pm 3.04}$	$69.78_{\pm 8.34}$	$69.39_{\pm 0.98}$	$50.64_{\pm 1.02}$	$64.19_{\pm 0.97}$	$38.40_{\pm 0.92}$
IDEAL	$69.00_{\pm 2.17}$	$44.66_{\pm 9.22}$	$73.17_{\pm 5.10}$	$67.70_{\pm 0.97}$	$52.47_{\pm 2.94}$	$63.27_{\pm 4.18}$	$36.97_{\pm 1.47}$
Vote$k$	$62.77_{\pm 4.82}$	$53.12_{\pm 4.07}$	$73.69_{\pm 9.05}$	$75.13_{\pm 0.98}$	$49.99_{\pm 0.32}$	$67.44_{\pm 2.96}$	$39.18_{\pm 1.60}$
Hardest	$65.10_{\pm 2.43}$	$49.34_{\pm 2.17}$	$71.48_{\pm 5.32}$	$75.00_{\pm 2.49}$	$52.86_{\pm 0.80}$	$61.84_{\pm 4.79}$	$37.49_{\pm 1.77}$
Patron	$69.39_{\pm 1.87}$	$49.34_{\pm 2.17}$	$71.48_{\pm 5.32}$	$75.00_{\pm 2.49}$	$52.86_{\pm 0.80}$	$61.84_{\pm 4.79}$	$37.49_{\pm 1.77}$
Active-Kmeans	$70.17_{\pm 1.84}$	$48.24_{\pm 0.98}$	$77.86_{\pm 1.02}$	$75.77_{\pm 3.62}$	$53.77_{\pm 0.73}$	$64.71_{\pm 7.39}$	$39.71_{\pm 1.03}$
Best (Avg.)	Active-Kmeans (59.21)		Active-Kmeans (76.82)		Vote$k$ (52.20)		
CoverICL	$70.95_{\pm 1.87}$	$55.33_{\pm 2.57}$	$79.68_{\pm 1.77}$	$77.73_{\pm 2.23}$	$53.12_{\pm 1.59}$	$67.05_{\pm 8.10}$	$42.96_{\pm 2.92}$
CoverICL+	$69.39_{\pm 1.35}$	$59.89_{\pm 2.07}$	$79.03_{\pm 2.47}$	$77.08_{\pm 1.50}$	$51.16_{\pm 1.39}$	$65.69_{\pm 8.92}$	$40.49_{\pm 2.04}$
Best (Avg.)	CoverICL+ (64.64)		CoverICL (78.71)		CoverICL (54.38)		
Δ-Gain (Absolute)	+5.53		+1.99		+2.28		

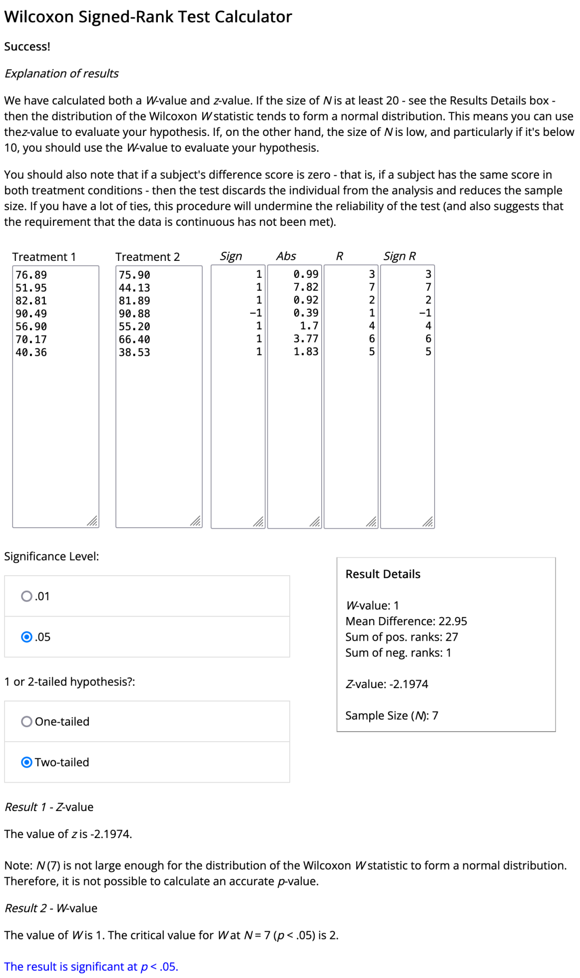


Figure 7: Significance test based on Wilcoxon Signed-Rank at p-value < 0.05.

Table 10: Performance comparison with LLMs of different sizes.

	MMLU-Bio		MMLU-Math	
	phi-2 (2.7B)	Falcon-40B	phi-2 (2.7B)	Falcon-40B
IDEAL	64.58	63.88	42.00	43.00
CoverICL	65.28	68.05	47.00	44.00

Table 11: Ablation study on hyper-parameter θ, which controls the number of the examples that are considered as hard ones.

	TREC	SST2	Amazon
Vote$k$	$53.12_{\pm 4.07}$	$73.69_{\pm 9.05}$	$75.13_{\pm 0.98}$
*CoverICL+ ($\theta = 0.5$)	$59.89_{\pm 2.07}$	$79.03_{\pm 2.47}$	$77.08_{\pm 1.50}$
CoverICL+ ($\theta = 0.33$)	$60.28_{\pm 3.13}$	$78.77_{\pm 2.59}$	$78.90_{\pm 1.14}$

*Denotes the default value.

G Visualization

We illustrate the selection process of CoverICL+ in Figure 8. Initially, the LLMs perform 0-shot ICL but do not make confident predictions (as the hue color indicates, that represents the model's uncertainty for each example). Note that different LLMs may consider different examples as hard or easy ones. Then, CoverICL+ selects 5 representative examples for 5-shot ICL, which improves the LLMs' understanding of the task and reduces its uncertainty (we observe fewer red nodes and more nodes with greener color).

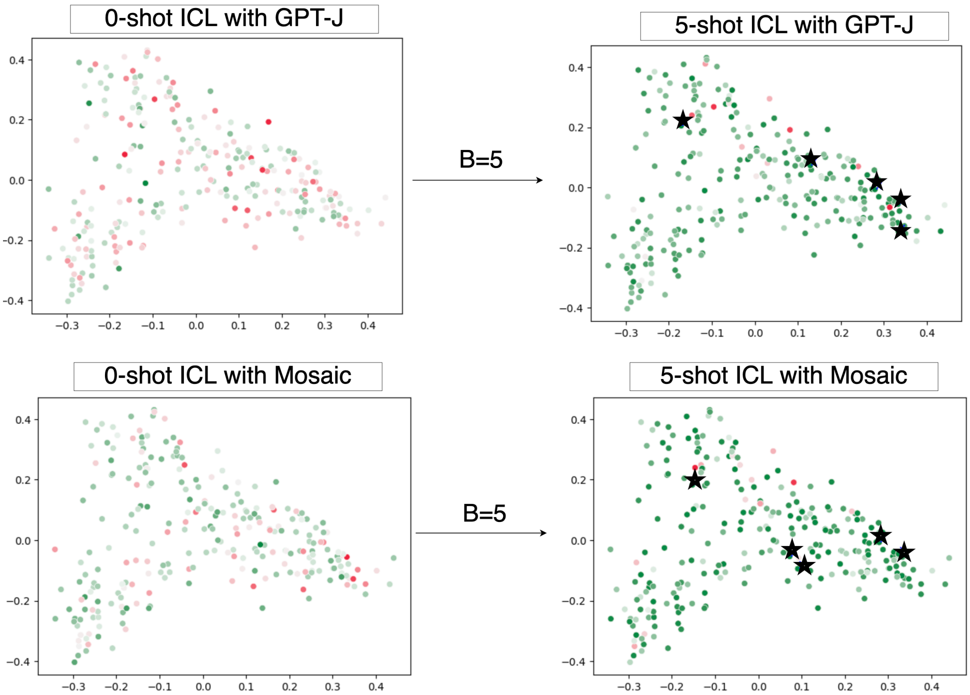


Figure 8: Visualization of COVERICL+'s selection process for AGNews. The plots visualize the SBERT embeddings (after PCA), where the hue color (green to red) represents the model's uncertainty (confident to uncertain) for each example. The selected examples by COVERICL+ are marked with the '⋆' symbol.

Table 12: Performance comparison of graph-based and non-graph methods.

	AGNews	SST2
(i) Kmeans (Active-Kmeans)	73.56	84.11
(i) Hierarchical clustering	72.93	80.79
(ii) Max Degree (threshold-based)	76.17	81.90
(ii) Graph Clustering (Fast-votek)	73.69	78.99
(ii) Graph Uncertainty Propagation (Patron)	75.90	81.89
(ii) Graph-based MAXCOVER (**COVERICL**)	**77.08**	**84.24**
(iii) Max Uncertainty (Hardest)	72.13	82.67

Table 13: Graph ablation study for COVERICL using GPT-J with different graph construction approaches.

	AGNews	TREC	SST2	Amazon	RTE	MRPC	MNLI
m-nn graph	$77.08_{\pm 1.11}$	$53.38_{\pm 5.10}$	$84.24_{\pm 1.32}$	$92.45_{\pm 1.50}$	$56.90_{\pm 1.75}$	$70.17_{\pm 1.72}$	$40.36_{\pm 1.75}$
δ-graph	$76.17_{\pm 3.45}$	$50.51_{\pm 4.65}$	$81.90_{\pm 2.48}$	$88.80_{\pm 1.57}$	$56.63_{\pm 3.23}$	$68.75_{\pm 1.93}$	$40.62_{\pm 2.08}$

Table 14: Prompt templates for the ICL demonstrations.

Task	**Template**	**Continuation (label word)**
	Default	
AGNews	Content: $< S_1 > \backslash n$	World, Sport, Business, Sci-Tech
TREC	Content: $< S_1 > \backslash n$	Abbreviation, Entity, Description, Human, Location, Numeric
SST2	$< S_1 >$. It was	great, terrible
Amazon	$< S_{1a} >< S_{1b} >$. It was	great, terrible
RTE	$< S_1 >$? [MASK], $< S_2 >$	[MASK]: Yes, [MASK]: No
MRPC	$< S_1 >$? [MASK], $< S_2 >$	[MASK]: Yes, [MASK]: No
MNLI	$< S_1 >$? [MASK], $< S_2 >$	[MASK]: Yes, [MASK]: Maybe, [MASK]: No
	Alternative	
AGNews	Content: $< S_1 >$ Topic:	World, Sport, Business, Sci-Tech
TREC	Content: $< S_1 >$ Answer Type:	Abbreviation, Entity, Description, Human, Location, Numeric
SST2	Content: $< S_1 >$ Sentiment:	Positive, Negative
Amazon	Title: $< S_{1a} >$ Review: $< S_{1b} >$ Sentiment:	Positive, Negative
RTE	$< S_1 >$. Question: $< S_2 >$. True or False? Answer:	True, False
MRPC	Are the following sentences equivalent or not equivalent? $< S_1 > \backslash n < S_2 >$	equivalent, not equivalent
MNLI	$< S_1 >$. Based on that information, is the claim $< S_2 >$ True, False, or Inconclusive? Answer:	True, Inconclusive, False

Table 15: Prompt template ablation study.

	GPT-Neo				GPT-J		
	AGNews	TREC	SST2	Amazon	RTE	MRPC	MNLI
	Default Prompts						
Random	$59.47_{\pm 8.54}$	$54.68_{\pm 1.68}$	$68.48_{\pm 1.87}$	$73.95_{\pm 2.03}$	$52.86_{\pm 2.41}$	$69.01_{\pm 4.61}$	$39.58_{\pm 3.98}$
Vote$k$	$62.77_{\pm 4.82}$	$53.12_{\pm 4.07}$	$73.69_{\pm 9.05}$	$75.13_{\pm 0.98}$	$54.16_{\pm 2.30}$	$68.10_{\pm 2.75}$	$39.72_{\pm 2.07}$
COVERICL	$70.95_{\pm 1.87}$	$55.33_{\pm 2.57}$	$79.68_{\pm 1.77}$	$77.73_{\pm 2.23}$	$56.90_{\pm 1.75}$	$70.17_{\pm 1.72}$	$40.36_{\pm 1.75}$
	Alternative Prompts						
Random	$73.69_{\pm 1.21}$	$51.76_{\pm 4.55}$	$59.89_{\pm 3.98}$	$73.82_{\pm 3.35}$	$56.41_{\pm 2.13}$	$56.37_{\pm 3.72}$	$38.93_{\pm 1.18}$
Vote$k$	$72.78_{\pm 2.12}$	$50.38_{\pm 5.90}$	$64.84_{\pm 2.92}$	$73.43_{\pm 2.23}$	$56.38_{\pm 2.70}$	$51.95_{\pm 2.53}$	$40.49_{\pm 2.05}$
COVERICL	$76.95_{\pm 1.27}$	$54.94_{\pm 1.43}$	$65.88_{\pm 4.58}$	$75.64_{\pm 1.29}$	$56.37_{\pm 1.29}$	$59.22_{\pm 2.39}$	$35.40_{\pm 1.31}$

Table 16: Dataset examples. $< S_1 >$ denotes the input sequences.

Dataset	**Task**	**Example** x	**Labels/Annotations** y
AGNews	Topic Classification	$< S_1 >$: “Amazon Updates Web Services Tools, Adds Alexa Access The Amazon Web Services (AWS) division of online retail giant Amazon.com yesterday released Amazon E-Commerce Service 4.0 and the beta version of Alexa Web Information Service.”	World, Sport, Business, Sci-Tech
TREC	Answer Type Classification	$< S_1 >$: “What is the date of Boxing Day?”	Abbreviation, Entity, Description, Human, Location, Numeric
SST2	Sentiment Analysis	$< S_1 >$: “covers this territory with wit and originality , suggesting that with his fourth feature”	Positive, Negative
Amazon	Sentiment Analysis	$< S_{1a} >$:“Very Not Worth Your Time”, $< S_{1b} >$:“The book was written very horribly. I would never in my life recommend such a book..."	Positive, Negative
RTE	Natural Language Inference	$< S_1 >$:“In a bowl, whisk together the eggs and sugar until completely blended and frothy.”, $< S_2 >$:“In a bowl, whisk together the egg, sugar and vanilla until light in color.”	Entailment, Not Entailment
MRPC	Paraphrase Detection	$< S_1 >$:“He said the foodservice pie business doesn’t fit the company’s long-term growth strategy.”, $< S_2 >$:“The foodservice pie business does not fit our long-term growth strategy.”	Equivalent, Not Equivalent
MNLI	Natural Language Inference	$< S_1 >$:“The new rights are nice enough”, $< S_2 >$: “Everyone really likes the newest benefits”	Entailment, Neutral, Contradiction
XSUM	Summarization	$< S_1 >$:“The 3kg (6.6lb) dog is set to become part of a search-and-rescue team used for disasters such as earthquakes. Its small size means it will be able to squeeze into places too narrow for dogs such as German Shepherds. Chihuahuas, named after a Mexican state, are one of the the smallest breeds of dog. "It’s quite rare for us to have a chihuahua work as a police dog (said a police spokeswoman in Nara, western Japan). We would like it to work hard by taking advantage of its small size. Momo, aged seven, will begin work in January.”	“A chihuahua named Momo (Peach) has passed the exam to become a dog in the police force in western Japan, in what seems to be a first.”
GSM8K	Math Reasoning	$< S_1 >$:“James writes a 3-page letter to 2 different friends twice a week. How many pages does he write a year?”	“He writes each friend 3*2=6 pages a week So he writes 6*2=12 pages every week That means he writes 12*52=624 pages a year. Thus, the answer is 624.”