Unraveling Indirect In-Context Learning Using Influence Functions

Anonymous ACL submission

Abstract

In this work, we introduce a novel paradigm for generalized In-Context Learning (ICL), termed *Indirect In-Context Learning*. In Indirect ICL, we explore demonstration selection strategies tailored for two distinct real-world scenarios: *Mixture of Tasks* and *Noisy Demonstrations*. We systematically evaluate the effectiveness of Influence Functions (IFs) as a selection tool for these settings, highlighting the potential of IFs to better capture the informativeness of examples within the demonstration pool. For the Mixture of Tasks setting, demonstrations are drawn from 28 diverse tasks, including MMLU, BigBench, StrategyQA, and CommonsenseQA. We demonstrate that combining BertScore-Recall (BSR) with an IF surrogate model can further improve performance, leading to average absolute accuracy gains of 0.37% and 1.45% for 3-shot and 5-shot setups when compared to traditional ICL metrics. In the Noisy Demonstrations setting, we examine scenarios where demonstrations might be mislabeled. Our experiments show that reweighting traditional ICL selectors (BSR and Cosine Similarity) with IF-based selectors boosts accuracy by an average of 2.90% for Cosine Similarity and 2.94% for BSR on noisy GLUE benchmarks. In sum, we propose a robust framework for demonstration selection that generalizes beyond traditional ICL, offering valuable insights into the role of IFs for Indirect ICL.

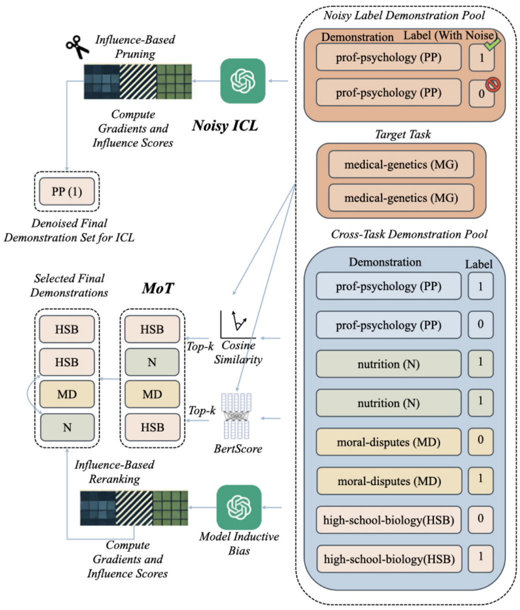

Figure 1: Example showcasing demonstration selection for Indirect ICL using Influence Functions (IFs). Consider web corpora with many tasks (different from the end-task) and noisy data– Indirect ICL can be formalized as: *Mixture of Tasks* (Section 3.1) and *Noisy* (Section 3.2) ICL, respectively. In MoT, for a given target task (e.g. *Medical Genetics*), we first filter from this (indirect) pool of candidate demonstrations using BertScore and Cosine Similarity, then re-rank with IFs to select suitable demonstrations (e.g. *High-School Biology*). For Noisy ICL, we leverage IFs to filter out the Noisy demonstrations before conducting ICL with the remaining clean demonstrations.

1 Introduction

In-Context Learning (ICL) has emerged as a powerful method for utilizing large language models (LLMs) to handle novel tasks at inference (Mann et al., 2020; Min et al., 2022). Unlike traditional approaches that require task-specific fine-tuning, ICL allows a single model to adapt to different tasks without additional training, relying solely on the demonstrations provided in the context. This flexibility not only reduces the cost of task adaptation but also offers a transparent and easily customizable way of guiding the model's behavior (Liu et al., 2021a; Wei et al., 2022). By leveraging the context provided in prompts, ICL has been shown to improve both generalization across diverse tasks and reasoning abilities (Anil et al., 2022; Drozdov et al., 2022). Despite its advantages, the success of ICL is closely tied to the choice of demonstrations used in the prompt. Even slight variations in these demonstrations can significantly influence the model's performance, as shown in numerous studies (Zhao et al., 2021; Liu et al., 2021a; Lu et al., 2022).

Traditional ICL makes numerous assumptions that restrict its applicability to real-world problem domains. For instance, traditional ICL (Min et al., 2021; Conneau, 2019; Halder et al., 2020) assumes that demonstrations to be selected are *directly* and accurately annotated for the end-task. However, this is not always the case – for low-resource, sparse, or specialized domains, end-task information and labeled demonstrations might not be available.[1] Similarly, when LLMs are deployed as services, the user query or the end task itself could be unknown beforehand, let alone providing direct demonstrations at inference.[2] Thus, in this paper, we explore a more generalized setting for ICL, which we refer to as ***Indirect ICL***.

In Indirect ICL, we aim to provide indirect (or incidental) supervision (Yin et al., 2023; Li et al., 2024) by selecting demonstrations from a pool of examples where the majority are not directly suited to the end task due to severe distribution/covariate shifts. This includes selecting demonstrations from a pool that predominantly consists of demonstrations belonging to other tasks, with few demonstrations from the end task possibly included. Additionally, the demonstration set may be mislabeled by humans (Yan et al., 2014; Zhu et al., 2022) or LLMs (Wu et al., 2023). Since the effectiveness of ICL heavily relies on the quality of demonstrations selected (Kossen et al., 2024; Wu et al., 2022; Wang et al., 2024), selecting the most helpful indirect demonstrations becomes imperative in these situations.

Despite these potential issues with the demonstration set, we wish to pave the way for extracting maximal benefit from any type of annotated dataset, irrespective of label purity or task relatedness. Thus, in order to combat the aforementioned issues with sub-optimal datasets for ICL, we leverage *Influence Functions (IFs)* (Hampel, 1974; Cook and Weisberg, 1980). IFs offer a formal method for assessing how individual training data points affect model predictions. They have proven effective in a range of downstream machine learning tasks, including mislabeled data detection (Koh and Liang, 2017; Pruthi et al., 2020), optimal subset selection (Feldman and Zhang, 2020; Guo et al., 2020; Xia et al., 2024), model interpretation (Han et al., 2020; Grosse et al., 2023; Chhabra et al., 2024b), data attribution (Bae et al., 2024), data valuation (Choe et al., 2024) and analyzing model biases (Wang et al., 2019; Kong et al., 2021).

Traditional (direct) ICL methods that use metrics such as BertScore-Recall (BSR; Gupta et al. 2023a) and cosine similarity (Reimers, 2019) inherently rely on the semantic similarity between demonstrations and test samples. In this paper, we posit that IFs can be a reasonable measure of affinity between the end task and any (indirect) demonstrations. We show that it is practical to use IFs to identify candidate demonstrations that represent a close inductive bias with the end-task, and utilize this information for highly accurate demonstration selection in the challenging Indirect ICL setting. As our experiments and results will demonstrate, this is indeed the case, and we find that IFs can aid in improved performance when simple semantic similarity is insufficient for demonstration selection. We provide additional examples of practical applications of Indirect ICL in Appendix A.

In sum, our work advances ICL demonstration selection and makes the following key contributions and findings:

- We formalize a new and general paradigm for ICL, namely Indirect In-Context Learning, where we benchmark demonstration selection for two distinct and real-world settings: (a) **Mixture of Tasks** and (b) **Noisy Demonstrations**. This novel paradigm with two settings is ubiquitous in the real world, and has yet been overlooked by existing research in ICL that assumes the availability of direct supervision.
- We propose utilizing Influence Functions (IFs) as an effective approach for demonstration selection in generalized ICL settings, leveraging their capacity to exploit the task inductive bias of models to enhance selection quality. We also examine multiple influence functions for Indirect ICL and conduct an extensive analysis on their benefits in this setting.
- For Mixture of Tasks, combining an IF Surrogate model with BertScore-Recall (BSR) can lead to a 0.37% and 1.45% average absolute increase in performance for $k = 3$ and $k = 5$ shots compared to the best performing traditional ICL metric.
- For Noisy Demonstrations, we observe that undertaking a weighted average selection using

[1] Consider the cases where we need to utilize ICL for diagnosing rare medical conditions, niche programming languages or indigenous spoken languages.

[2] Our proposed method can improve performance by selecting relevant demonstrations from a task agnostic pool of labeled data at test time.

traditional ICL selectors (BSR and Cosine Similarity) and IF based selectors increases the absolute accuracy on average by 2.90% for Cosine and 2.94% for BSR.

2 Preliminaries

We hereby introduce preliminaries of ICL and IF.

2.1 Traditional In-Context Learning

Before we define the more generalized problem of Indirect ICL, we first define traditional ICL.

In-Context Learning. ICL allows LLMs to solve test inputs from novel tasks by presenting a few examples of the task in the prompt. Formally, given a set of input x and output y pairs $\{(x_i, y_i)\}_{i=1}^{k}$, prompt template T, and the test input x_{test}, ICL using an LLM involves prompting it to conditionally generate the test output y_{test} according to the following distribution:

$$y_{\text{test}} \sim P_{\text{LM}}(\cdot \mid T(x_1, y_1, \ldots, x_k, y_k, x_{\text{test}}))$$

Demonstration Selection. In this work we study the problem of selecting k in-context examples from a pool of $N \gg k$ labeled candidates. This is often necessary due to context length limits and cost considerations (Rubin et al., 2021; Gupta et al., 2023a). Formally, the goal is to select a subset $S \subset \{(x_i, y_i)\}_{i=1}^{N}$ of size k that maximizes the probability of generating the desired y_{test} when the LLM is conditioned on x_{test} and S. It is noteworthy that prior studies mainly consider a task-dependent ICL scenario and assume that candidate demonstrations all directly match the end task (Min et al., 2021; Conneau, 2019; Halder et al., 2020).

2.2 Indirect In-Context Learning

Now, we describe two scenarios of Indirect ICL, one where the candidate pool comprises of demonstrations from various tasks and the other where the demonstrations may have noisy labels.

Mixture of Tasks. Unlike traditional ICL, where candidate demonstrations match the end task at inference, we consider the more generalized Indirect ICL setting where the demonstration pool is task-agnostic. In practice, this setting would allow for pooling annotated demonstrations from various accessible tasks. Formally, given a set of input x and output y pairs $\{(x_i, y_i)\}_{i=1}^{k}$, where the pairs (x_i, y_i) may originate from different tasks than the test input x_{test}, the model is prompted to maximize performance across test tasks.

Noisy Demonstrations. To further generalize the problem of Indirect ICL, we also consider noisy supervision that is likely existing in the pool of demonstrations. Formally, let $D = \{(x_i, y_i)\}_{i=1}^{n}$ represent the training dataset, where $x_i \in X$ is the input and $y_i \in Y$ is the corresponding binary label. We randomly select a percentage of the data points from D and flip their labels. Once the noisy dataset is generated, we use it for ICL. Formally, given the noisy set of input-output pairs $\{(x_i, y_i)\}_{i=1}^{k}$ and a test input x_{test}, the goal is to conditionally generate the test output y_{test} based on the noisy training data.

2.3 Influence Functions

Here we formally define how we will use IFs to perform Generalized Indirect ICL.

Let the input space be X and the label space be Y. The training dataset is denoted as $D = \{(x_i, y_i)\}_{i=1}^{n}$, where $x_i \in X$ and $y_i \in Y$ are the input and label of the i-th data point. Given a loss function ℓ and a parameter space Θ, the empirical risk minimization problem is defined as:

$$\theta^* = \arg\min_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f_\theta(x_i)),$$

where $f_\theta : X \to Y$ is the model parameterized by $\theta \in \Theta$. The gradient of the loss for the i-th data point with respect to a vector η is denoted as:

$$\nabla_\eta \ell_i = \nabla_\eta \ell(y_i, f_\theta(x_i)).$$

The IF evaluates the effect of individual training data points on the estimation of model parameters (Hampel, 1974; Cook and Weisberg, 1980; Martin and Yohai, 1986). It measures the rate at which parameter estimates change when a specific data point is up-weighted.

Specifically, for $k \in [n]$ and $\epsilon \in \mathbb{R}$, we consider the following ϵ-weighted empirical risk minimization problem:

$$\theta^{(k)}(\epsilon) = \arg\min_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f_\theta(x_i)) + \epsilon \ell(y_k, f_\theta(x_k)).$$

Here, the loss function $\ell(y, f_\theta(x))$ is assumed to be twice-differentiable and strongly convex in θ for all $(x, y) \in \mathcal{X} \times \mathcal{Y}$, the empirical risk minimizer (model weights) θ^* is well-defined, and the influence of the k-th data point $(x_k, y_k) \in D$ on the empirical risk minimizer (model weights) θ^* is defined as the derivative of $\theta^{(k)}(\epsilon)$ at $\epsilon = 0$:

$$I_{\theta^*}(x_k, y_k) := \left.\frac{d\theta^{(k)}}{d\epsilon}\right|_{\epsilon=0} = -H(\theta^*)^{-1} \nabla_\theta \ell(y_k, f_\theta(x_k)).$$

where $H(\theta) := \nabla_\theta^2 \left(\frac{1}{n}\sum_{i=1}^{n} \ell(y_i, f_\theta(x_i))\right)$ is the Hessian of the empirical loss.

The IF $I_{\theta^*}(x_k, y_k)$ on the empirical risk minimizer θ^* is generalized to assess its effect on prediction loss (Koh and Liang, 2017). Given a validation dataset $D^{\mathcal{V}} := \{(x_i^{\mathcal{V}}, y_i^{\mathcal{V}})\}_{i=1}^{m}$, the influence of (x_k, y_k) on the validation loss is defined as:

$$I(x_k, y_k) := \left(\frac{1}{m}\sum_{i=1}^{m} \nabla_\theta \ell(y_i^{\mathcal{V}}, f_\theta(x_i^{\mathcal{V}}))\Big|_{\theta=\theta^*}\right)^\top \times I_{\theta^*}(x_k, y_k).$$

This gives us

$$I(x_k, y_k) = -\sum_{i=1}^{m} \Big(\nabla_\theta \ell(y_i^{\mathcal{V}}, f_\theta(x_i^{\mathcal{V}}))^\top H(\theta^*)^{-1} \nabla_\theta \ell(y_k, f_\theta(x_k))\Big).$$

The IF $I(x_k, y_k)$ provides insight into how a single data point impacts the validation loss. Essentially, it indicates whether (x_k, y_k) contributes positively or negatively to the prediction loss. The more positive the influence value, the more it contributes to the loss decreasing, hence it is a beneficial data point to train the model.

Remark. As discussed above, IFs assume convexity of the loss function, which does not hold for LLMs and deep neural networks. Even though the IF formulations we employ in this paper (Kwon et al., 2023; Koh and Liang, 2017) make this underlying assumption, we find through empirical observations that for indirect ICL, they can work well. Circumventing the convexity assumption in IF is an ongoing area of research (Grosse et al., 2023; Chhabra et al., 2024a) and our framework is flexible enough to accommodate any future IF variants.

3 Proposed Approach

In this section, we describe our approach to select demonstrations in both sub tasks.

3.1 Selecting within Mixture of Tasks

In this scenario, we develop influence-based methods for demonstration selection. Specifically, for each validation example, we compute influence values to identify the most impactful examples from a pool of training examples containing a mixture of tasks. Two approaches are employed to calculate these influence scores:

- A **surrogate-model** based method, where a lightweight surrogate model such as RoBERTa (Liu, 2019) is fine-tuned on the candidate demonstrations to compute influence.
- A **pretrained-gradient** based method where the samples are passed through the LLM itself. We then compute IFs using the extracted gradients.

Formally, for each validation example $(x_{\text{val}}, y_{\text{val}})$, we compute the influence of each training example $(x_i, y_i) \in D_{\text{train}}$, where D_{train} is the set of the training examples. The influence score $I((x_i, y_i), (x_{\text{val}}, y_{\text{val}}))$ quantifies the effect of (x_i, y_i) on the loss function evaluated at $(x_{\text{val}}, y_{\text{val}})$. Using these computed influence values, we select the top k most influential demonstrations.

We compare two versions of computing the IF after extracting the gradient, DataInf (Kwon et al., 2023) and TracIn (Pruthi et al., 2020). DataInf uses an easy-to-compute closed-form expression, leading to better computational and memory complexities than other IF methods. TracIn traces how the loss on the test point changes during the training process simply using an inner product of training and validation set gradients. Since it does not compute the Hessian matrix, it is faster than DataInf, but at the cost of lower estimation performance.

Additionally, we compare the influence-only methods with well-performing ICL approaches BertScore-Recall (BSR; Gupta et al. 2023a) and Cosine Similarity (Reimers, 2019).[3] These methods excel at capturing semantic similarity between validation and training examples. We also compare with a performant sparse information retrieval baseline algorithm, BM25 (Jones et al., 2000).

Lastly, we combine the previously described approaches by implementing a two-stage selection process. First, we perform an initial pruning of the demonstration pool using either BSR or Cosine Similarity. Specifically, for a given number of desired demonstrations k, we prune the dataset to select $2k$ candidates from the original set of labeled examples $\{(x_i, y_i)\}_{i=1}^{N}$.

We then apply the IF-based methods to re-rank these remaining examples based on their influence on the validation loss. The final selection of k in-context demonstrations is performed by selecting the top k examples from the re-ranked subset.

3.2 Selecting Noisy Demonstrations

In this setting, we utilize IFs to identify noisy samples within the dataset. Formally, let

[3]We use the implementation from Gupta et al. (2023a,b).

$D = \{(x_i, y_i)\}_{i=1}^{N}$ represent the training dataset. First, we employ IFs to prune the dataset by detecting and removing noisy examples, following which the top k in-context demonstrations are selected using either BSR or Cosine Similarity.[4]

Additionally, we construct approaches that combine the influence values with the BSR or Cosine Similarity scores. To do so, both the influence values and similarity scores are min-max normalized, resulting in scores scaled between 0 and 1. We then reweigh the scores using a linear combination of the normalized values. Let α and β represent the weights assigned to the influence values and the similarity scores, respectively, where $\alpha + \beta = 1 \quad \text{and} \quad 0 < \alpha, \beta < 1$, the final combined score for each training example is:

$$\text{Score}(x_i, y_i) = \alpha \cdot I((x_i, y_i), (x_{\text{val}}, y_{\text{val}})) + \beta \cdot S(x_i, y_i),$$

where $I((x_i, y_i), (x_{\text{val}}, y_{\text{val}}))$ is the influence value and $S(x_i, y_i)$ represents either the BertScore (Zhang et al., 2019) or Cosine Similarity for the training example (x_i, y_i). The top k examples with the highest combined scores are selected as demonstrations.[5]

In this setting, we compute influence values using our surrogate model approach. In addition to using DataInf, we also conduct influence experiments using the LiSSA IF method which is a second-order method to compute the inverse Hessian vector product (Agarwal et al., 2017; Koh and Liang, 2017). Although LiSSA is generally computationally expensive (Kwon et al., 2023), we prioritize it over TracIn owing to its greater performance in detecting mislabeled samples, as the computational overhead is incurred only once in this setting.

4 Experiments

Here, we expand upon our experimental setup to conduct the experiments and analyze the results.

4.1 Experimental Setup

We discuss our, dataset details and model used to conduct the experiments.

Evaluation Data. For *Mixture of Tasks*, we collect a generalized pool of examples from different tasks such that the input x and output y pairs $\{(x_i, y_i)\}_{i=1}^{k}$ do not necessarily correspond to the same task as the test input x_{test}. The evaluation task pool contains three samples each from 28 different tasks from MMLU (Hendrycks et al., 2020), BigBench (Srivastava et al., 2022), StrategyQA (Geva et al., 2021) and CommonsenseQA (Talmor et al., 2018). We evaluate the ICL accuracy, using this train set, on 12 different tasks from MMLU and BigBench.

For *Noisy Demonstrations*, we employ the noisy dataset framework from Kwon et al. (2023). In their work, the four binary classification GLUE datasets (Wang, 2018) MRPC, QQP, QNLI, and SST2 are utilized. To simulate a scenario where a portion of the data samples are noisy, 20% of the training data samples are randomly selected and their labels are flipped. We use these noisy datasets as the candidate pool in our experiments and evaluate the ICL accuracy.

Base LLM. In Mixture of Tasks, for $k = 3$ shots, we conduct ICL experiments on Llama-2-13b-chat (Touvron et al., 2023), Mistral-7b-v0.3 (Jiang et al., 2023) and Zephyr-7b-beta (Tunstall et al., 2023). For $k = 5$ shots we conduct experiments on Llama-2-13b-chat. We extend on the framework designed by Gupta et al. (2023a,b). The temperature is set to 0 for inference. For Noisy ICL, we conduct experiments on Llama-2-13b-chat. All of our experiments run on 8×NVIDIA RTX 6000 Ada GPUs.

4.2 Method and Baseline Configurations

Here we expand on the methods and baselines we use for our experiments in both settings.

Mixture of Tasks. We construct 4 IF-only methods. 2 based on the Surrogate Model based approach, SUR and 2 based on the Pretrained LLM weights based approach, PRE. We test Data-Inf and TracIn based versions of these approaches, namely, Surrogate Model-DataInf SUR_D, Surrogate Model-TracIn SUR_T, Pretrained Model-DataInf PRE_D and Pretrained Model-TracIn PRE_T. As mentioned before, SUR_D and SUR_T use RoBERTa as the surrogate model, whereas PRE_D and PRE_T use Llama2-13b-chat as the pretrained LLM.

Additionally, we test traditional semantic approaches, such as BSR and Cosine Similarity (COS), as well as retrieval based approaches, such as BM25, as baselines. Finally, we test the combination of the aforementioned traditional and IF methods as well.[6]

[4]We will refer to this approach as IF Pruning.

[5]We will refer to this approach as IF Averaging.

[6]Specifically, $\textbf{MODEL}_{[\textbf{IF},\textbf{SEL}]}$, where $\textbf{MODEL} \in \{\textbf{SUR}, \textbf{PRE}\}$, $\textbf{IF} \in \{\textbf{D}, \textbf{T}\}$, and $\textbf{SEL} \in \{\textbf{COS}, \textbf{BSR}\}$.

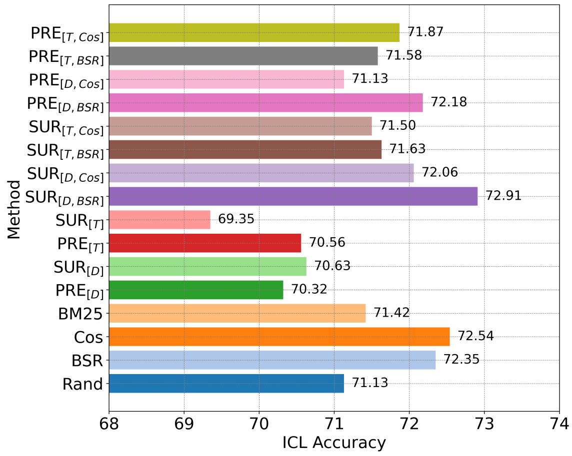

Figure 2: Average performance of different demonstration selection methods across 3 LLMs for $k = 3$ shots.

Noisy Demonstrations. As elaborated in Section 3.2, we explore two approaches, IF Pruning and IF Averaging, for the task of selecting the best demonstrations. We only use the surrogate model-based IF method in this setting, and we employ an additional method of computing IFs, LiSSA (Koh and Liang, 2017). We experiment with different levels of pruning and IF weights (α) as hyperparameters, namely 10% pruning and 0.5α. Furthermore, we also create a random pruning baseline for BSR and Cosine Similarity as well.[7]

For baselines, we again compare our IF based approaches with BSR, Cosine Similarity and BM25.

4.3 Results on Mixture of Tasks

We present the results on Mixture of Tasks in Figure 2. Additional results for varying the number of shots (k) and multiple LLMs are provided in Appendix B.1. Further, results for the alternative TracIn IF method are provided in Appendix B.2. Results for Pretrained Gradients combined with BertScore and Cosine Similarity are presented in Appendix B.3. We also present results on using DeBERTa (He et al., 2021a,b) as an alternative surrogate model in Appendix B.4.

Combining Surrogate Model DataInf with BertScore results in the best performance. As can be observed in Figure 2 and Table 7, the $\text{SUR}_{[\text{D,BSR}]}$ method has the highest average performance across the tasks, in both 3 and 5 shots. This shows the benefit of combining IF with BertScore as performance increased by 0.56 in $k = 3$ shots and by 1.52 in $k = 5$ shots. The results also show that the maximal benefit of IF methods is gained in combination with the semantic similarity methods. This is due to the fact that IF can leverage the model's inductive bias to re-rank the retrieved demonstrations effectively, but the initial $2k$ pruning via BSR is critical to shorten the candidate pool to demonstrations that are semantically relevant enough. However, it is important to note that, given the inclusion of only three shots in the prompt—where the overwhelming majority of demonstrations are unrelated to the test task—achieving significant improvements remains challenging.

Surrogate Models outperform Pretrained Gradients. We see that surrogate models outperformed Pretrained Gradients in demonstration selection for the Mixture of Tasks setting in both the $k = 3$ and $k = 5$ shots. The fine-tuning of the surrogate model leads it to better capture the test task affinity of the demonstration pool.

DataInf is better than TracIn as an IF method. The speed gains of TracIn come at a cost of performance as the DataInf method of IF computation routinely outperformed TracIn. TracIn likely underperforms because it does not utilize critical second order gradient information since the Hessian $H(\theta^*)$ is assumed to be the identity matrix. This trend has also been observed in past work on IF methods (Chhabra et al., 2024a).

Qualitative Analysis. Finally, to understand the unique benefits provided by IFs, we present a qualitative analysis examining the types of shots selected by our method in Appendix B.5. We see that even though BSR selects more semantically relevant samples, $\text{SUR}_{[\text{D,BSR}]}$ shots assist in guiding the model toward the correct answer by providing examples that promote more structured reasoning.

4.4 Results on Noisy ICL

In this section we provide analysis on the aforementioned General Noisy ICL setting. We also conduct two ablations on varying IF α weight in IF Averaging and varying the Noise level in the datasets.

IF Averaging works better than other baselines. Table 1 and Figure 3 clearly show that doing a weighted average between the surrogate model IF and both Cosine and BertScore leads to performance boosts. Atleast one and if not both of the highest performing methods in each of the datasets we tested were from the averaging method. We

[7]Formally, $\textbf{METHOD}[\textbf{MODEL}_{\textbf{IF},\textbf{SEL}}]$, where $\textbf{METHOD} \in \{\textbf{PRU}, \textbf{AVG}\}$, $\textbf{MODEL} \in \{\textbf{SUR}, \textbf{RAND}\}$, $\textbf{IF} \in \{\textbf{D}, \textbf{L}\}$, and $\textbf{SEL} \in \{\textbf{COS}, \textbf{BSR}\}$.

Table 1: ICL Accuracy across MRPC, QNLI, SST2, and QQP datasets using different methods for Noisy ICL, with 20% noise added to the datasets. The top 2 performers for each dataset are in bold.

	Method	MRPC	QNLI	SST2	QQP
	RAND	70.4	69.6	86.2	70.9
	BSR	71.3	74.6	80.4	71.4
	COS	72.3	68.2	82.6	**73.2**
	BM25	70.6	67.6	88.0	71.0
PRU-0.1	RAND[Cos]	70.1	68.4	87.0	69.4
PRU-0.1	SUR[D,Cos]	69.4	69.8	84.0	71.8
PRU-0.1	SUR[L,Cos]	68.9	68.2	86.8	70.8
PRU-0.1	RAND[BSR]	71.1	65.2	86.4	68.4
PRU-0.1	SUR[D,BSR]	70.6	68.0	82.0	71.0
PRU-0.1	SUR[L,Cos]	70.1	67.4	88.8	68.2
AVG-0.5	SUR[D,Cos]	**75.5**	**74.8**	89.8	67.8
AVG-0.5	SUR[L,Cos]	70.6	**75.8**	86.4	73.0
AVG-0.5	SUR[D,BSR]	**74.3**	69.6	**90.6**	**73.8**
AVG-0.5	SUR[L,BSR]	73.3	73.4	**93.6**	69.2

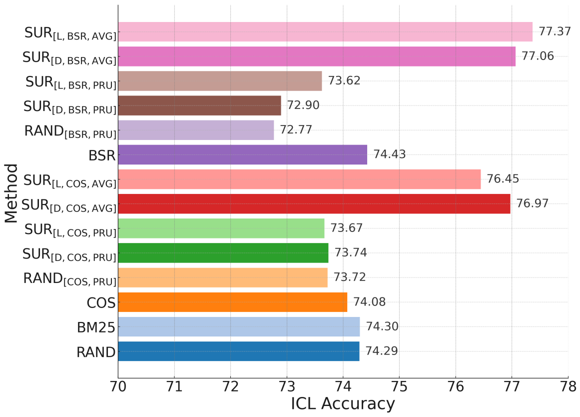

Figure 3: Average performance of the baselines across the 4 datasets.

see that LiSSA and DataInf are similarly effective, with DataInf being more computationally efficient.

Pruning hurts not helps performance. We can see that pruning actively hurts performance as Figure 3 shows that all 3 types of BertScore pruning and all 3 types of Cosine pruning had lower average scores than BertScore and Cosine Similarity. This might be due to the fact that we are removing potentially helpful samples from the demonstration pool, even if they might have noisy labels.

We further provide results on varying the noise levels in the datasets in C.1, varying the hyperparameters we tested in C.2 and an experiment analyzing the effectiveness of IF's in detecting noisy demonstrations in C.3.

4.5 Computational Complexity

We present the worst case time complexity (for inference) for our methods and related baselines in Table 2. As can be observed, our methods are comparable, if not more efficient than the other baselines. Note that the SUR methods require an additional fine-tuning step on a smaller surrogate model before the gradients are extracted, which the PRE methods do not. Furthermore, note that TracIn as an influence method is much faster than Hessian-based approaches (e.g. DataInf) as it assumes that the Hessian is the identity matrix. While this leads to more efficient influence computation, it comes at the cost of lower estimation performance, as our results with TracIn also show. Additionally, we present the maximum GPU memory consumption while performing demonstration selection in Appendix D.

Table 2: Computational complexity for each test sample at inference where N is #demonstration samples, p is #model parameters, d is embedding size, K is the max length among all candidates, L is the length (in tokens) of the test input, Z is #ngrams

Method	Time Complexity
BSR	$O(NLKd)$
COS	$O(Nd)$
BM25	$O(NZ)$
PRE$_D$	$O(Np)$
SUR$_D$	$O(Np)$
PRE$_T$	$O(Np)$
SUR$_T$	$O(Np)$
BSR COMBINED METHODS	$O(NLKd) + O(Np)$
COS COMBINED METHODS	$O(Nd) + O(Np)$

4.6 Scalability

Scalability to Large Models. We compare the time it takes to extract test set gradients and compute influence scores. We compute IF via the DataInf method, comparing RoBERTa-Large (125 million parameters), Llama2-13b-chat (13 billion parameters), and Llama2-70b-chat (70 billion parameters) on the MMLU-moral-disputes dataset with 200 test samples.

Table 3: Time taken for extracting test gradients and computing IF across different models.

Model	Test Gradients (s)	Computing IF (s)
RoBERTa	7.447	35.72
Llama-2-13b-chat	68.5	4.69
Llama-2-70b-chat	257.64	8.81

The relationship between model size and inference time grows sublinearly, with time increasing at roughly the square root of the model size. We also see that it takes longer to compute the IF in the RoBERTa model due to the fine-tuning process.

Additionally, the time required to compute IF using the TracIn method on Llama-2-13b-chat is just $\mathbf{3.576 \times 10^{-6}}$ seconds. This highlights the significant speed advantage offered by TracIn. We present additional results for scalability to large datasets in Appendix E.

We would like to emphasize that practitioners have the flexibility to choose between our models and methods based on their specific needs. If computational efficiency is the priority, the significantly faster surrogate model approach can be used. Conversely, if high accuracy is desired and compute is not a concern, a fine-tuned LLM is a better alternative.

5 Related Work

In-Context Learning (ICL). Following the scaling of model sizes and learning resources (Mann et al., 2020; Chowdhery et al., 2023; Touvron et al., 2023), LLMs have gained emergent abilities for efficient inference-time adaptation via ICL (Mann et al., 2020). However, ICL is critically sensitive to demonstration pool examples (An et al., 2023; Liu et al., 2021a; Zhang et al., 2022) and selection strategies (Rubin et al., 2021; Mavromatis et al., 2023). One line of work studies example scoring and retrieval, utilizing model-agnostic heuristic metrics like perplexity (Gonen et al., 2022), mutual information (Sorensen et al., 2022), semantic similarity (Liu et al., 2021a; Gupta et al., 2023b), etc. to select demonstrations. Another line of work optimizes selection based on empirically verified desirable features a priori, e.g. diversity (Su et al., 2022; Ye et al., 2023), coverage (Gupta et al., 2023a), etc. However, prior work assumes that the demonstration distribution is aligned with task distribution, which is not always the case (Chatterjee et al., 2024). Our work serves as a first to investigate ICL demonstration selection in the task and dataset quality shifts in the ICL settings.

Influence Functions. *Influence functions* (IFs) comprise a set of methods from robust statistics (Hampel, 1974; Cook and Weisberg, 1982) that have been recently proposed for deep learning data valuation and can provide a conceptual link that traces model performance to samples in the training set. For gradient-based models trained using empirical risk minimization, IFs can be used to approximate sample influence without requiring actual leave-one-out retraining. For deep learning models, the seminal work by Koh and Liang (2017) utilized a Taylor-series approximation and LiSSA optimization (Agarwal et al., 2017) to compute sample influences. Follow-up works such as Representer Point (Yeh et al., 2018) and Hydra (Chen et al., 2021) sought to improve IF performance for deep learning models, constrained to vision applications. More recently, efficient influence estimation methods such as DataInf (Kwon et al., 2023), Arnoldi iteration (Schioppa et al., 2022), and Kronecker-factored approximation curvature (Grosse et al., 2023) have been proposed which can be employed for larger generative language models, such as LLMs. Some other simpler IF approaches simply consider the gradients directly as a measure of influence (Pruthi et al., 2020; Charpiat et al., 2019), followed by some ensemble strategies (Bae et al., 2024; Kim et al., 2024). Recent work has also found that *self-influence* only on the training set can be a useful measure for detecting sample influence (Bejan et al., 2023; Thakkar et al., 2023).

IFs have been utilized with great success in a number of application scenarios (e.g. classification (Chhabra et al., 2024a; Koh and Liang, 2017), generative models (Kwon et al., 2023; Schioppa et al., 2022; Grosse et al., 2023), active learning (Chhabra et al., 2024b; Liu et al., 2021b), etc.). Moreover, while some recent works have considered using influence for selecting direct demonstrations (Nguyen and Wong, 2023; Van et al., 2024), neither of them has consider their effect on inductive bias selection in the indirect ICL setting, which is the focus of our work.

6 Conclusion

We formalize a new paradigm for generalized In-Context Learning, which we term *Indirect In-Context Learning*. We analyze two different real-world Indirect ICL settings and propose effective demonstration selection strategies for these scenarios. We explore using Influence Functions (IFs) to leverage the informativeness of the samples in the demonstration pool and the models' task inductive bias. We find that combining a surrogate model-based IF approach with BertScore performs better when there are an overwhelming majority of irrelevant tasks in the candidate pool. We also find that reweighting the surrogate model-based IF scores with traditional metric scores can be helpful in the case where noisy demonstrations are present. Future work will aim to augment the Pretrained Gradient approach by using better/larger LLMs or finetuning the LLMs.

Limitations

The main limitation of Influence Functions is that they are costly to compute, especially for large datasets and LLMs with lots of parameters. This is why we opted to fine-tune a smaller model such as RoBERTa and use pretrained LLMs for our methods. Further performance gains can be attained at the cost of computational speed if fine-tuned LLMs are employed instead. As research on influence estimation methods for LLMs is currently ongoing, faster influence functions developed in the future can also be utilized with our methods for highly efficient and accurate ICL performance.

References

Naman Agarwal, Brian Bullins, and Elad Hazan. 2017. Second-order stochastic optimization for machine learning in linear time. *Journal of Machine Learning Research*, 18(116):1–40.

Shengnan An, Zeqi Lin, Qiang Fu, Bei Chen, Nanning Zheng, Jian-Guang Lou, and Dongmei Zhang. 2023. How do in-context examples affect compositional generalization? *arXiv preprint arXiv:2305.04835*.

Cem Anil, Yuhuai Wu, Anders Andreassen, Aitor Lewkowycz, Vedant Misra, Vinay Ramasesh, Ambrose Slone, Guy Gur-Ari, Ethan Dyer, and Behnam Neyshabur. 2022. Exploring length generalization in large language models. *Advances in Neural Information Processing Systems*, 35:38546–38556.

Juhan Bae, Wu Lin, Jonathan Lorraine, and Roger Grosse. 2024. Training data attribution via approximate unrolled differentation. *arXiv preprint arXiv:2405.12186*.

Irina Bejan, Artem Sokolov, and Katja Filippova. 2023. Make every example count: On the stability and utility of self-influence for learning from noisy nlp datasets. In *Conference on Empirical Methods in Natural Language Processing*.

Guillaume Charpiat, Nicolas Girard, Loris Felardos, and Yuliya Tarabalka. 2019. Input similarity from the neural network perspective. *Advances in Neural Information Processing Systems*, 32.

Anwoy Chatterjee, Eshaan Tanwar, Subhabrata Dutta, and Tanmoy Chakraborty. 2024. Language models can exploit cross-task in-context learning for data-scarce novel tasks. *arXiv preprint arXiv:2405.10548*.

Yuanyuan Chen, Boyang Li, Han Yu, Pengcheng Wu, and Chunyan Miao. 2021. Hydra: Hypergradient data relevance analysis for interpreting deep neural networks. In *AAAI Conference on Artificial Intelligence*.

Anshuman Chhabra, Bo Li, Jian Chen, Prasant Mohapatra, and Hongfu Liu. 2024a. Outlier gradient analysis: Efficiently improving deep learning model performance via hessian-free influence functions. *arXiv preprint arXiv:2405.03869*.

Anshuman Chhabra, Peizhao Li, Prasant Mohapatra, and Hongfu Liu. 2024b. " what data benefits my classifier?" enhancing model performance and interpretability through influence-based data selection. In *The Twelfth International Conference on Learning Representations*.

Sang Keun Choe, Hwijeen Ahn, Juhan Bae, Kewen Zhao, Minsoo Kang, Youngseog Chung, Adithya Pratapa, Willie Neiswanger, Emma Strubell, Teruko Mitamura, et al. 2024. What is your data worth to gpt? llm-scale data valuation with influence functions. *arXiv preprint arXiv:2405.13954*.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.

A Conneau. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

R Dennis Cook and Sanford Weisberg. 1980. Characterizations of an empirical influence function for detecting influential cases in regression. *Technometrics*, 22(4):495–508.

R Dennis Cook and Sanford Weisberg. 1982. *Residuals and influence in regression*. New York: Chapman and Hall.

Andrew Drozdov, Nathanael Schärli, Ekin Akyürek, Nathan Scales, Xinying Song, Xinyun Chen, Olivier Bousquet, and Denny Zhou. 2022. Compositional semantic parsing with large language models. In *The Eleventh International Conference on Learning Representations*.

Vitaly Feldman and Chiyuan Zhang. 2020. What neural networks memorize and why: Discovering the long tail via influence estimation. *Advances in Neural Information Processing Systems*, 33:2881–2891.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361.

Hila Gonen, Srini Iyer, Terra Blevins, Noah A Smith, and Luke Zettlemoyer. 2022. Demystifying prompts in language models via perplexity estimation. *arXiv preprint arXiv:2212.04037*.

Roger Grosse, Juhan Bae, Cem Anil, Nelson Elhage, Alex Tamkin, Amirhossein Tajdini, Benoit Steiner, Dustin Li, Esin Durmus, Ethan Perez, et al. 2023. Studying large language model generalization with influence functions. *arXiv preprint arXiv:2308.03296*.

Han Guo, Nazneen Fatema Rajani, Peter Hase, Mohit Bansal, and Caiming Xiong. 2020. Fastif: Scalable influence functions for efficient model interpretation and debugging. *arXiv preprint arXiv:2012.15781*.

Shivanshu Gupta, Matt Gardner, and Sameer Singh. 2023a. Coverage-based example selection for in-context learning. *arXiv preprint arXiv:2305.14907*.

Shivanshu Gupta, Clemens Rosenbaum, and Ethan R Elenberg. 2023b. Gistscore: Learning better representations for in-context example selection with gist bottlenecks. *arXiv preprint arXiv:2311.09606*.

Kishaloy Halder, Alan Akbik, Josip Krapac, and Roland Vollgraf. 2020. Task-aware representation of sentences for generic text classification. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3202–3213.

Frank R Hampel. 1974. The influence curve and its role in robust estimation. *Journal of the american statistical association*, 69(346):383–393.

Xiaochuang Han, Byron C Wallace, and Yulia Tsvetkov. 2020. Explaining black box predictions and unveiling data artifacts through influence functions. *arXiv preprint arXiv:2005.06676*.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021a. Debertav3: Improving deberta using electra-style pretraining with gradient-disentangled embedding sharing.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021b. Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*.

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*.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

K Sparck Jones, Steve Walker, and Stephen E. Robertson. 2000. A probabilistic model of information retrieval: development and comparative experiments: Part 2. *Information processing & management*, 36(6):809–840.

SungYub Kim, Kyungsu Kim, and Eunho Yang. 2024. Gex: A flexible method for approximating influence via geometric ensemble. *Advances in Neural Information Processing Systems*, 36.

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pages 1885–1894. PMLR.

Shuming Kong, Yanyan Shen, and Linpeng Huang. 2021. Resolving training biases via influence-based data relabeling. In *International Conference on Learning Representations*.

Jannik Kossen, Yarin Gal, and Tom Rainforth. 2024. In-context learning learns label relationships but is not conventional learning. In *The Twelfth International Conference on Learning Representations*.

Yongchan Kwon, Eric Wu, Kevin Wu, and James Zou. 2023. Datainf: Efficiently estimating data influence in lora-tuned llms and diffusion models. *arXiv preprint arXiv:2310.00902*.

Bangzheng Li, Ben Zhou, Xingyu Fu, Fei Wang, Dan Roth, and Muhao Chen. 2024. Famicom: Further demystifying prompts for language models with task-agnostic performance estimation. *arXiv preprint arXiv:2406.11243*.

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

Yinhan Liu. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Zhuoming Liu, Hao Ding, Huaping Zhong, Weijia Li, Jifeng Dai, and Conghui He. 2021b. Influence selection for active learning. In *IEEE/CVF International Conference on Computer Vision*.

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

Ben Mann, N Ryder, M Subbiah, J Kaplan, P Dhariwal, A Neelakantan, P Shyam, G Sastry, A Askell, S Agarwal, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

R Douglas Martin and Victor J Yohai. 1986. Influence functionals for time series. *The annals of Statistics*, pages 781–818.

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 preprint arXiv:2310.20046*.

Sewon Min, Mike Lewis, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2021. Metaicl: Learning to learn in context. *arXiv preprint arXiv:2110.15943*.

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

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

Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. 2020. Estimating training data influence by tracing gradient descent. *Advances in Neural Information Processing Systems*, 33:19920–19930.

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

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2021. Learning to retrieve prompts for in-context learning. *arXiv preprint arXiv:2112.08633*.

Andrea Schioppa, Polina Zablotskaia, David Vilar, and Artem Sokolov. 2022. Scaling up influence functions. In *AAAI Conference on Artificial Intelligence*.

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

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. 2022. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*.

Hongjin Su, Jungo Kasai, Chen Henry Wu, Weijia Shi, Tianlu Wang, Jiayi Xin, Rui Zhang, Mari Ostendorf, Luke Zettlemoyer, Noah A Smith, et al. 2022. Selective annotation makes language models better few-shot learners. *arXiv preprint arXiv:2209.01975*.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2018. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *arXiv preprint arXiv:1811.00937*.

Megh Thakkar, Tolga Bolukbasi, Sriram Ganapathy, Shikhar Vashishth, Sarath Chandar, and Partha Talukdar. 2023. Self-influence guided data reweighting for language model pre-training. In *Conference on Empirical Methods in Natural Language Processing*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, Nathan Sarrazin, Omar Sanseviero, Alexander M. Rush, and Thomas Wolf. 2023. Zephyr: Direct distillation of lm alignment.

Minh-Hao Van, Xintao Wu, et al. 2024. In-context learning demonstration selection via influence analysis. *arXiv preprint arXiv:2402.11750*.

Alex Wang. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Hao Wang, Berk Ustun, and Flavio Calmon. 2019. Repairing without retraining: Avoiding disparate impact with counterfactual distributions. In *International Conference on Machine Learning*, pages 6618–6627. PMLR.

Xinyi Wang, Wanrong Zhu, Michael Saxon, Mark Steyvers, and William Yang Wang. 2024. Large language models are latent variable models: Explaining and finding good demonstrations for in-context learning. *Advances in Neural Information Processing Systems*, 36.

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

Tongshuang Wu, Haiyi Zhu, Maya Albayrak, Alexis Axon, Amanda Bertsch, Wenxing Deng, Ziqi Ding, Bill Guo, Sireesh Gururaja, Tzu-Sheng Kuo, et al. 2023. Llms as workers in human-computational algorithms? replicating crowdsourcing pipelines with llms. *arXiv preprint arXiv:2307.10168*.

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

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. 2024. Less: Selecting influential data for targeted instruction tuning. *arXiv preprint arXiv:2402.04333*.

Yan Yan, Rómer Rosales, Glenn Fung, Ramanathan Subramanian, and Jennifer Dy. 2014. Learning from multiple annotators with varying expertise. *Machine learning*, 95:291–327.

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

Chih-Kuan Yeh, Joon Kim, Ian En-Hsu Yen, and Pradeep K Ravikumar. 2018. Representer point selection for explaining deep neural networks. *Advances in Neural Information Processing Systems*.

Wenpeng Yin, Muhao Chen, Ben Zhou, Qiang Ning, Kai-Wei Chang, and Dan Roth. 2023. Indirectly supervised natural language processing. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 6: Tutorial Abstracts)*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Yiming Zhang, Shi Feng, and Chenhao Tan. 2022. Active example selection for in-context learning. *arXiv preprint arXiv:2211.04486*.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 12697–12706. PMLR.

Zhaowei Zhu, Zihao Dong, and Yang Liu. 2022. Detecting corrupted labels without training a model to predict. In *International conference on machine learning*, pages 27412–27427. PMLR.

Appendix

A Integration to Practical Workflows

We provide several use cases of Indirect ICL in real-world scenarios. Here are a few examples:

A.1 Practical Applications of Indirect ICL

- **Enhancing prompt performance at test time:** If an LLM service provider needs to use in-context learning (ICL) to improve prompt performance during test time, they may not know the precise task beforehand or during inference (e.g., a novel task requested by a user in real-time). Indirect ICL and our proposed methods can improve performance by selecting relevant demonstrations from a task-agnostic pool of labeled data (i.e., the MoT setting), ensuring the model can adapt to various scenarios even when task-specific labeled samples (direct supervision) are not available.

- **Medical diagnosis:** Indirect ICL can be used to diagnose rare medical conditions based on symptoms. Since such conditions are rare, demonstrations for these specific cases are often unavailable. However, the model can learn diagnostic reasoning patterns from more common conditions with overlapping symptoms, improving accuracy for the rare cases.

- **Code generation for obscure programming languages:** Indirect ICL can aid in generating code for rarely-used or proprietary programming languages. Demonstrations from code generation tasks in related languages with similar structures can be leveraged, enabling the model to generalize and perform well in these low-resource scenarios.

- **Ideology Estimation from Underrepresented Contexts:** We can use our paradigm to estimate political ideology, or any other sort of text classification, from text in an underrepresented cultural or linguistic context. We can use demonstrations from ideology estimation in well-represented contexts such as Western political texts. The can transfer learned associations between linguistic cues and ideological stances, adapting them to the new context.

These examples highlight just a few of the practical applications of indirect ICL, particularly in low-resource settings.

B Indirect ICL Results for LLM based Influence

B.1 Full Results for Mixture of Tasks

Following are the full results for the Mixture of Tasks setting. For $k = 3$ shots in Tables 4, 5, and 6. For $k = 5$ shots in Table 7.

B.2 TracIn Results

Here we provide results for the TracIn method of Influence Computation for $k = 3$ shots in Tables 8, 9, and 10. We also provide results for $k = 5$ shots in Table 11.

B.3 Pretrained Gradient Results

Here we provide results for pretrained gradients method of computing IF. These can be found in for $k = 3$ shots in Tables 12, 13 and 14 and for $k = 5$ shots in Table 15.

B.4 Results using a different surrogate model

We present results where DeBERTa-v3-Large replaces RoBERTa-Large as the surrogate model. Evaluated on Llama2-13B-Chat with $k = 3$ shots. We compare the best-performing baseline $\textsc{Sur}_{[\text{D,BSR}]}$ with BSR in Table 16.

The results indicate that the DeBERTa surrogate model outperforms BSR. However, it is important to note that, given the inclusion of only three shots in the prompt—where the overwhelming majority of demonstrations are unrelated to the test task—achieving significant improvements remains challenging.

Table 4: Performance across different datasets and demonstration selection methods with $k = 3$ shots. The datasets are sampled from sub-tasks of the MMLU and BigBench datasets for Llama-2-13b-chat.

Dataset	RAND	BSR	COS	BM25	PRE$_D$	SUR$_D$	SUR$_{[D,BSR]}$	SUR$_{[D,Cos]}$
medical-genetics	80.60	87.00	84.00	79.00	76.00	81.00	86.00	82.00
prof-psychology	68.30	70.00	73.00	65.50	65.00	68.50	72.00	68.50
formal-logic	60.16	59.52	60.32	58.73	61.11	59.52	57.14	55.56
moral-disputes	81.00	78.00	79.50	80.50	78.50	76.00	80.50	76.50
public-relations	72.18	79.09	80.91	73.64	71.82	75.45	79.09	79.09
comp-security	76.80	76.00	80.00	76.00	76.00	80.00	76.00	76.00
astronomy	80.26	80.26	78.95	80.92	74.34	79.61	80.26	78.95
abstract-algebra	57.00	58.00	62.00	55.00	57.00	47.00	72.00	72.00
nutrition	75.50	77.50	79.00	78.00	77.50	77.00	79.50	81.50
high-school-biology	76.70	76.50	78.00	76.50	73.50	79.00	80.50	76.50
formal-fallacies	47.25	52.00	50.00	49.50	47.00	56.50	47.50	50.00
tracking-3	40.20	44.00	39.00	45.00	40.00	37.00	43.50	39.00
Average	68.00	69.82	**70.39**	68.19	66.48	68.04	**71.16**	69.63

Table 5: Performance across different datasets and demonstration selection methods with $k = 3$ shots. The datasets are sampled from sub-tasks of the MMLU and BigBench datasets for Mistral-7b-v3

Dataset	RAND	BSR	COS	BM25	PRE$_D$	SUR$_D$	SUR$_{[D,BSR]}$	SUR$_{[D,Cos]}$
medical-genetics	88.25	88.00	91.00	89.00	88.00	86.00	87.00	89.00
prof-psychology	84.50	84.00	84.50	82.00	85.00	83.00	85.50	84.00
formal-logic	66.47	66.67	69.05	65.08	63.49	66.67	68.25	69.05
moral-disputes	87.00	85.00	85.50	87.00	88.5	86.00	87.00	87.50
public-relations	83.41	82.73	81.82	84.55	84.55	83.64	84.55	81.82
comp-security	87.25	83.00	89.00	88.00	90.00	88.00	86.00	90.00
astronomy	87.34	88.82	89.47	88.16	84.87	86.18	86.18	86.18
abstract-algebra	57.75	63.00	60.00	60.00	59.00	50.00	64.00	64.00
nutrition	84.75	86.50	87.50	83.00	83.00	83.50	88.50	87.00
high-school-biology	85.63	84.00	86.50	85.00	87.00	84.00	87.50	87.00
formal-fallacies	49.63	53.50	50.00	53.50	48.00	46.50	52.50	53.50
tracking-3	46.38	49.00	49.00	49.50	46.50	39.00	48.50	45.50
Average	75.70	76.19	**76.95**	76.23	75.66	73.54	**77.13**	77.05

Table 6: Performance across different datasets and demonstration selection methods with $k = 3$ shots. The datasets are sampled from sub-tasks of the MMLU and BigBench datasets for Zephyr-7b-beta

Dataset	RAND	BSR	COS	BM25	PRE$_D$	SUR$_D$	SUR$_{[D,BSR]}$	SUR$_{[D,Cos]}$
medical-genetics	79.50	80.00	77.00	76.00	78.00	82.00	76.00	78.00
prof-psychology	74.50	74.00	74.50	74.50	72.00	74.50	73.00	74.00
formal-logic	69.44	65.87	65.87	65.08	66.67	71.43	65.87	61.11
moral-disputes	77.63	78.50	78.00	76.50	75.00	78.00	78.50	77.00
public-relations	75.00	80.91	72.73	73.64	76.36	75.45	76.36	76.36
comp-security	76.00	73.00	77.00	78.00	75.00	77.00	79.00	79.00
astronomy	79.77	81.58	80.26	80.26	74.34	80.92	79.61	82.24
abstract-algebra	52.00	53.00	53.00	51.00	51.00	50.00	62.00	55.00
nutrition	75.63	77.00	79.50	75.50	74.00	74.50	75.50	77.50
high-school-biology	77.63	81.00	80.50	79.50	78.50	80.00	78.50	78.00
formal-fallacies	49.75	57.50	55.50	56.50	52.50	55.00	51.50	46.50
tracking-3	49.25	49.50	49.50	51.50	52.50	45.00	49.50	49.50
Average	69.68	**70.99**	70.28	69.83	68.83	70.31	**70.45**	69.51

Table 7: Performance across different datasets and demonstration selection methods with $k = 5$ shots for Llama-2-13b-chat

Dataset	RAND	BSR	COS	BM25	PRE$_D$	SUR$_D$	SUR$_{[D,BSR]}$	SUR$_{[D,Cos]}$
medical-genetics	80.00	86.00	81.00	81.00	84.00	80.00	84.00	83.00
prof-psychology	71.00	71.00	73.50	66.00	70.00	68.50	77.00	71.00
formal-logic	62.70	59.52	57.94	56.35	58.73	68.25	62.70	61.90
moral-disputes	79.50	77.50	81.00	81.00	82.00	81.00	81.50	81.50
public-relations	70.00	78.18	81.82	76.36	70.91	77.82	80.91	78.18
comp-security	75.00	78.00	82.00	77.00	76.00	77.00	81.00	77.00
astronomy	78.95	85.53	82.89	80.92	80.92	82.89	84.87	81.58
abstract-algebra	52.00	63.00	62.00	58.00	63.00	55.00	67.00	65.00
nutrition	71.50	81.00	80.00	78.00	76.00	78.00	79.00	81.00
high-school-biology	73.00	79.00	80.00	76.50	77.00	79.00	81.50	75.50
formal-fallacies	46.50	48.50	48.00	50.00	47.50	43.50	53.00	43.50
tracking-3	36.50	49.00	47.00	46	42.50	52.00	42.00	39.00
Average	66.38	71.35	**71.42**	68.93	69.04	70.24	**72.87**	69.84

Table 8: Performance across different datasets and different TracIn Influence methods with $k = 3$ shots for Llama2-13b-chat

Dataset	PRE_{T}	SUR_{T}	$\text{SUR}_{[\text{T,BSR}]}$	$\text{SUR}_{[\text{T,Cos}]}$
medical-genetics	77.00	79.00	81.00	85.00
prof-psychology	67.00	66.00	69.50	70.00
formal-logic	56.35	61.11	59.52	59.52
moral-disputes	79.50	76.00	82.00	79.50
public-relations	73.64	72.73	76.36	77.27
comp-security	78.00	74.00	76.00	79.00
astronomy	80.26	75.66	82.24	79.61
abstract-algebra	57.00	61.00	67.00	63.00
nutrition	79.50	79.50	78.50	81.00
high-school-biology	76.50	76.00	79.00	75.00
formal-fallacies	46.50	42.50	47.50	43.00
tracking-3	41.00	35.50	40.00	39.50
Average	67.69	66.58	69.89	69.28

Table 9: Performance across different datasets and different TracIn Influence methods with $k = 3$ shots for Mistral-7b-v0.3

Dataset	PRE_{T}	SUR_{T}	$\text{SUR}_{[\text{T,BSR}]}$	$\text{SUR}_{[\text{T,Cos}]}$
medical-genetics	88.00	85.00	87.00	88.00
prof-psychology	83.00	80.00	84.00	86.50
formal-logic	65.08	69.05	65.87	68.25
moral-disputes	87.50	84.50	84.50	89.00
public-relations	80.91	82.73	85.45	81.82
comp-security	87.00	83.00	86.00	87.00
astronomy	86.18	86.18	88.16	90.13
abstract-algebra	61.00	51.00	62.00	57.00
nutrition	85.00	83.00	88.50	86.00
high-school-biology	86.50	84.50	88.00	85.00
formal-fallacies	47.00	52.00	54.50	62.50
tracking-3	44.00	42.00	35.00	38.00
Average	75.10	73.58	75.75	76.60

Table 10: Performance across different datasets and different TracIn Influence methods with $k = 3$ shots for Zephyr-7b-beta

Dataset	PRE_{T}	SUR_{T}	$\text{SUR}_{[\text{T,BSR}]}$	$\text{SUR}_{[\text{T,Cos}]}$
medical-genetics	76.00	77.00	74.00	80.00
prof-psychology	72.50	71.50	72.50	72.50
formal-logic	67.46	69.84	67.46	64.29
moral-disputes	77.50	75.50	76.00	77.50
public-relations	76.36	74.55	79.09	72.73
comp-security	77.00	78.00	76.00	75.00
astronomy	76.32	78.95	81.58	80.92
abstract-algebra	50.00	48.00	53.00	52.00
nutrition	75.50	77.00	75.50	78.00
high-school-biology	77.50	75.50	80.50	78.00
formal-fallacies	50.00	41.00	46.00	50.00
tracking-3	50.50	48.00	49.50	42.50
Average	68.89	67.90	69.26	68.62

Table 11: Performance across different datasets and different TracIn Influence methods with $k = 5$ shots for Llama2-13b-chat.

Dataset	PRE_{T}	SUR_{T}	$\text{SUR}_{[\text{T,BSR}]}$	$\text{SUR}_{[\text{T,Cos}]}$
medical-genetics	82.00	78.00	83.00	81.00
prof-psychology	69.00	67.50	73.50	73.50
formal-logic	61.90	61.11	57.14	57.14
moral-disputes	81.00	81.00	82.50	81.50
public-relations	70.00	70.00	73.64	78.18
comp-security	75.00	76.00	77.00	76.00
astronomy	82.24	76.32	83.55	78.95
abstract-algebra	53.00	56.00	65.00	64.00
nutrition	77.50	75.50	79.00	79.50
high-school-biology	77.00	74.50	82.50	77.00
formal-fallacies	45.50	47.00	39.50	48.00
tracking-3	40.50	41.50	46.50	36.50
Average	67.87	67.03	70.23	69.27

Table 12: Performance across different datasets and Pre-training based demonstration selection methods ($k = 3$ shots) for Llama2-13b-chat.

Dataset	$\text{PRE}_{[\text{D,BSR}]}$	$\text{PRE}_{[\text{D,Cos}]}$	$\text{PRE}_{[\text{T,BSR}]}$	$\text{PRE}_{[\text{T,Cos}]}$
medical-genetics	85.00	80.00	86.00	84.00
prof-psychology	68.50	73.50	70.50	69.50
formal-logic	55.56	57.14	57.94	54.76
moral-disputes	80.50	80.00	78.50	82.00
public-relations	80.91	77.27	74.55	78.18
comp-security	75.00	79.00	80.00	76.00
astronomy	80.26	76.32	78.95	77.63
abstract-algebra	57.00	56.00	58.00	61.00
nutrition	81.00	80.00	78.00	80.50
high-school-biology	75.50	73.50	80.00	76.00
formal-fallacies	52.50	48.00	46.50	45.50
tracking-3	48.50	47.50	37.00	38.00
Average	70.02	69.02	68.83	68.59

Table 13: Performance across different datasets and Pre-training based demonstration selection methods with $k = 3$ shots for Mistral-7b-v0.3.

Dataset	$\text{PRE}_{[\text{D,BSR}]}$	$\text{PRE}_{[\text{D,Cos}]}$	$\text{PRE}_{[\text{T,BSR}]}$	$\text{PRE}_{[\text{T,Cos}]}$
medical-genetics	86.00	88.00	86.00	88.00
prof-psychology	87.50	83.50	85.00	85.50
formal-logic	64.29	65.87	65.08	65.87
moral-disputes	86.00	85.00	85.00	85.50
public-relations	85.45	80.00	86.36	84.55
comp-security	82.00	88.00	85.00	89.00
astronomy	88.16	90.13	87.50	89.47
abstract-algebra	62.00	59.00	62.00	60.00
nutrition	86.50	84.00	86.50	84.50
high-school-biology	86.00	85.50	86.50	87.00
formal-fallacies	54.50	46.00	51.00	50.00
tracking-3	49.50	48.50	50.00	53.00
Average	76.49	75.29	76.32	76.87

Table 14: Performance across different datasets and Pre-training based demonstration selection methods with $k = 3$ shots for Zephyr-7b-beta.

Dataset	$\text{PRE}_{[\text{D,BSR}]}$	$\text{PRE}_{[\text{D,Cos}]}$	$\text{PRE}_{[\text{T,BSR}]}$	$\text{PRE}_{[\text{T,Cos}]}$
medical-genetics	82.00	77.00	81.00	82.00
prof-psychology	73.00	70.50	74.50	73.00
formal-logic	69.84	67.48	65.08	66.67
moral-disputes	73.00	76.00	75.50	76.50
public-relations	78.18	77.27	76.36	70.00
comp-security	78.00	73.00	75.00	76.00
astronomy	76.97	78.29	77.63	79.61
abstract-algebra	51.00	55.00	58.00	58.00
nutrition	79.00	74.50	76.50	78.50
high-school-biology	78.00	77.00	80.00	78.50
formal-fallacies	54.50	55.50	46.50	54.50
tracking-3	47.00	47.50	49.00	48.50
Average	70.04	69.08	69.59	70.15

Table 15: Performance across different datasets and Pre-training based demonstration selection methods with $k = 5$ shots for Llama2-13b-chat.

Dataset	$\text{PRE}_{[\text{D,BSR}]}$	$\text{PRE}_{[\text{D,Cos}]}$	$\text{PRE}_{[\text{T,BSR}]}$	$\text{PRE}_{[\text{T,Cos}]}$
medical-genetics	**83.00**	**83.00**	82.00	**83.00**
prof-psychology	73.50	**74.00**	**74.00**	71.50
formal-logic	60.32	56.35	**62.70**	**62.70**
moral-disputes	82.00	**82.50**	81.00	79.50
public-relations	75.45	75.45	**78.18**	77.27
comp-security	77.00	77.00	76.00	**80.00**
astronomy	**82.24**	81.58	81.58	77.63
abstract-algebra	60.00	59.00	**62.00**	60.00
nutrition	**81.00**	78.50	80.50	79.50
high-school-biology	**80.50**	78.50	79.50	76.00
formal-fallacies	48.50	**50.50**	49.00	49.00
tracking-3	50.00	46.00	47.50	**51.50**
Average	71.13	70.20	**71.16**	70.63

Table 16: Performance comparison between BSR and $\text{SUR}_{[\text{D,BSR}]}$ with DeBERTa as the surrogate model with ($k = 3$ shots) for Llama2-13b-chat.

Dataset	BSR	$\text{PRE}_{[\text{D,BSR}]}$
medical-genetics	87.00	85.00
prof-psychology	70.00	72.50
formal-logic	59.52	59.52
moral-disputes	78.00	84.50
public-relations	79.09	76.36
comp-security	76.00	77.00
astronomy	80.26	80.26
abstract-algebra	58.00	59.00
nutrition	77.5	79.00
high-school-biology	76.50	79.00
formal-fallacies	52.00	45.5
tracking-3	44.00	46.50
Average	69.82	**70.30**

B.5 Qualitative Analysis

For MoT, to understand the merits of our method, we compare demonstrations selected via BSR and SUR$_{[\text{D,BSR}]}$ on the MMLU-abstract-algebra dataset:

Example 1:

Q: Compute the product in the given ring. $(2, 3)(3, 5)$ in $\mathbb{Z}_5 \times \mathbb{Z}_9$

Options: (B) (3,1) (C) (1,6)

BSR Shots.

1. **Q:** Statement 1 | Every element of a group generates a cyclic subgroup of the group. Statement 2 | The symmetric group S_{10} has 10 elements. **Options:** (A) True, True (C) True, False **Answer:** (C)

2. **Q:** Statement 1 | Every function from a finite set onto itself must be one-to-one. Statement 2 | Every subgroup of an abelian group is abelian. **Options:** (A) True, True (D) False, True **Answer:** (A)

3. **Q:** How many attempts should you make to cannulate a patient before passing the job on to a senior colleague, according to the medical knowledge of 2020? **Options:** (A) 4 (B) 3 (C) 2 (D) 1 **Answer:** (C)

SUR$_{[\text{D,BSR}]}$.

1. **Q:** Statement 1 | Every function from a finite set onto itself must be one-to-one. Statement 2 | Every subgroup of an abelian group is abelian. **Options:** (A) True, True (D) False, True **Answer:** (A)

2. **Q:** Olivia used the rule "Add 11" to create the number pattern shown below: 10, 21, 32, 43, 54. Which statement about the number pattern is true? **Options:** (B) The number pattern will never have two even numbers next to each other. (D) If the number pattern started with an odd number, then the pattern would have only odd numbers in it. **Answer:** (B)

3. **Q:** Tomorrow is 11/12/2019. What is the date one year ago from today in MM/DD/YYYY format? **Options:** (B) 11/11/2018 (C) 08/25/2018 **Answer:** (B)

We can see that while BSR selects more semantically relevant samples, SUR$_{[\text{D,BSR}]}$'s selected shots guide the model toward the correct answer (C) instead of (B) by encouraging more structured reasoning.

Example 2:

Q: Statement 1 | If R is an integral domain, then $R[x]$ is an integral domain. Statement 2 | If R is a ring and $f(x)$ and $g(x)$ are in $R[x]$, then

$$\deg(f(x)g(x)) = \deg f(x) + \deg g(x).$$

Options: (C) True, False (B) False, False

BSR Shots.

1. **Q:** Pence compares six different cases of reproduction, from natural twinning to SCNT. What conclusion does he draw from this comparison? **Options:** (A) SCNT is not a different kind of reproduction because there are no morally relevant differences between it and other permissible means of reproduction. (B) Because there is a low risk of harm for natural twinning, there will be a low risk of harm for SCNT. (C) Both A and B (D) Neither A nor B **Answer:** (A)

2. **Q:** Statement 1 | Every element of a group generates a cyclic subgroup of the group. Statement 2 | The symmetric group S_{10} has 10 elements. **Options:** (A) True, True (C) True, False **Answer:** (C)

3. **Q:** Statement 1 | Every function from a finite set onto itself must be one-to-one. Statement 2 | Every subgroup of an abelian group is abelian. **Options:** (A) True, True (D) False, True **Answer:** (A)

SUR$_{[\text{D,BSR}]}$.

1. **Q:** Statement 1 | Every function from a finite set onto itself must be one-to-one. Statement 2 | Every subgroup of an abelian group is abelian. **Options:** (A) True, True (D) False, True **Answer:** (A)

2. **Q:** Select the best translation into predicate logic. George borrows Hector's lawnmower. (g: George; h: Hector; l: Hector's lawnmower; $Bxyx$: x borrows y from z). **Options:** (A) B_{lgh} (B) B_{hlg} (C) B_{glh} (D) B_{ghl} **Answer:** (C)

3. **Q:** Statement 1 | Every element of a group generates a cyclic subgroup of the group. Statement 2 | The symmetric group S_{10} has 10 elements. **Options:** (A) True, True (C) True, False **Answer:** (C)

Here again, BSR selects the more semantically relevant shots (with the top three shots ordered in ascending order of relevance), while SUR$_{[D,BSR]}$ selects less semantically similar but more influential shots, which ultimately improves model performance.

C Extended Noisy ICL Results

C.1 Varying Noise Levels

We also test whether our method can perform well on varying noise levels in the dataset. To test this, we create 2 datasets of MRPC with 10% and 30% noise added. As seen in Table 17, in both the cases IF Averaging outperformed other baselines. With the DataInf configuration performing better for the 10% noise dataset and the LiSSA configuration performing better for the 30% noise dataset.

Table 17: MRPC 10% and 30% Noise added results using various methods (top 2 performers in bold).

	Method	MRPC 0.1	MRPC 0.3
	RAND	70.1	70.8
	BSR	70.3	70.3
	COS	70.6	70.6
	BM25	73.5	73.0
PRU-0.1	RAND$_{[Cos]}$	68.6	69.9
	SUR$_{[D,Cos]}$	68.4	69.9
	SUR$_{[L,Cos]}$	68.9	68.9
	RAND$_{[BSR]}$	69.1	68.6
	SUR$_{[D,BSR]}$	66.4	69.1
	SUR$_{[L,BSR]}$	66.7	71.8
AVG-0.5	SUR$_{[D,Cos]}$	**75.0**	70.3
	SUR$_{[L,Cos]}$	69.9	**76.0**
	SUR$_{[D,BSR]}$	**73.8**	70.8
	SUR$_{[L,Cos]}$	72.1	**75.7**

C.2 Varying Hyperparameters

Here we provide results for different IF pruning and IF averaging hyperparameters that we tested with varying levels of noise in Table 18 and Table 19.

C.3 Effectiveness of IFs

We conduct a toy experiment to evaluate the effectiveness of IF-based methods in detecting noisy samples. We introduce 20% noise to the datasets and compute IF values using the Surrogate Model approach. We then calculate the percentage of noisy samples in the top 100 values selected by our IF methods. Results are presented in Table 20

As shown in the table, IF-based methods are highly effective in identifying mislabeled data, significantly aiding demonstration selection in Noisy ICL.

Table 18: MRPC results using various methods and configurations for 10% and 30% Noise.

	Method	MRPC 0.1	MRPC 0.3
PRU-0.2	RAND$_{[Cos]}$	68.9	69.1
	SUR$_{[D,Cos]}$	71.1	69.9
	SUR$_{[L,Cos]}$	70.6	72.3
	RAND$_{[BSR]}$	70.1	68.1
	SUR$_{[D,BSR]}$	70.1	70.1
	SUR$_{[L,BSR]}$	71.3	66.7
PRU-0.3	RAND$_{[Cos]}$	69.9	69.6
	SUR$_{[D,Cos]}$	71.8	69.4
	SUR$_{[L,Cos]}$	71.3	71.3
	RAND$_{[BSR]}$	69.4	67.9
	SUR$_{[D,BSR]}$	70.3	71.1
	SUR$_{[L,BSR]}$	72.1	73.0
AVG-0.4	SUR$_{[D,Cos]}$	71.8	69.9
	SUR$_{[L,Cos]}$	69.4	73.0
	SUR$_{[D,BSR]}$	72.1	75.4
	SUR$_{[L,BSR]}$	67.4	70.3
AVG-0.6	SUR$_{[D,Cos]}$	70.3	73.5
	SUR$_{[L,Cos]}$	74.3	73.3
	SUR$_{[D,BSR]}$	70.3	71.8
	SUR$_{[L,BSR]}$	71.3	73.8

Table 19: Noisy ICL Accuracy with different hyperparameters for our methods.

	Method	MRPC 0.2	QNLI 0.2	SST2 0.2	QQP 0.2
PRU-0.2	RAND$_{[Cos]}$	68.1	68.6	86.6	70.0
	SUR$_{[D,Cos]}$	67.7	67.2	86.2	70.0
	SUR$_{[L,Cos]}$	71.3	65.8	86.8	67.2
	RAND$_{[BSR]}$	69.1	72.2	85.4	69.8
	SUR$_{[D,BSR]}$	70.3	67.4	81.8	71.6
	SUR$_{[L,BSR]}$	70.1	66.4	85.2	70.8
PRU-0.3	RAND$_{[Cos]}$	70.1	68.4	87.0	67.4
	SUR$_{[D,Cos]}$	68.8	69.2	84.6	70.6
	SUR$_{[L,Cos]}$	70.6	68.2	86.6	72.6
	RAND$_{[BSR]}$	70.8	65.8	86.0	70.8
	SUR$_{[D,BSR]}$	70.6	67.4	84.8	72.0
	SUR$_{[L,BSR]}$	69.4	68.2	87.4	68.4
AVG-0.4	SUR$_{[D,Cos]}$	75.7	71.4	91.6	62.6
	SUR$_{[L,Cos]}$	73.3	67.2	90.6	75.2
	SUR$_{[D,BSR]}$	74.3	68.8	89.8	65.2
	SUR$_{[L,BSR]}$	56.6	72.8	78.0	73.0
AVG-0.6	SUR$_{[D,Cos]}$	73.5	69.2	87.2	66.2
	SUR$_{[L,Cos]}$	72.1	68.6	84.6	70.8
	SUR$_{[D,BSR]}$	73.5	69.2	94.2	71.8
	SUR$_{[L,BSR]}$	72.3	65.8	94.2	75.4

Table 20: Percentage of noisy samples in the top 100 values selected by IF methods.

Dataset	DataInf	LiSSA
MRPC	83%	66%
QNLI	54%	86%
QQP	79%	95%
SST-2	90%	96%

D Memory Consumption

To analyze the added computational costs associated with IFs, we calculate the maximum GPU memory consumption while performing demonstration selection with the Pretrained Gradients-DataInf PRE$_{\text{D}}$ and Surrogate Model-DataInf SUR$_{\text{D}}$ methods. The experiments are performed on 4 NVIDIA RTX 6000 Ada Generation GPUs. The maximum memory consumption for Pre-D was 18,188 MiB, while for Sur-D it was 7,998 MiB. These memory requirements are relatively modest, and the use of IFs can be justified given the benefits they provide.

E Scalability to large datasets

For larger datasets, we compare the time taken to extract test gradients and compute IF for 100 samples, 200 samples, and 1000 samples in Table 21.

Table 21: Time taken for computing test gradients and influence functions (IF) across different models and sample sizes.

Model & Samples	Test Gradients (s)	Computing IF (s)
RoBERTa (100 Samples)	6.7	26.4
RoBERTa (200 Samples)	7.4	35.7
RoBERTa (1000 Samples)	67.8	273.7
Llama-2-13b-chat (100 Samples)	41.0	3.6
Llama-2-13b-chat (200 Samples)	68.5	4.7
Llama-2-13b-chat (1000 Samples)	410.8	35.4

A 10x increase in sample size corresponds to an approximately 10x increase in computational time, indicating a linear relationship between sample size and computational time.

Finally, in MoT, the computational time of IF can further be optimized by only computing IF for the $2k$ shots being pruned by BSR or Cosine similarity instead of the entire set of training demonstrations. Another optimization to the DataInf code could be replacing their handling of gradients with tensor operations instead of the current dict of dicts format. This enables the use of GPU processing for influence computation instead of CPU and can offer a considerable runtime speedup.