# An Experimental Design Framework for Label-Efficient Supervised Finetuning of Large Language Models

**Gantavya Bhatt**[*,1], **Yifang Chen**[*,1], **Arnav M. Das**[*,1], **Jifan Zhang**[*,2]
**Sang T. Truong**[3], **Stephen Mussmann**[4], **Yinglun Zhu**[5], **Jeffrey Bilmes**[1],
**Simon S. Du**[1], **Kevin Jamieson**[1], **Jordan T. Ash**[6], **Robert D. Nowak**[2]
[1]University of Washington, Seattle    [2]University of Wisconsin-Madison
[3]Stanford University    [4]Georgia Institute of Technology
[5]University of California, Riverside    [6]Microsoft Research NYC

## Abstract

Supervised finetuning (SFT) on instruction datasets has played a crucial role in achieving the remarkable zero-shot generalization capabilities observed in modern large language models (LLMs). However, the annotation efforts required to produce high quality responses for instructions are becoming prohibitively expensive, especially as the number of tasks spanned by instruction datasets continues to increase. Active learning is effective in identifying useful subsets of samples to annotate from an unlabeled pool, but its high computational cost remains a barrier to its widespread applicability in the context of LLMs. To mitigate the annotation cost of SFT and circumvent the computational bottlenecks of active learning, we propose using experimental design. Experimental design techniques select the most informative samples to label, and typically maximize some notion of uncertainty and/or diversity. In our work, we implement a framework that evaluates several existing and novel experimental design techniques and find that these methods consistently yield significant gains in label efficiency with little computational overhead. On generative tasks, to reach the same generalization performance, our methods save 50% of the annotation cost compared to random sampling.

## 1 Introduction

Supervised finetuning (SFT) on instruction datasets has shown immense potential in improving the zero-shot performance of large language models (LLMs) (Wei et al., 2022). Recent developments in this field have been fostered by the availability of large-scale instruction datasets, consisting of natural language instructions with desired responses based on human judgment (Wei et al., 2022; Mishra et al., 2022; Longpre et al., 2023). Throughout the community, there have been several efforts to further increase the number of tasks included in these

datasets to improve LLM generalization (Xu et al., 2022; Wang et al., 2022; Honovich et al., 2022; Wang et al., 2023c). In addition, supervised finetuning is especially important for handling novel forms of tasks. For example, to develop new multimodal capabilities or defend against novel prompt hacking strategies, one must need human generated responses to finetune existing state-of-art models like GPT-4.

A crucial bottleneck of SFT is the need for annotating a massive set of instructions with detailed responses, which can be done either manually or automatically. For improving state-of-art models, one necessarily need to rely on manual approaches. This involves the use of crowd-workers or expert annotators, who produce high quality responses for almost any task but can become extremely expensive at scale due to the labor intensive process of annotation. To improve small scale and more domain-specific language models, automatic annotation methods have been proposed to reduce the burden on human annotators by labeling instructions using preexisting LLMs (Honovich et al., 2022; Wang et al., 2023c; Peng et al., 2023). However, models like GPT-4 are still costly to query, making the annotation for a large set of prompts potentially prohibitive. In addition, for specialized domains, human experts are still crucial in this annotation process as general LLMs may not provide high quality responses. Therefore, we seek to answer the following question:

> *Can we propose label-efficient strategies that annotate fewer prompts while obtaining equally good generalization performance?*

Towards improving the label efficiency of deep models, researchers have been predominantly focusing on *Active Learning* (Settles, 2009; Gal et al., 2017; Sener and Savarese, 2017; Ash et al., 2019; Zhang et al., 2023). These techniques have been proven useful in some relatively easier NLP tasks such as machine translation (Honovich et al., 2022)

---

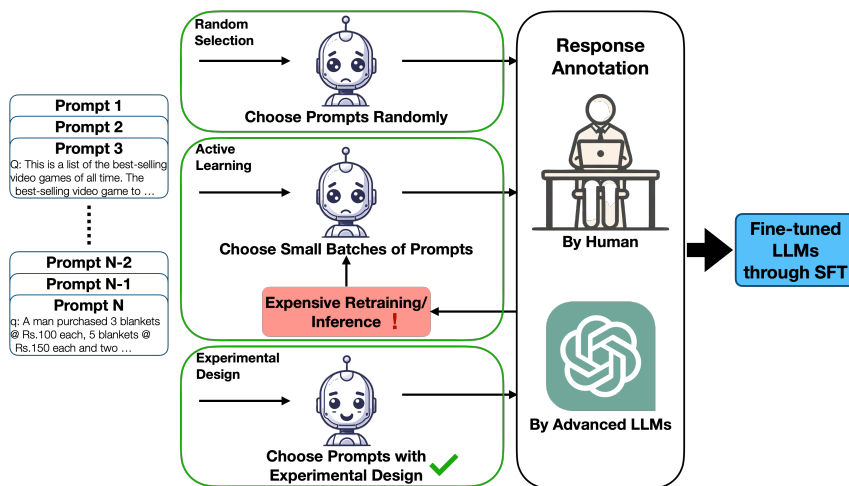*Equal contribution, alphabetically ordered.

Figure 1: Comparison between different annotation schemes for label-efficient SFT. Random sampling simply chooses prompts uniformly at random which underperforms as it is prone to redundancy and may oversample from the major modes. On the other hand, one can choose them more strategically both through active learning and experimental design. Active learning, however, is an adaptive procedure and requires computationally expensive model retraining and inference for every batch of annotation. In this paper, we study the problem through the lens of experimental design, which enjoys increased label-efficiency compared to random sampling, while incurring minimal computation cost compared to active learning.

and classification (Perlitz et al., 2023), but remain under-explored in general natural language generation tasks. As shown in Figure 1, the active learning framework incrementally adds batches of samples to the labeled pool by repeatedly: (1) training a model on the currently labeled data and (2) using some model dependent measure of informativeness to select a new batch of points to query the annotator (Atlas et al., 1989; Settles, 2011). However, each iteration of active learning requires performing inference on all unlabeled samples and retraining the model on the expanded set of labeled samples (Coleman et al., 2020; Das et al., 2023). In the context of parameter-heavy LLMs where inference and training are expensive, the computational cost associated with active learning may outweigh its potential savings in annotation costs.

In this work, we propose leveraging *experimental design* to select the optimal set of instructions to annotate. Experimental design concerns situations where we have to organize an experiment in order to gain some information about an object of interest. In the context of training an AI system, the "experiment" involves selecting a subset of unlabeled examples from a large pool for labeling in order to create a training set to learn a good model (the object of interest, in this case). As demonstrated in Figure 1, in contrast to active learning where the labeled set is expanded incrementally, experimental design techniques select the set of

samples to label in *a single step based solely on the initial model*. This circumvents almost all of the computational cost of active learning, allowing for gains in label efficiency to be realized with practically no overhead. While experimental design has been studied theoretically, its empirical benefits have been underexplored.

We introduce a framework for evaluating experimental design techniques for finetuning LLMs, and propose a suite of experimental design techniques that demonstrably improve the label efficiency of SFT. We develop novel scores, such as maximum token uncertainty, that quantify the LLMs uncertainty on a particular sample and correlate well with its usefulness as training data. We also propose a class of experimental design heuristics that employ the facility location function as an objective, to select a diverse and representative set of samples for annotation.

Overall, our contributions can be summarized as follows: **(1)** we are the first, to the best of our knowledge, to utilize experimental design for SFT **(2)** we introduce a framework to perform comprehensive evaluation on existing experimental design techniques **(3)** we propose a suite of novel strategies that improve the label efficiency, significantly outperforming random sampling by more than $2\%$ accuracy across different annotation budgets **(4)** compared to previous works (Kung et al., 2023; Perlitz et al., 2023), our work is the first to see an-

notation cost savings on *generative* tasks. To reach the same generalization performance, our methods save 50% of the annotation cost compared to random sampling (unlike random sampling that fails to achieve the same generalization).

## 2 Related Work

**Experimental Design**   Experimental design generally refers to situations where the experiment is chosen before the collection of information (labels) starts. This is in contrast to situations where the experiment is designed in a sequential adaptive fashion, responding to information contained in labels to guide selection of the next points in the design. This is called sequential experimental design or active learning in the parlance of machine learning. See Pukelsheim (2006) for an excellent overview of classical experimental design techniques and Pronzato and Pázman (2013) for a treatment of modern approaches to experimental design in nonlinear models.

Our study focuses on experimental design, rather than active learning. Active learning generally requires retraining the model and forward inference on the entire pool after each batch of labels is gathered in order to select the next batch for labeling. The retraining and repeated inference can be computationally expensive, particularly in the case of LLMs. Experimental design does not require additional computation of this sort, since the subset selection of unlabeled examples is done just once before any labels are collected.

**Deep Active Learning**   Data selection strategies for label-efficient learning have been largely studied under the framework of deep active learning, which sequentially and adaptively choose informative examples to annotate. Deep active learning methods typically use measures of uncertainty (Atlas et al., 1989; Settles, 2011; Gal et al., 2017; Ducoffe and Precioso, 2018; Beluch et al., 2018), diversity (Sener and Savarese, 2018; Geifman and El-Yaniv, 2017; Citovsky et al., 2021), or some combination of both (Wei et al., 2015; Ash et al., 2019, 2021; Zhang et al., 2022; Nuggehalli et al., 2023) in order to determine a set of useful samples to annotate. In addition to classical instance-level selection, there are also many existing works focused on task-wise selection (Xu et al., 2023; Chen et al., 2023b; Wang et al., 2023a; Fifty et al., 2021). While we propose an experimental design framework for the label-efficient SFT problem, our strategies are inspired by the latest instance-level deep active learning literature and adapted to work under this framework for supervised finetuning of LLMs.

A few works have considered the application of active learning for SFT or other closely related settings. Perlitz et al. (2023) explores the use of active learning to improve the label efficiency of the closely related task of natural language generation. However, this work reports inconsistent findings showing that there are limited settings where active learning demonstrates any significant advantages over random sampling. Kung et al. (2023) is the most similar to our work, proposing an active learning framework for instruction tuning. Unlike our work, Kung et al. (2023) performs task-level selection, as opposed to instance-level selection. In other words, their framework estimates the usefulness of each unlabeled task and annotates all instances within the tasks that are deemed most useful. However, task-level selection makes the simplifying assumption that every instance is equally useful within a task, which may inhibit the quality and reduce the resolution of the selected subset. Moreover, the approach in Kung et al. (2023) suffers from the aforementioned computational cost of active learning which is avoided in our framework with the use of experimental design.

**Data Selection for SFT**   Many recent works demonstrate that small subsets of instruction data can be sufficient for finetuning an LLM. Zhou et al. (2023) demonstrate that finetuning an LLM on a dataset consisting of only manually-curated 1000 instruction/response pairs is sufficient to achieve strong generalization, but do not propose any general algorithmic procedure for subset selection. Other works propose instruction dataset pruning techniques that select subsets based on some combination of quality, diversity, and/or difficulty (Chen et al., 2023a; Bukharin and Zhao, 2023; Du et al., 2023; Li et al., 2023a, 2024). These approaches use both the instruction and its corresponding response to choose which training samples should be retained to improve the computational efficiency of SFT. Unlike these approaches, our framework is designed to maximize label efficiency and assumes that the response to an instruction are not available until selected for annotation. Finally, Hu et al. (2023) propose techniques to reduce the annotation cost to construct validation sets that are used for model selection, which is complementary to the

method proposed in our work.

## 3 Prompt Selection Strategy

Under the experimental design framework, the learner is given a set of initial $N$ prompts $X = \{x_1, x_2, ..., x_N\}$, where each prompt $x_i \in \mathcal{X}$ is a sequence of input tokens of length $\ell_i$, $x_i = \{x_{i,1}, ..., x_{i,\ell_i}\}$, where $\mathcal{X}$ denotes the domain of all possible sequence of input tokens. Additionally, we let $g$ denote the pretrained language model. Given a budget of $k < N$, a selection strategy chooses $k$ prompts from $X$, denoted as $S \subset X$, based on different measures of informativeness of annotating an example to the pretrained model $g$. One then gathers well-written responses to prompts in $S$ from annotators (e.g., human experts or advanced LLMs). Experimental design aims to optimize the performance of the model $g'$ finetuned on the selected prompts $S$ and their responses.

### 3.1 Uncertainty-Based Selection

During the prompt selection step, since the ground truth responses are oblivious to the learner, one type of label-efficient selection strategy is to choose examples with the highest uncertainty to the model. Specifically, let $U : X \to \mathbb{R}$ define some notion of uncertainty of prompts in $X$, uncertainty-based methods simply choose the top-k most uncertain examples by

$$S = \underset{\substack{S' \subset X \\ |S'| = k}}{\operatorname{argmax}} \min_{x \in S'} U(x). \qquad (1)$$

Below, we provide four instances of uncertainty measures. Formally, we let the pretrained model (with greedy decoding) $g : \mathcal{X} \to (\mathcal{D} \times \triangle_{\mathcal{D}})^{[L]}$ map prompts to a sequence of predicted tokens and the softmax probability distribution at each decoding step, up to length $L$. Here each token is from a dictionary $\mathcal{D}$ and each corresponding softmax probability distribution lives in the probability simplex $\triangle_{\mathcal{D}}$ over the dictionary. We let $g_y(\cdot)$ denote the sequence of $L$ tokens and $g_p(\cdot)$ denote the sequence of $L$ softmax probability distributions.

**Mean Entropy** (Settles, 2011; Kremer et al., 2014) measures the tokenwise negative entropy of the softmax probability scores. The uncertainty measure is taken as the mean across tokens as follows:

$$U_{\text{entropy}}(x) = \frac{1}{|g_p(x)|} \sum_{p \in g_p(x)} \sum_{t \in \mathcal{D}} p_t \log(p_t).$$

Where $p_t$ represents the softmax score (i.e., the probability) of the token $t$, and the entire set of $p$ values represents the distribution across the entire dictionary for a specific position in an $L$-word sentence.

**Least Confidence** (Settles, 2009, 2011) measures the model's confidence as the product of probabilities of the generated sequence. A model is more confident when the likelihood of the generated sequence is high. We take the negative confidence score as the uncertainty, which allows us to choose the least confident sequences for annotation:

$$U_{\text{conf}}(x) = - \prod_{(t,p) \in g(x)} p_t.$$

**Mean Margin** (Tong and Koller, 2001; Balcan et al., 2006; Settles, 2011) measures uncertainty by taking the different between most likely and second likely token for each element in the generated sequence . A higher difference corresponds with a clearer separation between the model's best choice from its second best choice. We use the negative margin score average over all tokens as the uncertainty measure:

$$\bar{U}_{\text{margin}}(x) = -\frac{1}{|g_p(x)|} \left( \sum_{p \in g_p(x)} \beta_1(p) - \beta_2(p) \right)$$

where $\beta_1(p)$ and $\beta_2(p)$ denotes the largest and second largest element of $p$.

**Min Margin** is a novel strategy where we measure uncertainty based on the token with the smallest margin score instead of taking the average over all tokens. Intuitively, two sequences could have equal average token-wise margin score, but the one with smaller minimum margin is more likely to be generated as a different sequence with top-2 decoding. Mathematically, this is defined as:

$$\widetilde{U}_{\text{min margin}}(x) = - \left( \min_{p \in g_p(x)} \beta_1(p) - \beta_2(p) \right).$$

### 3.2 k-Center Selection

Another class of label-efficient selection strategy is to annotate prompts that are *diverse* in the representation space. Sener and Savarese (2018) proposed a k-center objective that chooses $k$ examples as centers of balls with equal radius. The objective promotes selections that would minimize the radius of these balls while covering all examples.

Formally, the objective of choosing $k$ centers can be written as:

$$S = \operatorname*{argmin}_{\substack{S' \subset X \\ |S'|=k}} \max_{i \in X} \min_{j \in S'} \|f(x_i) - f(x_j)\|, \quad (2)$$

where $f$ is a feature extractor mapping prompts into feature space in $\mathbb{R}^d$ and is derived from the pretrained model $h$. For decoder-only architectures, we use the last hidden state as the feature. To optimize the above NP-hard object (Cook et al., 1998), we follow the greedy methods proposed by Sener and Savarese (2018), which enjoys a 2 multiplicative approximation guarantee to the optimal selection.

### 3.3 Submodular Selection

Equation (2) is commonly recognized as the minimax facility location objective (Farahani and Hekmatfar, 2009). Additionally, we explore the conventional Facility Location (FL) function (Mirchandani and Francis, 1990), extensively used in machine learning (Wei et al., 2015; Mirzasoleiman et al., 2020; Bilmes, 2022; Bukharin and Zhao, 2023). Given a nonnegative score $w_{ij}$ that measures the similarity between features $f(x_i)$ and $f(x_j)$, the facility location problem is formulated as follows:

$$S = \operatorname*{argmax}_{\substack{S' \subset X \\ |S'|=k}} \sum_{i \in X} \max_{j \in S'} w_{ij} \quad (3)$$

In Equation (3), every client $i \in X$ must have a facility within $S$, which is chosen to be the element $j \in S$ closest to $i$. FL is a known submodular function, so the greedy heuristic applied to this objective achieves a $1 - 1/e$ multiplicative approximation guarantee to the optimal solution (Nemhauser et al., 1978) despite its NP-Hard nature. The greedy algorithm can be further accelerated with the use of data structures (Minoux, 2005), or with stochastic variants (Mirzasoleiman et al., 2014).

We primarily use the radial basis function as a similarity metric, where $w_{ij} = \exp\left(-\frac{\|f(x_i) - f(x_j)\|^2}{\gamma}\right)$ and $\gamma > 0$ is a controllable hyperparameter often referred to as the kernel width. Intuitively, $\gamma$ tunes is the degree of similarity between two data points. As $\gamma$ decreases, the similarity between $x_i$ and $x_j$ also decreases. In limiting case as $\gamma \to 0$, similarity $w_{ij} = 0$ for all $i \neq j$. For situations where hyperparameter tuning is not feasible, we propose an alternate function

where the similarity metric is fixed and not tuned, where $w_{ij} = \max\{0, \frac{f(x_i)^T f(x_j)}{\|f(x_i)\| \|f(x_j)\|}\}$.

## 4 Experiments

In this section, we compare multiple experimental design strategies against random sampling and observe significant improvements. We describe the various setups in Section 4.1 and report the evaluation results based on common benchmarks (Section 4.2) and comparisons by GPT-4 (Section 4.3). Lastly, we conduct ablation study and document the selection of hyperparameters in Section 4.4. Details about our experimental compute cost and complexity can be found in Appendix 5.

### 4.1 Experiment Setup

**Dataset** FLAN V2 (Longpre et al., 2023) is a widely-used instruction fine-tuning dataset that combines FLAN 2021, P3++, Super-Natural Instructions, along with additional reasoning, dialogue, and program synthesis datasets. We utilize a 99K subset of FLAN V2, processed by (Wang et al., 2023b), as our training data pool, from which we select prompts and annotations (i.e., responses).

**Models and Training Procedure** We conduct experiments with the 7B version of the prefix language model LLaMA-2 (Touvron et al., 2023), across different annotation budgets. Prior to fine-tuning, we choose a subset of prompts for annotation by either random sampling or using experimental design strategies, including uncertainty-based selection, k-Center selection, and submodular selections. These strategies are computed based on the prefix model only. Subsequently, we fine-tune the model on the annotated prompt/response pairs using Low-rank Adaptation (LoRA) (Hu et al., 2021).

**Evaluation metrics** We adopt the similar evaluation tasks as in the original FLAN V2, using the MMLU (Hendrycks et al., 2020) and BBH (Suzgun et al., 2022) benchmarks to evaluate the zero-shot generalization capability of our fine-tuned model. Massive Multitask Language Understanding dataset (MMLU) is a classification task with a set of questions about 57 subjects ranging in difficulty from elementary levels to professional levels, broadly testing mode's factual knowledge and reasoning. Big-Bench-Hard (BBH) is a generation task with 23 challenging tasks from Big-Bench (Srivastava et al., 2022), broadly testing models' gen-

| Strategy | $k = 0$ | $k = 20K$ | $k = 30K$ | $k = 45K$ |
|---|---|---|---|---|
| Random | 37.58 | 44.33($\pm$0.32) | 44.91($\pm$0.50) | 45.99($\pm$0.37) |
| Mean Entropy | 37.58 | 43.85($\pm$0.14) | <u>45.38</u>($\pm$0.21) | <u>46.45</u>($\pm$0.34) |
| Confidence | 37.58 | 43.26($\pm$0.58) | 44.56($\pm$0.42) | <u>46.55</u>($\pm$0.25) |
| Mean Margin | 37.58 | 43.85($\pm$0.33) | 44.88($\pm$0.23) | <u>46.40</u>($\pm$0.16) |
| Min Margin | 37.58 | <u>44.55</u>($\pm$0.32) | <u>45.62</u>($\pm$0.14) | 45.31($\pm$0.14) |
| k-Center | 37.58 | 43.77($\pm$0.47) | <u>46.14</u>($\pm$0.12) | <u>46.27</u>($\pm$0.14) |
| FL (cosine) | 37.58 | 43.77($\pm$0.23) | <u>45.89</u>($\pm$0.50) | <u>47.01</u>($\pm$0.37) |
| FL ($\gamma = .002$) | 37.58 | **45.08**($\pm$0.33) | **47.12**($\pm$0.35) | **47.63**($\pm$0.24) |

Table 1: Massive Multitask Language Understanding (MMLU) evaluation of models trained on subsets selected by strategies from a pool of 99k under different annotation budgets. Each result of random strategy is averaged over 6 seeds due to the high variance from both data selection and training. Other results are averaged over 3 random seeds where the randomness mainly comes from the training. The confidence intervals are based on standard error.

| Strategy | $k = 0$ | $k = 20K$ | $k = 30K$ | $k = 45K$ |
|---|---|---|---|---|
| Random | 37.66 | 38.95($\pm$0.48) | 39.42($\pm$0.56) | 39.44($\pm$0.52) |
| Mean Entropy | 37.66 | <u>40.28</u>($\pm$0.60) | 38.18($\pm$0.43) | <u>39.99</u> ($\pm$0.67) |
| Confidence | 37.66 | **40.33**($\pm$0.49) | 38.28($\pm$0.39) | <u>41.04</u>($\pm$0.74) |
| Mean Margin | 37.66 | 38.33($\pm$0.67) | <u>40.05</u>($\pm$0.39) | 38.43($\pm$0.34) |
| Min Margin | 37.66 | <u>39.74</u> ($\pm$0.19) | <u>40.20</u> ($\pm$0.20) | <u>39.66</u> ($\pm$0.37) |
| k-Center | 37.66 | 37.44($\pm$0.60) | 38.35 ($\pm$0.63) | 38.6 ($\pm$0.39) |
| FL (cosine) | 37.66 | 38.25($\pm$0.35) | <u>39.82</u>($\pm$0.25) | <u>40.46</u>($\pm$1.03) |
| FL ($\gamma = .002$) | 37.66 | 38.33 ($\pm$0.27) | **41.12**($\pm$0.71) | **41.30**($\pm$0.60) |

Table 2: Big-Bench-Hard chain-of-thoughts (BBH-CoT) evaluation of models trained on subsets selected by strategies from a pool of 99k under different annotation budgets. Each result of random strategy is averaged over 6 seeds where each result of other strategies is averaged over 3 random seeds with standard error shown as confidence interval.

eral reasoning capabilities. In our study, we specifically select 5-shot MMLU and a random 20% subset of Chain-Of-Thought BBH inputs, due to the computational resource limitations. In addition, we also evaluate our methods based on AlpacaEval (Li et al., 2023b), where given a prompt, the responses from two models are compared by GPT-4 turbo. The performance of the two models are reported as the win rate against each other.

## 4.2 Evaluation on Standard Benchmarks

When comparing different experimental design (i.e. prompt selection) strategies to random sampling in Tables 1,2, we see almost dominant improvements (around 1% to 2%) on Facility Location ($\gamma = 0.002$) strategy except on the 20K BBH-CoT case (where it closely aligns with that of random sampling). Compared to 90K budget where we get MMLU = 47.76($\pm$0.57), BBH-CoT = 40.49($\pm$0.30), we save approximately 50% annotation cost with respect to both classification (MMLU) and generation (BBH) tasks. The facility location function with the cosine kernel also exhibits improvement

over random sampling, although the magnitude of improvement is not as pronounced as observed with the tuned kernel. This disparity could be attributed to the saturation of the Facility Location function, wherein the greedy order fails to generate a diverse summary conducive to downstream applications. In practice, since one could only choose one set of examples to annotate, we provide details in how to choose the hyperparameter in Section 4.4 before annotation and demonstrate the robustness through an ablation study. In addition, our proposed Min Margin strategy also gains much larger improvements compared to others, including the commonly used Mean Margin score.

On the other hand, we observe that most uncertainty-based selection approaches did not improve over random sampling. In some cases, they did not even exhibit monotonic improvements with the selection size. This phenomenon has a well-known explanation – uncertainty-based methods often annotate similar/redundant examples, which could potentially hinder generalization performance when fine-tuning the model.
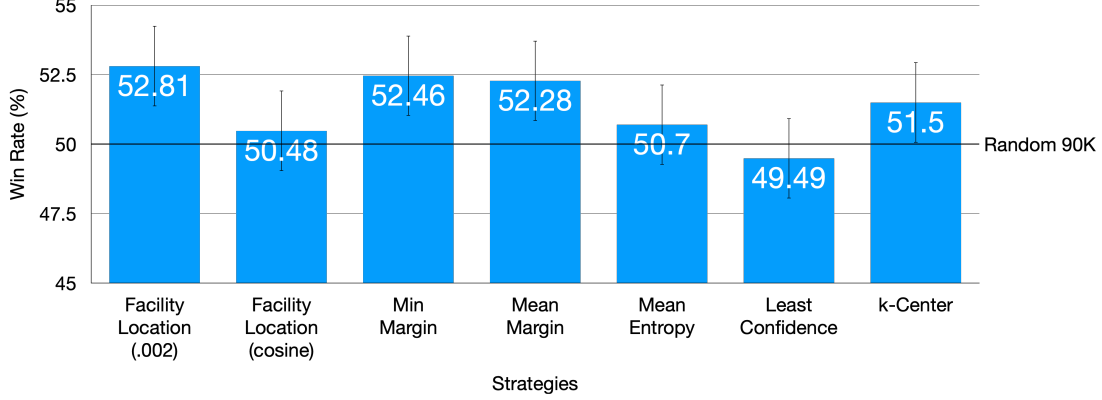
Figure 2: Evaluation by GPT-4 turbo by comparing model trained on 45K prompts selected by various strategies with the model trained on 90K random prompts. We use the win rate weighted by the continuous preferences of the GPT-4 turbo model. Error bars are reported as the standard errors across prompts.

|  | BBH-CoT | MMLU |
|---|---|---|
| FL+MinMargin | 40.71($\pm$0.63) | 46.26($\pm$0.36) |
| MinMargin | 39.66($\pm$0.67) | 45.31($\pm$0.14) |
| FL | 41.30($\pm$0.60) | 47.63($\pm$0.24) |

Table 3: Big-Bench-Hard chain-of-thoughts (BBH-CoT) and Massive Multitask Language Understanding (MMLU) evaluation of mixture of diversity and uncertainty strategy at the budget of 45K, averaged over 3 random seeds with standard error shown as confidence interval. We observe that the performance of the mixture interpolated between the two methods.

**Ablation** We now consider a combination of uncertainty-based selection with diversity, in the following Equation (4).

$$S = \operatorname*{argmax}_{\substack{S' \subset X \\ |S'|=k}} \sum_{i \in X} \max_{j \in S'} w_{ij}$$
$$+ \log \left( 1 + \sum_{x \in S'} \widetilde{U}_{\text{min margin}}(x) \right) \quad (4)$$

The first part governs the diversity and the latter part governs the uncertainty, for which, we consider the min-margin as the uncertainty score. Note that the objective in Equation (4) is a submodular maximization and hence we can use the greedy algorithm. This is a corollary of the fact that modular functions ($\sum_{x \in S'} \tilde{U}(x)$) when composed with monotone, non-decreasing concave function is submodular, and submodularity is closed under conic combination (Bilmes and Bai, 2017; Lin and Bilmes, 2011; Bhatt et al., 2024). In general, one can apply any monotone non-decreasing concave function, which becomes a design choice. Here, we consider $\log(1 + x)$ which was chosen by Das

et al. (2023). We consider min-margin for the uncertainty and RBF kernel based facility location. Table 3 shows the BBH-CoT and MMLU metrics for the mixture of min-margin and diversity at a budget of 45K and observe the performance of the mixture interpolated between the two methods. We posit that with the right choice of uncertainty metric, it is possible to improve beyond solely using uncertainty and diversity when mixed appropriately.

### 4.3 Evaluation by GPT-4

To further demonstrate the annotation cost savings or various experimental design methods, we evaluate models finetuned on 45K prompts against the model trained on 90K prompts using random sampling with GPT-4 turbo as the judge, which displays a high agreement rate with ground truth human annotations . We follow the AlpacaEval framework (Li et al., 2023b) and report the win rate against the model finetuned on 90K randomly selected prompts. The models are evaluated on the 805 prompts from the standard AlpacaEval set (Li et al., 2023b). As showin in Figure 2, multiple strategies reliably achieve the same or better performance while only taking 50% of the annotation budget as compared to the model trained on 90K random prompts. Compared to result from standard benchmark, we again observed that Facility Location ($\gamma = 0.002$) strategy gains dominating improvement. On the other hand, margin-based methods also exhibited significant improvements, while other uncertainty-based methods performed similarly or even worse than random sampling. This suggests that, although uncertainty-based methods may still be helpful in aligning with human preference, selecting the right metric is crucial. Over-

|         | Random         | $\gamma = 0.001$ | $\gamma = 0.002$ | $\gamma = 0.003$ | $\gamma = 0.004$ |
|---------|----------------|------------------|------------------|------------------|------------------|
| MMLU    | $45.99(\pm0.37)$ | $47.04(\pm0.69)$ | $47.63(\pm0.24)$ | $47.96(\pm0.25)$ | $47.94(\pm0.49)$ |
| BBH-CoT | $39.44(\pm0.52)$ | $40.56(\pm0.86)$ | $41.30(\pm0.60)$ | $41.99(\pm0.73)$ | $41.35(\pm1.04)$ |

Table 4: Sensitivity to kernel width $\gamma$ at 45K budget. Each result of random strategy is averaged over 6 seeds where each result of other strategies is averaged over 3 random seeds with standard error shown as confidence interval.

all, compared to standard benchmark results, the evaluation by GPT-4 turbo highlights even greater potential for experimental design in instruction fine-tuning. However, further investigation into how different strategies relate to various metrics (i.e., the model's ability to perform different tasks) is needed to better understand the differences between standard and AlpacaEval benchmark results.

### 4.4 Hyperparameter Selection for Facility Location

In this section, we describe how to choose hyperparameter $\gamma$ before annotation begins. For a submodular function to be able to generate good summaries upon maximization, it should not *saturate*. That is, the facility location objective (equation (3)) eventually increases only minimally by increasing the budget $k$, which suggests the new elements no longer remain representative of the downstream task. Formally, let $S_1 \subset S_2 \subset \ldots \subset S_k$ denote the greedy solution up to size $k$ and $F(S_k) = \sum_{i \in X} \max_{j \in S_k} w_{ij}$ denote the facility location function evaluation for set $S_k$ (defined in equation (3)). The *gain* by adding the $k$-th element is defined as $F(S_k|S_{k-1}) \triangleq F(S_k) - F(S_{k-1})$. In the case of RBF kernel, kernel width $\gamma$ controls the degree of saturation. Higher values of $\gamma$ lead to decreasing gain by adding additional elements. On the other hand, very low values such as $10^{-4}$ will result in the kernel becoming close to the diagonal matrix, and therefore not being able to capture the interactions between data points.

Therefore, to determine the potential range of kernel width, in Figure 3 we first visualize the gains of the submodular function when selecting 45K examples, for $\gamma \in \{10^{-3}, 5 \times 10^{-3}, 10^{-2}, 10^{-1}, 1, 10\}$. The observed trend reveals that for higher $\gamma$, gains tend to attain a very small value (and continue to decrease) even before 1K (for $\gamma = 10$) and 10K (for $\gamma = 1$) elements are selected. Although reducing $\gamma$ helps, gains continue to decrease sublinearly ($\gamma = 0.1$ after 20K). Notably, gains exhibit relative stability for $\gamma \in \{10^{-3}, 5 \times 10^{-3}, 10^{-2}\}$ until we reach the desired budget of 45K, suggesting that we can safely
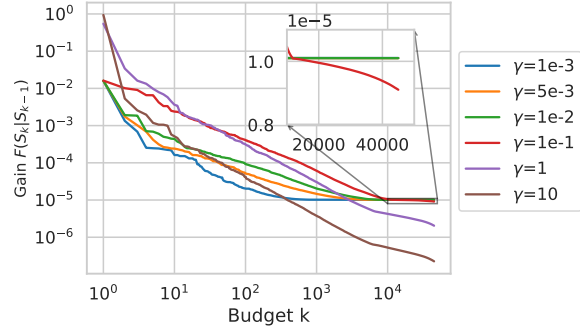


Figure 3: Plot of gains with set size as the course of greedy maximization for different kernel width $\gamma$; we run the greedy procedure till the budget of 45K is reached. The observed trend reveals that for higher $\gamma$, gains tend to attain a very small value (and continue to decrease linearly) even before 1K (for $\gamma = 10$) and 10K (for $\gamma = 1$) elements are selected. Although reducing $\gamma$ helps, gains continue to decrease sublinearly ($\gamma = 0.1$ after 20K). Notably, gains exhibit relative stability for $\gamma \in \{10^{-3}, 5 \times 10^{-3}, 10^{-2}\}$ until we reach the desired budget of 45K, suggesting a potential range for $\gamma$.

choose $\gamma$ in between these values.

With the range of potential $\gamma$ determined, we run an ablation study on 45K budget with four different kernel widths within this range to demonstrate the robustness of any hyperparameter in this range. As detailed in Table 4, the performance is consistently better than random selection strategy across different $\gamma$, which suggests the Facility location methods are less subject to hyperparameter changes once the appropriate range for $\gamma$ has been identified.

## 5 Computational Complexity

We primarily used A100 and A40 GPUs. We precompute the embeddings which takes 12 hours. For A100, each trial takes roughly 24 GPU hours including the evaluation for 90K budget, 15 GPU hours for 45K and 9 hours for 30K respectively. For A40, roughly 30 GPU hours for 90K budget.

## 6 Conclusion and Future Work

Overall, we proposed using experimental design for finetuning LLMs with fewer annotated instructions. We present several heuristics that can be used to determine which set of instructions to annotate, and for the first time, demonstrate significant

gains in label efficiency on generative tasks. Future research directions may include the following:

- Our work demonstrates the first empirical results of experimental design for SFT, but future work can focus on devising new methods within this framework to further improve label efficiency.
- Our setting finetunes the LLM only on the annotated instructions; future work could improve the utilization of unlabeled samples.

## Limitations

While we successfully demonstrate the efficacy of experimental design on SFT, there are a few limitations of this work that we discuss. Firstly, we focus our experiments on the 7B version of Llama-2 since it is the most common and requires the least compute. Secondly, we conduct our experiments on curated datasets where we assume accurate responses to prompts are available. However, we do not consider the effect of inaccurate/noisy responses which can be prevalent when procuring responses from crowd-workers. Finally, we evaluate our model after SFT, and do not consider the effect of reinforcement learning with human feedback (Ouyang et al., 2022) which is frequently applied after SFT.

## Ethics Statement

Our work proposes a computationally inexpensive method of reducing the number of labeled instructions needed for SFT while preserving the performance of the resulting LLM. In doing so, we also mitigate the carbon footprint associated with training such models, which is critical given growing concerns about the environmental impact of modern ML systems. Furthermore, researchers and organizations with limited compute and insufficient financial resources to procure a vast number of crowd-workers can benefit from our work.

It is important to note, however, that all of the experimental design techniques that we propose are forms of *biased* sampling. If the subset of data chosen for annotation is not representative, it is possible biases of pretrained model will be amplified upon finetuning. Thus, it is critical for practitioners to ensure that the subsets selected by our experimental design framework are inclusive.

## Acknowledgements

# References

Jordan Ash, Surbhi Goel, Akshay Krishnamurthy, and Sham Kakade. 2021. Gone fishing: Neural active learning with fisher embeddings. *Advances in Neural Information Processing Systems*, 34:8927–8939.

Jordan T Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. 2019. Deep batch active learning by diverse, uncertain gradient lower bounds. *arXiv preprint arXiv:1906.03671*.

Les Atlas, David Cohn, and Richard Ladner. 1989. Training connectionist networks with queries and selective sampling. *Advances in neural information processing systems*, 2.

Maria-Florina Balcan, Alina Beygelzimer, and John Langford. 2006. Agnostic active learning. In *Proceedings of the 23rd international conference on Machine learning*, pages 65–72.

William H Beluch, Tim Genewein, Andreas Nürnberger, and Jan M Köhler. 2018. The power of ensembles for active learning in image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9368–9377.

Gantavya Bhatt, Arnav Das, and Jeff Bilmes. 2024. Deep submodular peripteral network. *arXiv preprint arXiv:2403.08199*.

Jeff Bilmes. 2022. Submodularity in machine learning and artificial intelligence. *arXiv preprint arXiv:2202.00132*.

Jeffrey Bilmes and Wenruo Bai. 2017. Deep submodular functions. *arXiv preprint arXiv:1701.08939*.

Alexander Bukharin and Tuo Zhao. 2023. Data diversity matters for robust instruction tuning.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, and Hongxia Jin. 2023a. Alpagasus: Training a better alpaca with fewer data.

Yifang Chen, Yingbing Huang, Simon S Du, Kevin Jamieson, and Guanya Shi. 2023b. Active representation learning for general task space with applications in robotics. *arXiv preprint arXiv:2306.08942*.

Gui Citovsky, Giulia DeSalvo, Claudio Gentile, Lazaros Karydas, Anand Rajagopalan, Afshin Rostamizadeh, and Sanjiv Kumar. 2021. Batch active learning at scale. *Advances in Neural Information Processing Systems*, 34:11933–11944.

Cody Coleman, Christopher Yeh, Stephen Mussmann, Baharan Mirzasoleiman, Peter Bailis, Percy Liang, Jure Leskovec, and Matei Zaharia. 2020. Selection via proxy: Efficient data selection for deep learning. In *International Conference on Learning Representations*.

William J Cook, William H Cunningham, William R Pulleyblank, and Alexander Schrijver. 1998. Combinatorial optimisation. *Wiley-Interscience Series in Discrete Mathematics and Optimization, USA*, 1:998.

Arnav Das, Gantavya Bhatt, Megh Bhalerao, Vianne Gao, Rui Yang, and Jeff Bilmes. 2023. Accelerating batch active learning using continual learning techniques.

Qianlong Du, Chengqing Zong, and Jiajun Zhang. 2023. Mods: Model-oriented data selection for instruction tuning.

Melanie Ducoffe and Frederic Precioso. 2018. Adversarial active learning for deep networks: a margin based approach. *arXiv preprint arXiv:1802.09841*.

Reza Zanjirani Farahani and Masoud Hekmatfar. 2009. Facility Location — link.springer.com. https://link.springer.com/book/10.1007/978-3-7908-2151-2. [Accessed 10-01-2024].

Chris Fifty, Ehsan Amid, Zhe Zhao, Tianhe Yu, Rohan Anil, and Chelsea Finn. 2021. Efficiently identifying task groupings for multi-task learning. *Advances in Neural Information Processing Systems*, 34:27503–27516.

Yarin Gal, Riashat Islam, and Zoubin Ghahramani. 2017. Deep bayesian active learning with image data. In *International Conference on Machine Learning*, pages 1183–1192. PMLR.

Yonatan Geifman and Ran El-Yaniv. 2017. Deep active learning over the long tail. *arXiv preprint arXiv:1711.00941*.

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*.

Or Honovich, Thomas Scialom, Omer Levy, and Timo Schick. 2022. Unnatural instructions: Tuning language models with (almost) no human labor.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Zhengyu Hu, Jieyu Zhang, Yue Yu, Yuchen Zhuang, and Hui Xiong. 2023. How many validation labels do you need? exploring the design space of label-efficient model ranking.

Jan Kremer, Kim Steenstrup Pedersen, and Christian Igel. 2014. Active learning with support vector machines. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 4(4):313–326.

Po-Nien Kung, Fan Yin, Di Wu, Kai-Wei Chang, and Nanyun Peng. 2023. Active instruction tuning: Improving cross-task generalization by training on prompt sensitive tasks. *arXiv preprint arXiv:2311.00288*.

Ming Li, Yong Zhang, Zhitao Li, Jiuhai Chen, Lichang Chen, Ning Cheng, Jianzong Wang, Tianyi Zhou, and Jing Xiao. 2023a. From quantity to quality: Boosting llm performance with self-guided data selection for instruction tuning.

Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023b. Alpacaeval: An automatic evaluator of instruction-following models. https://github.com/tatsu-lab/alpaca_eval.

Yunshui Li, Binyuan Hui, Xiaobo Xia, Jiaxi Yang, Min Yang, Lei Zhang, Shuzheng Si, Junhao Liu, Tongliang Liu, Fei Huang, and Yongbin Li. 2024. One shot learning as instruction data prospector for large language models.

Hui Lin and Jeff Bilmes. 2011. A class of submodular functions for document summarization. In *Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies*, pages 510–520.

Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V Le, Barret Zoph, Jason Wei, et al. 2023. The flan collection: Designing data and methods for effective instruction tuning. *arXiv preprint arXiv:2301.13688*.

Michel Minoux. 2005. Accelerated greedy algorithms for maximizing submodular set functions. In *Optimization Techniques: Proceedings of the 8th IFIP Conference on Optimization Techniques Würzburg, September 5–9, 1977*, pages 234–243. Springer.

Pitu B Mirchandani and Richard L Francis. 1990. *Discrete location theory*.

Baharan Mirzasoleiman, Ashwinkumar Badanidiyuru, Amin Karbasi, Jan Vondrak, and Andreas Krause. 2014. Lazier than lazy greedy.

Baharan Mirzasoleiman, Jeff Bilmes, and Jure Leskovec. 2020. Coresets for data-efficient training of machine learning models. In *International Conference on Machine Learning*, pages 6950–6960. PMLR.

Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. 2022. Cross-task generalization via natural language crowdsourcing instructions. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3470–3487, Dublin, Ireland. Association for Computational Linguistics.

George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. 1978. An analysis of approximations for maximizing submodular set functions—i. *Mathematical programming*, 14:265–294.

Shyam Nuggehalli, Jifan Zhang, Lalit Jain, and Robert Nowak. 2023. Direct: Deep active learning under imbalance and label noise. *arXiv preprint arXiv:2312.09196*.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback.

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. 2023. Instruction tuning with gpt-4.

Yotam Perlitz, Ariel Gera, Michal Shmueli-Scheuer, Dafna Sheinwald, Noam Slonim, and Liat Ein-Dor. 2023. Active learning for natural language generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9862–9877, Singapore. Association for Computational Linguistics.

Luc Pronzato and Andrej Pázman. 2013. Design of experiments in nonlinear models. *Lecture notes in statistics*, 212(1).

Friedrich Pukelsheim. 2006. *Optimal design of experiments*. SIAM.

Ozan Sener and Silvio Savarese. 2017. Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489*.

Ozan Sener and Silvio Savarese. 2018. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*.

Burr Settles. 2009. Active learning literature survey.

Burr Settles. 2011. From theories to queries: Active learning in practice. In *Active Learning and Experimental Design workshop In conjunction with AISTATS 2010*, volume 16 of *Proceedings of Machine Learning Research*, pages 1–18, Sardinia, Italy. PMLR.

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*.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, et al. 2022. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*.

Simon Tong and Daphne Koller. 2001. Support vector machine active learning with applications to text classification. *Journal of machine learning research*, 2(Nov):45–66.

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*.

Yiping Wang, Yifang Chen, Kevin Jamieson, and Simon Shaolei Du. 2023a. Improved active multitask representation learning via lasso. In *International Conference on Machine Learning*, pages 35548–35578. PMLR.

Yizhong Wang, Hamish Ivison, Pradeep Dasigi, Jack Hessel, Tushar Khot, Khyathi Raghavi Chandu, David Wadden, Kelsey MacMillan, Noah A Smith, Iz Beltagy, et al. 2023b. How far can camels go? exploring the state of instruction tuning on open resources. *arXiv preprint arXiv:2306.04751*.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023c. Self-instruct: Aligning language models with self-generated instructions.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Gary Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Maitreya Patel, Kuntal Kumar Pal, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Shailaja Keyur Sampat, Savan Doshi, Siddhartha Mishra, Sujan Reddy, Sumanta Patro, Tanay Dixit, Xudong Shen, Chitta Baral, Yejin Choi, Noah A. Smith, Hannaneh Hajishirzi, and Daniel Khashabi. 2022. Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks.

Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2022. Finetuned language models are zero-shot learners.

Kai Wei, Rishabh Iyer, and Jeff Bilmes. 2015. Submodularity in data subset selection and active learning. In *International conference on machine learning*, pages 1954–1963. PMLR.

Hanwei Xu, Yujun Chen, Yulun Du, Nan Shao, Yanggang Wang, Haiyu Li, and Zhilin Yang. 2022. Zeroprompt: Scaling prompt-based pretraining to 1,000 tasks improves zero-shot generalization.

Hu Xu, Saining Xie, Po-Yao Huang, Licheng Yu, Russell Howes, Gargi Ghosh, Luke Zettlemoyer, and Christoph Feichtenhofer. 2023. Cit: Curation in training for effective vision-language data. *arXiv preprint arXiv:2301.02241*.

Jifan Zhang, Gregory Canal, Yinglun Zhu, Robert D Nowak, Yifang Chen, Arnav M Das, Gantavya Bhatt, Stephen Mussmann, Jeffrey Bilmes, Simon S Du, et al. 2023. Labelbench: A comprehensive framework for benchmarking adaptive label-efficient learning. *arXiv preprint arXiv:2306.09910*.

Jifan Zhang, Julian Katz-Samuels, and Robert Nowak. 2022. Galaxy: Graph-based active learning at the extreme. In *International Conference on Machine Learning*, pages 26223–26238. PMLR.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. 2023. Lima: Less is more for alignment.