# Few-shot initializing of Active Learner via Meta-Learning

**Zi Long Zhu, Vikrant Yadav, Zubair Afzal, George Tsatsaronis**
Elsevier, the Netherlands
{z.zhu,v.yadav,m.afzal.1,g.tsatsaronis}@elsevier.com

## Abstract

Despite the important evolutions in few-shot and zero-shot learning techniques, domain specific applications still require expert knowledge and significant effort in annotating and labeling a large volume of unstructured textual data. To mitigate this problem, active learning and meta-learning attempt to reach a high performance with the least amount of labeled data. In this paper, we introduce a novel approach to combine both lines of work by initializing an active learner with meta-learned parameters obtained through meta-training on tasks similar to the target task during active learning. In this approach we use the pre-trained *BERT* as our text-encoder and meta-learn its parameters with *LEOPARD*, which extends the model-agnostic meta-learning method by generating task dependent softmax weights to enable learning across tasks with different number of classes. We demonstrate the effectiveness of our method by performing active learning on five natural language understanding tasks and six datasets with five different acquisition functions. We train two different meta-initializations and we use the pre-trained *BERT* base initialization as baseline. We observe that our approach performs better than the baseline at low budget, especially when closely related tasks were present during meta-learning. Moreover, our results show that better performance in the initial phase, i.e., with fewer labeled samples, leads to better performance when larger acquisition batches are used. We also perform an ablation study of the proposed method, showing that active learning with only the meta-learned weights is beneficial and adding the meta-learned learning rates and generating the softmax have negative consequences for the performance.

## 1 Introduction

In recent years, transformer-based models such as *BERT* (Devlin et al., 2018) have been very successful in achieving high performance in natural language processing (NLP) tasks. These results are achieved by training with a significant amount of labeled data, which is often necessary to optimize a large number of weights in these types of models during the fine-tuning stage. This is a major obstacle because many machine learning applications lack widely available labeled data and the labeling task in high volumes can be tedious and expensive.

Two principal research areas to overcome this obstacle are *Active Learning* (AL) and *Few-Shot Learning* (FSL). Few-shot learning was initially introduced to simulate the human ability to generalize quickly with only a few labeled examples (Yip and Sussman, 1997). Thus, the goal is to reach the highest possible performance with a small number of labelled data points (e.g., $4$, $8$, $16$, $\dots$). The field has made great progress after the introduction of optimization-based few-shot learning (Ravi and Larochelle, 2016) using the idea of *meta-learning* (ML) (Schmidhuber et al., 1997). The basic principle of meta-learning in this context is to allow the neural network to utilize the knowledge acquired from multiple tasks, represented in the network by its weights, for adaptation in new tasks. Hence, initializing the network with weights learned from a variety of tasks can enable faster learning of similar tasks.

The active learning field approaches this problem by only partially annotating the unlabeled data while attempting to achieve the highest performance. This is done by iteratively selecting a subset of unlabeled data points to be annotated by an *"oracle"* such that the selected points offer the highest learning benefit according to some metric, e.g., representativeness, diversity, uncertainty (Cohn et al., 1996).

In this work, we introduce a novel method to extend active learning with meta-learning to minimize the number of new data points that need to be annotated for achieving good performance. We do this by learning a favorable model initialization,

*Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 1117–1133
December 7-11, 2022 ©2022 Association for Computational Linguistics

via meta-learning, for the target task during active learning. We show a general approach on what to transfer to the active learning model and what to leave out. We demonstrate the effectiveness of our methodology by showing that it achieves higher performance than the baseline initialization on different natural language understanding tasks and datasets with the same number of annotation queries to the oracle; or eventually performs equally but by requiring fewer queries to the oracle. These results also show that our approach offers an advantage during active learning when larger annotation queries are used. Importantly, we show that performance improvement is significantly enhanced when tasks based on similar principle are available, especially in a cold-start setting.

## 2 Related Work

Active learning research focuses on developing novel acquisition functions for selecting data points to be annotated by the oracle. The primary metrics on which the acquisition functions act are uncertainty (Gal et al., 2017), diversity (Zhdanov, 2019), and representativeness (Sener and Savarese, 2017). Furthermore, acquisition functions that incorporate a multitude of metrics achieve better performance (Yuan et al., 2020; Margatina et al., 2021). Nevertheless, there is no capture function that consistently performs the best across different datasets or query sizes (Dor et al., 2020; Citovsky et al., 2021). Which is why we implemented different acquisition functions for our methodology.

Active learning and meta-learning have been used in the same context before, but the role of meta-learning has been treated as an acquisition function (Contardo et al., 2017; Fang et al., 2017). Several of these works show that meta-learned acquisition functions are effective in a cold-start environment (Konyushkova et al., 2017; Shao et al., 2019), which is mainly applied in computer vision. In this work, meta-learning is used to initialize the active learner and we show that it is effective in a cold-start environment and in a low-budget setting.

There has been one similar work from Barrett and White (2021), in the context of chemical peptide design. The mentioned work focuses on twelve different, but closely related tasks and uses meta-learning to optimize the initial parameters for active learning of the twelve tasks. However, the methodology is not entirely explained. In the current work, we do give a clear methodology of combining meta-learning and active learning, which is broadly applicable in the machine learning domain. We show this within the NLP context using a wide range of tasks.

A somewhat similar approach is using domain adaption to improve active learning (Rai et al., 2010; Su et al., 2020). The difference with our proposed method is that meta-learning is able to learn an initialization for fast adaptation using multiple tasks and domains. On the other hand domain adaptation is only able to learn common features between two domains in the same task and is therefore not comparable.

## 3 Method

### 3.1 Meta-Learning

To obtain a favorable initialization, we train a singular model, denoted $f$, over a set of tasks $\mathcal{T}$ using few-shot meta-learning. Each task $T$ consists of $k$-shot $n$-way mini-datasets, meaning each mini-dataset $\mathcal{D}$ consists of $n$ classes with each class containing $k$ samples.

The specific meta-learning method we employed is called LEOPARD, introduced by Bansal et al. (2019). LEOPARD is a BERT-based application of the Model-Agnostic Meta-Learning (MAML) (Finn et al., 2017) algorithm, which is a model-independent second-order optimization-based meta-learning method. The algorithm considers the model $f_\theta$ with parameters $\theta$. These parameters $\theta$ are updated in an inner- and outer loop. In the inner-loop, $\theta$ is updated into $\theta'_i$, by performing gradient descent with $\mathcal{D}_i^{tr} \sim T_i$:

$$\theta'_i = \theta - \alpha \nabla_\theta \mathcal{L}_{T_i}(f_\theta, \mathcal{D}_i^{tr}) \qquad (1)$$

This update is performed on $i = 1, 2, \ldots, t$ tasks, this collection of tasks is denoted as a meta-batch $\mathcal{B}$. Each task $T_i$ in $\mathcal{B}$ is selected using a predefined distribution $T_i \sim \mathcal{P}(\mathcal{T})$. In the outer loop, the parameters $\theta$ are updated via the meta-objective, where the goal is to minimize the sum of the error on $\mathcal{D}_i^{val} \sim T_i$ for each task in the inner loop:

$$\theta \leftarrow \theta - \beta \nabla_\theta \sum_{T_i \in \mathcal{B}} \mathcal{L}_{T_i}(\theta'_i, \mathcal{D}_i^{val}) \qquad (2)$$

Using the inner- and outer-loop, MAML is able to learn a favorable initialization for few-shot adaptation.

However, a non-trivial problem is how to apply different tasks within MAML, since not every task has the same number of classes. In the work of Bansal et al. (2019), they overcome this issue by generating task-dependent softmax parameters. This is realised by partitioning a mini-dataset $\mathcal{D}_i$ from $T_i$ with $N_i$ classes, where each partition $C_i^n$ contains samples $x_j$ from a corresponding class $n \in [N_i]$. Each class partition is fed into the text-encoder (BERT) $h_\theta$ and then the encoded samples undergo a non-linear projection $g_\phi$. Resulting in a representation for class $n$:

$$w_i^n, b_i^n = \frac{1}{|C_i^n|} \sum_{x_j \in C_i^n} g_\phi(h_\theta(x_j)) \tag{3}$$

Here, $g_\phi$ is a simple Multi-Layer Perceptron (MLP), where the parameters $\phi$ are meta-learned according to the MAML algorithm. The softmax parameters are then constructed by concatenating the class representation in eq. (3), as such:

$$\boldsymbol{W}_i = [w_i^1; \ldots; w_i^{N_i}] \quad \boldsymbol{b}_i = [b_i^1; \ldots; b_i^{N_i}] \tag{4}$$

These parameters are further adjusted during the inner adaptation loop in MAML. Another extension LEOPARD implements is meta-learning the learning rates (Li et al., 2017) in the inner loop, i.e., $\alpha$ in eq. (1), on a per layer basis. For further details, we refer the reader to Bansal et al. (2019).

### 3.2 Active Learning

The scenario we consider is pool-based sampling AL, where a large data pool is readily available for a given task $\mathbb{T}$. It consists of an initial small set of labeled data $L$ of seed size $s$ and a large pool of unlabeled data $U$. In each AL iteration, a model $f$ is trained on $L$, then using some acquisition function $a$ a batch $Q$ of size $q$ is selected from $U$. The acquired samples are then annotated by some oracle and added to the $L$. In the next AL iteration, model $f$ is retrained from its initial parameter initialization with the new $L$. Retraining is done from scratch to avoid overfitting from data samples in previous AL iterations (Hu et al., 2019).

We considered the following acquisition functions for our experiments:

- *Random*: $q$ samples are selected randomly from the unlabeled pool $U$.

- *Entropy*: An uncertainty-based acquisition function that ranks all unlabeled samples by their predictive entropy (Lewis and Gale, 1994) according to the model $f_{\theta_L}$ trained on $L$, and then selects the $q$ highest-ranked samples.

- *BADGE*: Tries to select uncertain, yet diverse samples, by calculating gradient embeddings $g_x$, and then performs $k$-MEANS++ using $q$ centers on $g_x$ for each unlabeled sample $x$. The gradient embeddings contain information about model confidence and hidden representations. So, by clustering uncertainty and diversity are included (Ash et al., 2019).

- *ALPS*: Is a model $f$ independent acquisition function that selects samples by calculating surprisal embeddings $s_x$ for each unlabeled sample $x$. $s_x$ is calculated by using a pre-trained BERT to compute the masked language modeling (MLM) loss on 15% randomly chosen, unmasked tokens in $x$ by evaluating their true token labels via cross-entropy loss. Then $k$-MEANS clustering is performed on $s_x$ with $q$ centers and the closest unlabeled sample to each center is selected to be annotated (Yuan et al., 2020).

- *CAL*: Selects samples via contrastive examples by calculating the $KL$-divergence between an unlabeled sample $x$ and its neighbourhood, consisting of the four nearest labeled samples. Samples $x$ with the highest average $KL$-divergence are chosen for annotation (Margatina et al., 2021).

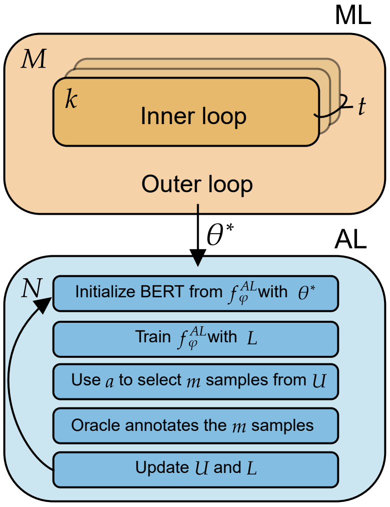

ML
$M$
$k$
Inner loop
$t$
Outer loop
$\theta^*$
AL
$N$
Initialize BERT from $f_\varphi^{AL}$with $\theta^*$
Train $f_\varphi^{AL}$with $L$
Use $a$ to select $m$ samples from $U$
Oracle annotates the $m$ samples
Update $U$ and $L$

Figure 1: A block diagram of our approach. We first meta-learn on set of tasks $\mathcal{T}$ and then initialize the active learner $f^{AL}$ with the meta-learned parameters.

### 3.3 Meta-initializing of Active Learner

The first step in our methodology is to choose a set of tasks $\mathcal{T}$ for meta-learning, that could benefit the target task $\mathbb{T}$ during active learning. We suggest picking several tasks for $\mathcal{T}$, preferably with substantial amounts of data, on how similar their objectives are and how close the vocabulary domains are to the target task. There are more rigorous ways to choose between different tasks for transfer learning (Poth et al., 2021).

Next is meta-learning the model $f_\theta^{ML}$, with the LEOPARD method as described in section. 3.1. The meta-learning model consists of three different layers:

$$f_\theta^{ML}(x) = \text{softmax}\{\boldsymbol{W}_i h_{\theta_P}(h_{\theta_B}(x)) + \boldsymbol{b}_i\} \quad (5)$$

where $h_{\theta_B}$ is a pretrained BERT-base encoder (Devlin et al., 2018) and $h_{\theta_P}$is a pre-classification layer consisting of two linear layers with a $\tanh$ activation function between them. $\boldsymbol{W}_i$ and $\boldsymbol{b}_i$ are generated by $g_\phi$ as in eq. (3) and (4), where $g_\phi$ has the same structure as $h_{\theta_P}$. The model is trained over $M$ meta-iterations, i.e., $M$ outer loops are performed which are characterized by eq. (2). For each meta-iteration, $t$ tasks are sampled for the inner loop, such that the probability of sampling a task is the proportion of the square root of each task size. For each task $T_i$, one $\mathcal{D}_i^{gen}$ is sampled to generate the softmax weights with $g_\phi$, $m$ mini-datasets $\mathcal{D}_i^{tr}$ for iteratively updating $\theta_i'$, and one $\mathcal{D}_i^{val}$ for the meta-objective. When the model is finished meta-learning, we choose the model $f_\theta^{ML}$ with the highest average accuracy across all tasks in $\mathcal{T}$. The weights of this model are denoted by $\theta^*$.

The active learning model $f_\varphi^{AL}$ has a similar architecture as $f_\theta^{ML}$:

$$f_\varphi^{AL}(x) = \text{softmax}\{h_{\varphi_C}(h_{\varphi_P}(h_{\varphi_B}(x)))\} \quad (6)$$

The difference is that the $f_\varphi^{AL}$ does not generate its softmax parameters, instead it has a basic classification layer $h_{\varphi_C}$ which is a simple linear layer. Here we only transfer the weights $\theta_P^*$ and $\theta_B^*$ from $f_{\theta^*}^{ML}$ to the corresponding layers in $f_\varphi^{AL}$:

$$f_\varphi^{AL}(x) = \text{softmax}\{h_{\varphi_C}(h_{\theta_P^*}(h_{\theta_B^*}(x)))\} \quad (7)$$

Figure 1 visualizes the initialization step. The decision to not include the generated softmax layer is due to its dependence on the availability of data samples of every class in a task. Otherwise, the weights created with eq. (4) will have incorrect dimensions, since it lacks representations for one or more classes. This scenario is probable when the initial size of $L$ is small and where the tasks have a higher number of classes. Additionally, generated softmax weights have a bias toward the samples used to generate them, which can be detrimental to the performance. Lastly, during active learning, there is only one target task $\mathbb{T}$, so the main advantage of learning across multiple tasks is unnecessary. Unless it's a multi-task active learning scenario (Ikhwantri et al., 2018).

The meta-learned learning rates are also not included, because they are trained in a strictly controlled environment with balanced batches, where each class is represented evenly and they are optimized for fast adaption using only a few examples. There is no guarantee that these learning rates are optimal for larger amount of data and if the batches are more random. In fact, these learning rates can become negative during meta-training (Starshak, 2022). This is beneficial for meta-learning because according to Starshak (2022) it pushes parameters with negative learning rates to learn universal features. However, it is obvious that during adaptation to the target task, positive learning rates are required to learn (Bernacchia, 2021), but there is no clear strategy on how to change the negative learning rates for adaptation. For these reasons, we opt for a standard learning rate $\alpha^{AL}$ for all parameters.

In the ablation study, we compared the performance between including and excluding the generated softmax weights and the meta-learned learning rates.

## 4 Experiments

### 4.1 Training Tasks

For our experiments, we train two different meta-initalizations with different sets of tasks. The first initialization is trained with a set of tasks $\mathcal{T}$ consisting of GLUE benchmark tasks: MNLI(m/mm), SST-2, QNLI, QQP, MRPC, and RTE (Wang et al., 2018) and the SNLI dataset (Bowman et al., 2015). These diverse tasks have been considered to be valuable for general language understanding and have been useful for transfer and few-shot learning across multiple domains (Poth et al., 2021; Bansal et al., 2019) [1].

The tasks in $\mathcal{T}$ provide general linguistic knowledge, however for topic classification it is essential to extract relevant keywords or phrases for a specific topic. By adding topic classification tasks the

[1] These are the same tasks used in Bansal et al. (2019), however, we do not modify SST-2 to learn phrase-level classification.

model can learn a pooling strategy to create a vector that represent these keywords or phrases for classification. Our second set of tasks $\mathcal{T}^{topic}$, therefore includes topic classification tasks by swapping QNLI and MRPC for Yahoo! Answers and DBPedia (Zhang et al., 2015). Yahoo! Answers is a question and answer dataset with topic classes and we use the question title for classification. DBPedia is a dataset created with Wikipedia articles from 14 different topics.

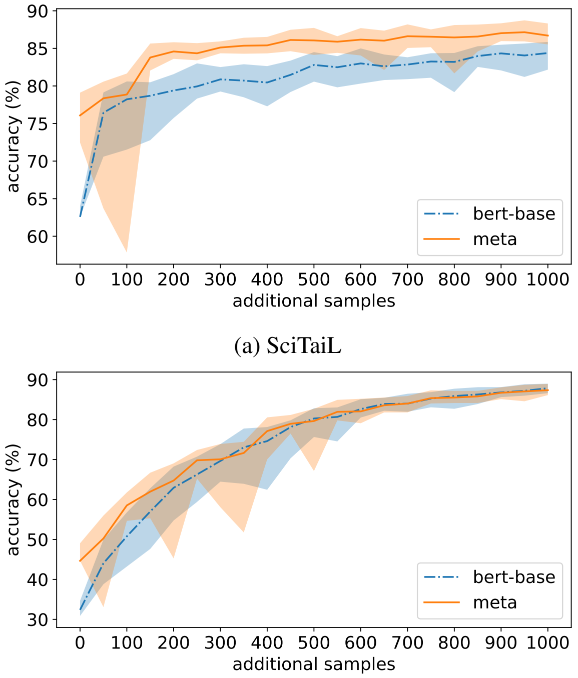
### 4.2 Evaluation and baselines

To evaluate the meta-initialization with $\mathcal{T}$, we perform active learning on SciTaiL, (Khot et al., 2018) (Saravia et al., 2018), AG news, Yelp Review (Zhang et al., 2015), Amazon Kitchen Reviews[2], and TREC (Li and Roth, 2002). We modify the Amazon Kitchen dataset to transform it into a sentiment analysis task by classifying the 1- and 2-star reviews as negative, the 4- and 5-star reviews as positive, and filtering out the 3-star reviews since their sentiment is ambiguous. We keep the Yelp review dataset unchanged as a rating task. We have AG News and TREC as topic classification tasks, using TREC's fine-grained labels. SciTaiL is an NLI task in the scientific domain and Emotion is a task that classifies sentences by their emotion. All training and evaluation datasets were downloaded from the HuggingFace online datasets repository[3]. In Appendix A are the dataset statistics in Table 2 and 3.

We use the pretrained *BERT*-base initialization [4] $\theta_{BERT}$ from Devlin et al. (2018) as our baseline for all evaluation tasks and we also compare the performance of meta-initialization $\mathcal{T}^{topic}$ on the topic classification tasks. For all target task $\mathbb{T}$ and initializations, we perform active learning with the five acquisition functions mentioned in section 3.2: *random*, *entropy*, *BADGE*, *ALPS* and *CAL*.

### 4.3 Implementation Details

For training the two meta-based initialization $\theta^*_{\mathcal{T}}$ and $\theta^*_{\mathcal{T}^{topic}}$, we use $M = 100K$ meta-iterations (i.e. outer loops), $t = 4$ task per meta-iteration, $m = 7$ mini-datasets $\mathcal{D}_i^{tr}$ is sampled for each $T_i$ in the inner-loops. For all tasks, we always classify between every pair of labels, even for tasks with more than 2 labels (Bansal et al., 2019). Therefore, each mini-dataset $\mathcal{D}$ consists of $n = 2$ classes, with each class having $k = 5$ examples. The learning rate for the outer-loop is $\beta = 1e-5$ and the per-layer learning rates in the inner-loop are initialized with $\alpha = 1e-5$. These are the best hyper-parameters given in Bansal et al. (2019). The weights are updated using the Adam optimizer (Kingma and Ba, 2015) in the outer-loop and the inner-loop with SGD (Ruder, 2016). The meta-trained models with the highest average accuracy across all training tasks are chosen for meta-initializing the active learner.



(a) SciTaiL

(b) Emotion

Figure 2: Results for AL on with $s = 20$ and $q = 50$ showing the mean accuracy (%). The mean is the average accuracy across all acquisition functions. The shade is twice the std-dev, calculated separately for above and under the mean. The meta-initialization $\theta^*_{\mathcal{T}}$ outperforms the baseline SciTaiL consistently by a larger margin. Whereas, for Emotion, $\theta^*_{\mathcal{T}}$ only outperforms the baseline for the first 300 additional samples.

During active learning, we consider a low-budget scenario, where a maximum of 1000 additional annotated data samples are acquired. We examine several set-ups with seed data $s = 20, 50, 100$ and acquisition sizes $q = 50, 100$ , constraining the number of AL iterations to 20 and 10, respectively. Any results not shown in Section 5, are shown in Appendix C. In each AL iteration, we train each model for 25 epochs on a given target task $\mathbb{T}$ and choose the model with the highest accuracy on the validation set to evaluate on the test set. We do this

[2] We use the online HuggingFace version from this dataset.

[3] https://huggingface.co/datasets

[4] Pretrained BERT is used off-the-shelf (no fine-tuning).

|  |  |  | additional acquired samples |  |  |  |  |  |
|---|---|---|---|---|---|---|---|---|
| Task | s | $a^*$ | 0 | 200 | 400 | 600 | 800 | 1000 |
| Scitail | 20 | $B^{base}$ | 62.57±13.89 | 79.82±3.32 | 81.90±2.14 | 83.17±0.47 | 83.24±2.32 | 85.80±1.03 |
|  |  | $B^{meta}$ | **76.08**±8.88 | **83.85**±1.35 | **85.76**±2.15 | **86.13**±2.44 | **87.13**±1.26 | **87.14**±1.49 |
|  | 50 | $B^{base}$ | 75.86±8.30 | 80.30±2.71 | 80.22±4.87 | 84.40±2.89 | 84.33±0.91 | 84.66±0.97 |
|  |  | $B^{meta}$ | **78.27**±14.57 | **84.45**±2.95 | **86.40**±1.37 | **86.98**±2.07 | **86.96**±1.51 | **86.96**±1.87 |
|  | 100 | $B^{base}$ | 80.48±1.87 | 82.41±1.26 | 83.15±0.82 | 84.39±2.38 | 83.95±3.22 | 84.57±1.90 |
|  |  | $B^{meta}$ | **83.71**±1.88 | **84.86**±3.98 | **86.66**±1.71 | **86.92**±1.07 | **85.87**±3.43 | **88.05**±2.51 |
| Amazon | 20 | $B^{base}$ | 53.82±6.38 | 80.16±8.95 | 88.91±2.68 | 88.78±1.67 | 89.21±0.62 | 90.08±3.28 |
|  |  | $B^{meta}$ | **72.35**±15.27 | **86.76**±3.70 | **89.04**±1.89 | **89.64**±0.86 | **90.40**±2.44 | **90.57**±0.56 |
|  | 50 | $E^{base}$ | 52.27±1.76 | 85.86±7.58 | 87.21±3.34 | 89.01±1.35 | 89.35±1.83 | **89.82**±2.05 |
|  |  | $E^{meta}$ | **76.56**±16.99 | **87.86**±3.11 | **88.57**±1.15 | **89.84**±2.39 | **90.14**±2.07 | 89.69±2.11 |
|  | 100 | $E^{base}$ | 64.60±24.02 | 85.68±2.31 | 85.82±4.28 | 88.26±2.58 | **91.00**±0.89 | 89.76±1.22 |
|  |  | $E^{meta}$ | **74.31**±16.46 | **87.03**±2.37 | **89.81**±1.87 | **88.81**±2.67 | 90.23±1.44 | **90.84**±1.37 |
| Yelp | 20 | $A^{base}$ | 23.28±6.09 | 41.56±7.90 | **49.30**±2.91 | **51.12**±2.29 | **51.33**±0.70 | **52.38**±2.41 |
|  |  | $A^{meta}$ | **33.82**±5.50 | **47.16**±2.74 | 47.08±3.71 | 50.13±1.04 | 50.53±1.30 | 51.21±2.42 |
|  | 50 | $A^{base}$ | 24.62±2.22 | 43.60±4.04 | 49.71±0.34 | **50.65**±2.36 | **52.44**±0.85 | **51.96**±2.61 |
|  |  | $A^{meta}$ | **37.81**±3.97 | **48.21**±1.53 | **49.72**±3.29 | 50.31±1.95 | 51.66±1.78 | 51.61±0.70 |
|  | 100 | $A^{base}$ | 31.39±6.93 | 47.36±4.50 | **50.07**±1.07 | **51.40**±1.01 | **51.59**±1.00 | **52.68**±2.18 |
|  |  | $A^{meta}$ | **39.43**±7.56 | **47.44**±3.26 | 49.63±1.76 | 50.30±2.12 | 50.62±1.39 | 51.60±1.27 |
| Emotion | 20 | $B^{base}$ | 32.38±5.36 | 66.72±7.64 | 74.05±4.92 | 84.94±2.51 | 87.41±0.43 | 88.80±0.71 |
|  |  | $B^{meta}$ | **44.65**±15.80 | **69.88**±5.47 | **81.11**±3.59 | **85.17**±2.84 | **87.63**±1.36 | **89.48**±0.58 |
|  | 50 | $C^{base}$ | 41.11±6.46 | 66.26±4.35 | 79.21±2.66 | **84.76**±3.18 | 87.55±1.26 | 88.97±0.72 |
|  |  | $C^{meta}$ | **52.28**±1.98 | **70.47**±6.00 | **81.63**±1.57 | 84.68±2.72 | **88.37**±1.24 | **89.46**±2.09 |
|  | 100 | $C^{base}$ | 47.53±10.44 | 70.05±5.84 | **80.79**±2.34 | 85.17±2.14 | 87.58±1.32 | **89.88**±1.87 |
|  |  | $C^{meta}$ | **56.99**±6.12 | **70.82**±10.37 | 80.75±4.40 | **86.74**±2.33 | **88.32**±1.12 | 89.70±1.11 |

Table 1: AL performance between the base and meta-learned initialization across different tasks and seed sizes, with acquisition size of $q = 50$. The table shows the accuracy at $0, 200, 400, 600, 800$ and $1000$ additionally acquired samples for the best performing acquisition function $a^*$ on the base initialization in each specific scenario. Where $R = random, E = entropy, B = BADGE, A = ALPS$ and $C = CAL$. The table shows the performance under an increase of annotated sample size.

for 4 different seeds: 41, 43, 47, and 53. The learning rate used is $a^{AL} = 2 - e5$ and the weights are updated using the AdamW optimizer (Loshchilov and Hutter, 2019) with epsilon $1e - 8$. If the training set is larger than $100K$ samples we down-size it to $20K$ to lessen the computational load. For every dataset we used a maximum sequence length of 128. If there is no validation set or test set available we randomly sample $10\%$ or $20\%$ from the training set, respectively.

# 5 Results

## 5.1 Performance on NLP tasks

In Table 1 we present the results on the SciTaiL, Amazon, Yelp, and Emotion datasets with $q = 50$, with the acquisition function $a^*$ that performed the best on the base initialization. This is to showcase how the meta-initialized model performs in comparison with the baseline over different seed sizes and additionally acquired samples. In most of the scenarios, the meta-initialized model outperforms the baseline and in the remaining, it has a very similar performance to the baseline. To see the full results see Appendix C.

The $\theta^*_{\mathcal{T}}$ initialization outperforms the baseline on SciTaiL and Amazon datasets consistently as shown in Table 1. Since $\mathcal{T}$ contains four NLI tasks (MNLI, QNLI, RTE, and SNLI), $\theta^*_{\mathcal{T}}$ must have learned the semantic relationships required for the NLI task. As a result, the difference between the two initialization decreases slowly, but stays significantly larger, as can be observed in Figure 2a. Interestingly, the same is true for the Amazon dataset, while $\mathcal{T}$ only contains one sentiment classification task (SST-2). It seems that SST-2 in combination with general NLU tasks provides sufficient information to learn how to discriminate between negative and positive sentiment.

The same is less pronounced for Yelp and Emotion. After about 400 to 600 annotated samples, the gap between the two initializations becomes smaller and eventually negligible, as shown in Figure 2b for Emotion. $\mathcal{T}$ does not contain a task that is in the same domain as Yelp or Emotion. However, SST-2 is somewhat related because Yelp is a fine-grained sentiment classification task, and emotions are often expressed in terms of positive and negative emotions. Therefore, $\mathcal{T}$ only provides the model with information about intermediate features needed for the downstream task of Yelp and Emotion.

The key observation is that the meta-initialization $\theta^*_{\mathcal{T}}$ always performs better than the baseline on average for 200 or fewer samples sampled, and almost always for 600 or fewer acquired samples. Especially, when less seed data is available. This shows that meta-initialization provides a competitive advantage over baseline in a low-budget and cold start setting by being able to learn in a few-shot steps.

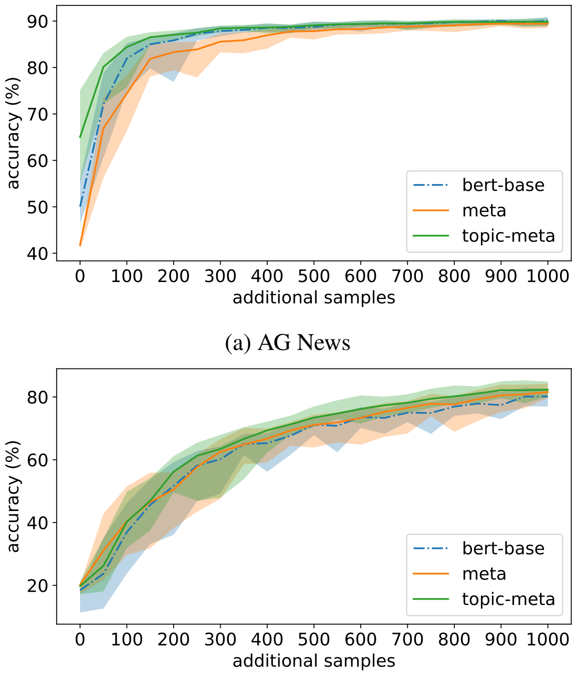
However, the difference in performance shrinks as more samples are acquired. This trend is expected because the ability to learn rapidly is the most useful when less information is available. When enough information is obtained the same performance can be achieved by learning with a large number of data. Crucially, this means that learning fast with $\theta^*_{\mathcal{T}}$ is not detrimental when a large number of data is available. This is generally what Table 1 shows, when the base initialization performs better at 600 or more acquired samples, often by a small margin

Additionally, learning slows down for both when the proportion $\frac{q}{|L|}$ becomes smaller. Each newly acquired batch of annotated sample provides less and less new information as $L$ grows larger, which might mask some difference in performance between the two initialization at larger $L$.

We also notice that the best acquisition function for the baseline $\theta_{BERT}$ mostly coincides with the best acquisition function for $\theta^*_{\mathcal{T}}$. It is an indication that selecting the best acquisition function is mostly dataset dependent.

## 5.2 Topic Classification

The set of tasks $\mathcal{T}$ does not contain any topic classification tasks or any related tasks, consequently in Figure 3, it shows that $\theta^*_{\mathcal{T}}$ is actually detrimental for AG news and provides no significant advantage towards TREC. So, for the proposed method it is essential to have similar or related tasks for meta-training the initialization.



(a) AG News

(b) TREC

Figure 3: Results for AL with $s = 20$ and $q = 50$ showing the mean accuracy (%) of all acquisition functions. The baseline outperforms the meta-initialized model. However, the initialization with $\theta^*_{\mathcal{T}_{Topic}}$ outperforms the baseline. Showing the importance of task selection.

This necessity is clearly portrayed in Figure 3, since by active learning with $\theta^*_{\mathcal{T}_{Topic}}$ the performance is improved by a significant amount for both tasks, by adding two topic classification task during training. TREC performs increasingly better than $\theta^*_{\mathcal{T}}$ and $\theta_{BERT}$ as the number of annotated samples grows. For AG news the difference in performance is immediately noticeable at low amounts of annotated samples, due to Yahoo and DBPedia containing similar topics as AG news. Although, the performance reaches a plateau at around 500 additional samples, most likely due to new samples not providing a critical amount of information as mentioned in Section 5.1.

### 5.3 Query size

In the previous two sections we have only considered the results with a query size of $q = 50$. When compared to the results with $q = 100$ another advantage is exposed. If we compare Figure 2 and Figure 4 , we see that $\mathcal{T}$ with $q = 100$ perform consistently better than $\theta_{BERT}$, as opposed to $q = 50$, where their performance become indistinguishable around 300-400 additional samples. This advantage is important when the amount of AL iterations is constraint (Bullard et al., 2019).

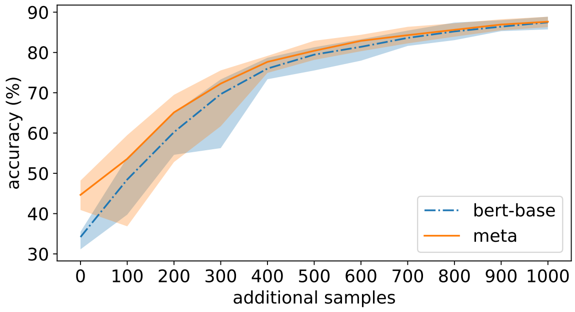


Figure 4: Results for AL on Emotion with $s = 20$ and $q = 100$ showing the mean accuracy (%) of all acquisition functions. The meta-learned initialization outperforms the baseline more consistently on Emotion.

The meta-initialization $\theta^*_{\mathcal{T}}$ works better with a larger $q$ compared to $\theta_{BERT}$, since the initial performance gain implies a better representation of the samples in the encoded space. This directly benefits the acquisition functions that depend on the encoded space, such as *CAL* . *BADGE* and *entropy* benefit indirectly, as their evaluation method is affected by how well the samples are represented by the model in action. Similarly, when using a more advanced text encoder architecture (Lu and MacNamee, 2020). In short, the model-dependent acquisition functions score unlabeled samples more accurately, and by accumulating better-annotated samples, the meta-initialised active learner outperforms the baseline at larger query sizes $q$.

## 6 Ablation study

### 6.1 Generated Softmax and Learning Rates

To see the effects of using the generated softmax weights and the meta-learned learning rates $\alpha$ during active learning, we perform the active learning experiment with these elements included and initialized with $\theta^*_{\mathcal{T}}$, where $s = 100$, $q = 50$ and the BADGE acquisition function. As mentioned in Section 3.3, $\alpha$ can become negative during meta-learning and Figure 5 shows that there are a high density of negative learning rates after meta-learning on $\mathcal{T}$. For these experiments, we set any negative learning rate to $\alpha = 2e - 5$.

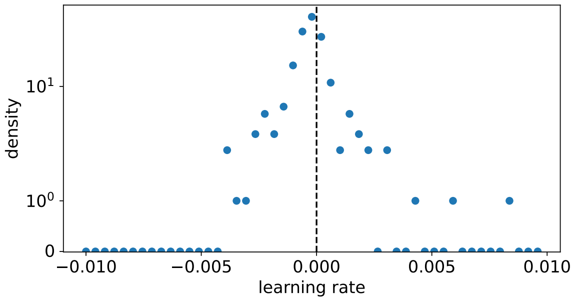


Figure 5: Meta-learned learning rates $\alpha$ from LEOPARD trained on $\mathcal{T}$ and shows negative learning rates.

The results are shown in Figure 6. Here, we observe that after an initial gain in performance, the model seems to stop to learn. The average performance fluctuates around accuracy of 83%. We suspect this is caused by learning with learning rates above $1e - 4$ (see Figure 5) with imbalanced batches during the start of training. For example, when there is a sufficient amount annotated samples its probable to have consecutive batches which all consist of samples of the same class. Thus, pushes the active learner rapidly towards biased learning, that it cannot even recover when trained with a large amount of samples. A solution could be learning with more balanced batches at the start of training, this we will leave as future work.

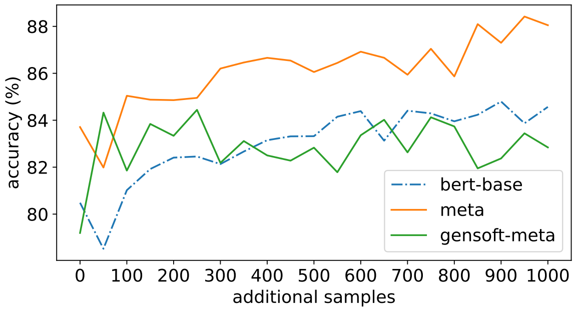


Figure 6: Results for AL on SciTaiL with $s = 100$ and query size $q = 50$ with BADGE, showing the mean accuracy (%) across four runs. Gensoft-meta is initialized with $\theta^*_{\mathcal{T}}$ and includes generated softmax weights and meta-learned learning rates. The gensoft-meta performance does not improve with more annotated samples.

### 6.2 Additional Comparisons

To further demonstrate the effectiveness of meta-learning the initial parameters for active learning, we provide two additional baselines. In the experiments of Section 5, the baseline initialization was used off-the-shelf and has not seen any data from tasks $\mathcal{T}$ used in meta-learning. Meaning that, the meta-learned initialization has seen 10 million more datapoints. To make a more even comparison, we fine-tune the baseline by performing Masked Language Model (MLM) training on the datapoints in task $\mathcal{T}$ to provide equal data support. Then, we perform active learning on SciTail with the entropy acquisition function with $s = 20$ and $q = 50$.

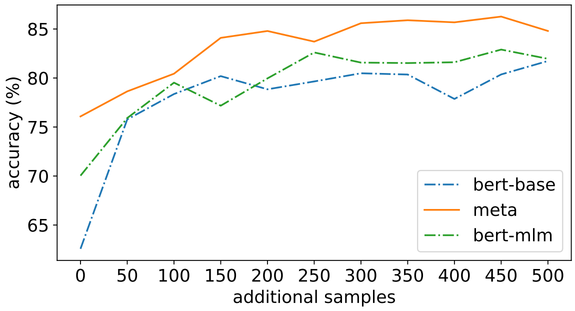


Figure 7: Results for AL on SciTail with $s = 20$ and query size $q = 50$ with entropy, showing the mean accuracy (%) across four runs. The MLM trained BERT initialization performs better than the baseline, but not the meta-learned initialization.

Figure 7 shows that the MLM trained initialization performs moderately better than the default baseline, but significantly performs worse than the meta-learned initialization. This shows that the performance gain by meta-learning an initialization is not simply due to a larger data support, but due to its ability to few-shot learn.

The second additional comparison we make, is fine-tuning the baseline initialization by pre-training on the DBPedia dataset and then perform active learning on AG News to show that our methodology is more beneficial than simple transfer learning. This experiment is performed with $s = 20$ and $q = 50$ with the entropy acquisition function.

We see that by simply pre-training on DBPedia the baseline is able to perform closely to the topic meta-initialization. However, if we put more emphasis (80%) on the topic tasks in $\mathcal{T}^{topic}$ we can outperform both initializations significantly, as shown by topic-meta80 in Figure 8. This shows that the distribution of tasks during meta-learning is an important factor that can impact the learning capabilities given a target task.

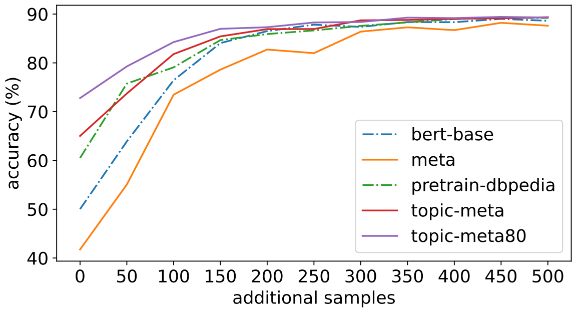


Figure 8: Results for AL on AG News with $s = 50$ and query size $q = 50$ with entropy, showing the mean accuracy (%) across four runs.

## 7 Conclusion

In this work, we have shown how to initialize the active-learner with meta-trained parameters on related tasks. This method provides a significant boost in performance in a low-budget setting. This effect is exaggerated during the early stages of active learning, due to fast adaptation capabilities by being able to few-shot learn. Furthermore, the ability to few-shot learn provides better representations in the encoded space, making the scoring from implemented acquisition functions more accurate. Which consequently, results in better active learning performance using larger acquisition batches.

## 8 Limitations

An obvious limitation is that the proposed method likely does not provide a significant advantage in performance when applied in a high-budget situation, where large amounts of data points are annotated. One way it might have a noticeable effect is to start with a smaller acquisition/query size and increase it during the active learning process, with the assumption that with a high budget, a larger acquisition size is used. This could leverage the better encoding space that few-shot meta-learning provides to improve the selection of samples to be annotated in the early stages. Possibly, resulting in a cumulative advantage in performance.

Another limitation is that closely related tasks might be nonexistent or rare. This would be detrimental, because if there are no related tasks in the meta-learning process, our method is worse than the BERT baseline, as shown with AG News.

## References

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*.

Trapit Bansal, Rishikesh Jha, and Andrew McCallum. 2019. Learning to few-shot learn across diverse natural language classification tasks. *arXiv preprint arXiv:1911.03863*.

Rainier Barrett and Andrew D White. 2021. Investigating active learning and meta-learning for iterative peptide design. *Journal of Chemical Information and Modeling*, 61:95 – 105.

Alberto Bernacchia. 2021. Meta-learning with negative learning rates. *ArXiv*, abs/2102.00940.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Kalesha Bullard, Yannick Schroecker, and S. Chernova. 2019. Active learning within constrained environments through imitation of an expert questioner. *ArXiv*, abs/1907.00921.

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.

David A Cohn, Zoubin Ghahramani, and Michael I Jordan. 1996. Active learning with statistical models. *Journal of artificial intelligence research*, 4:129–145.

Gabriella Contardo, Ludovic Denoyer, and Thierry Artières. 2017. A meta-learning approach to one-step active learning. *arXiv preprint arXiv:1706.08334*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Liat Ein Dor, Alon Halfon, Ariel Gera, Eyal Shnarch, Lena Dankin, Leshem Choshen, Marina Danilevsky, Ranit Aharonov, Yoav Katz, and Noam Slonim. 2020. Active learning for bert: an empirical study. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7949–7962.

Meng Fang, Yuan Li, and Trevor Cohn. 2017. Learning how to active learn: A deep reinforcement learning approach. *arXiv preprint arXiv:1708.02383*.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR.

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.

Peiyun Hu, Zachary Chase Lipton, Anima Anandkumar, and Deva Ramanan. 2019. Active learning with partial feedback. *ArXiv*, abs/1802.07427.

Fariz Ikhwantri, Samuel Louvan, Kemal Kurniawan, Bagas Abisena, Valdi Rachman, Alfan Farizki Wicaksono, and Rahmad Mahendra. 2018. Multi-task active learning for neural semantic role labeling on low resource conversational corpus. In *Proceedings of the Workshop on Deep Learning Approaches for Low-Resource NLP*, pages 43–50, Melbourne. Association for Computational Linguistics.

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2018. Scitail: A textual entailment dataset from science question answering. In *AAAI*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Ksenia Konyushkova, Raphael Sznitman, and Pascal Fua. 2017. Learning active learning from data. *Advances in neural information processing systems*, 30.

David D Lewis and William A Gale. 1994. A sequential algorithm for training text classifiers. In *SIGIR'94*, pages 3–12. Springer.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Zhenguo Li, Fengwei Zhou, Fei Chen, and Hang Li. 2017. Meta-sgd: Learning to learn quickly for few-shot learning. *arXiv preprint arXiv:1707.09835*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *ICLR*.

Jinghui Lu and Brian MacNamee. 2020. Investigating the effectiveness of representations based on pretrained transformer-based language models in active learning for labelling text datasets. *ArXiv*, abs/2004.13138.

Katerina Margatina, Giorgos Vernikos, Loïc Barrault, and Nikolaos Aletras. 2021. Active learning by acquiring contrastive examples. *arXiv preprint arXiv:2109.03764*.

Clifton Poth, Jonas Pfeiffer, Andreas Ruckl'e, and Iryna Gurevych. 2021. What to pre-train on? efficient intermediate task selection. In *EMNLP*.

Piyush Rai, Avishek Saha, Hal Daumé III, and Suresh Venkatasubramanian. 2010. Domain adaptation meets active learning. In *Proceedings of the NAACL HLT 2010 Workshop on Active Learning for Natural Language Processing*, pages 27–32.

Sachin Ravi and Hugo Larochelle. 2016. Optimization as a model for few-shot learning.

Sebastian Ruder. 2016. An overview of gradient descent optimization algorithms. *ArXiv*, abs/1609.04747.

Elvis Saravia, Hsien-Chi Toby Liu, Yen-Hao Huang, Junlin Wu, and Yi-Shin Chen. 2018. CARER: Contextualized affect representations for emotion recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3687–3697, Brussels, Belgium. Association for Computational Linguistics.

Jürgen Schmidhuber, Jieyu Zhao, and Marco Wiering. 1997. Shifting inductive bias with success-story algorithm, adaptive levin search, and incremental self-improvement. *Machine Learning*, 28(1):105–130.

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

Jingyu Shao, Qing Wang, and Fangbing Liu. 2019. Learning to sample: an active learning framework. In *2019 IEEE International Conference on Data Mining (ICDM)*, pages 538–547. IEEE.

Tom Starshak. 2022. Negative inner-loop learning rates learn universal features. *ArXiv*, abs/2203.10185.

Jong-Chyi Su, Yi-Hsuan Tsai, Kihyuk Sohn, Buyu Liu, Subhransu Maji, and Manmohan Chandraker. 2020. Active adversarial domain adaptation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 739–748.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Kenneth Yip and Gerald Jay Sussman. 1997. Sparse representations for fast, one-shot learning.

Michelle Yuan, Hsuan-Tien Lin, and Jordan Boyd-Graber. 2020. Cold-start active learning through self-supervised language modeling. *arXiv preprint arXiv:2010.09535*.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *ArXiv*, abs/1509.01626.

Fedor Zhdanov. 2019. Diverse mini-batch active learning. *arXiv preprint arXiv:1901.05954*.

| Dataset | Task | Labels | Training | Validation |
|---|---|---|---|---|
| MNLI$_{\text{(m/mm)}}$ | NLI | 3 | 392702 | 19647 |
| SST-2 | SA | 2 | 67349 | 872 |
| QNLI | QA/NLI | 2 | 10473 | 5463 |
| QQP | PP | 2 | 363846 | 40430 |
| MRPC | PP | 2 | 3668 | 408 |
| RTE | NLI | 2 | 2490 | 277 |
| SNLI | NLI | 3 | 550152 | 10000 |
| DBPedia | TC | 14 | 560000 | 70000 |
| Yahoo | TC | 10 | 1400000 | 60000 |

Table 2: Statistics for datasets used during training. NLI stands for Natural Language inference, SA for Sentiment Analysis, QA for Question and Answering, PP for ParaPhrase and TC for Topic Classification. The test set is excluded because it was not used during meta-learning. If there was no validation set we used the test set as validation set.

| Dataset | Task | Labels | Training | Validation | test |
|---|---|---|---|---|---|
| SciTaiL | NLI | 2 | 23097 | 1304 | 2126 |
| Amazon | SA | 2 | 4880466 | - | - |
| Yelp | FSA | 5 | 130000 | - | 10000 |
| Emotion | E | 6 | 16000 | 2000 | 2000 |
| AG News | TC | 4 | 120000 | - | 7600 |
| TREC | TC | 47 | 5452 | - | 500 |

Table 3: Statistics for datasets used during evaluation / Active Learning. The task shorthands are the same in Table 2, with one addition: FSA meaning Fine-grained Sentiment Analysis. '-' means that the set was not available

# A Data

## A.1 Statistics

# B Model Parameters and Computation Time

## B.1 Model Parameters

The model $f^{ML}$ and $f^{AL}$ both use the BERT-base text-encoder, which has roughly 110M parameters (Devlin et al., 2018). The pre-classification and softmax-generating layers consists of two linear transformation layers, with $768 \times 384 = 294912$ and $384 \times 256 = 98304$ parameters. The classification layer consists of two linear transformation layers with $256 \times |C|$ parameters, where $|C|$ is the number of classes. Lastly, $f^{ML}$ learns 209 learning rates for the inner loop.

### B.2 Computation Time

Training with LEOPARD is computationally expensive. For every task $\mathcal{T}_i$ the gradient needs to be calculated 8 times for all parameters, and therefore 32 times per meta-batch. The average time to train on one meta-batch $\mathcal{B}$ or meta-iteration with 4 tasks using LEOPARD is 3.8 seconds. Therefore, training with 100K meta-iterations takes roughly 105 hours. By using parallelization across the 4 tasks, the time should be able to be brought down to about 30 hours.

During AL, at most$|L^{max}| = 1100$ are available if a seed size $s = 100$ is used. A full epoch of training with $L^{max}$ annotated samples through $f^{AL}$ takes roughly 15.5 seconds. The largest dataset we perform our experiments on is SciTaiL, because other datasets above 100K are downsized to 20K as mentioned in Section 4.3. We measure the time it takes to acquire $q = 50$ samples for 20 iterations up to a 1000 additional annotated samples for each acquisition function with SciTaiL. The time to update the $L$ and $U$ and process the samples is included. For $random$ it takes 694 seconds, $entropy$ 1672 seconds, $BADGE$ 1794 seconds, $ALPS$ 1812 seconds and for $CAL$ 1762 seconds.

The training of the meta-learned models is run on a single Nvidia Tesla V100 GPU and each active learning experiment is run on four Nvidia Tesla V100 GPUs.

## C Additional Results

### C.1 General NLP tasks

The results for all acquisition functions from Table 1 for each $s = 20, 50, 100$ with $q = 50$ can be found in Table 4, 5 and 6, respectively. Similarly, we also show here the graphs for all tasks discussed in Section 5.1 with $s = 20$ and $q = 50$ in Figure 9. We observe the same trends as described in Section 5.1.

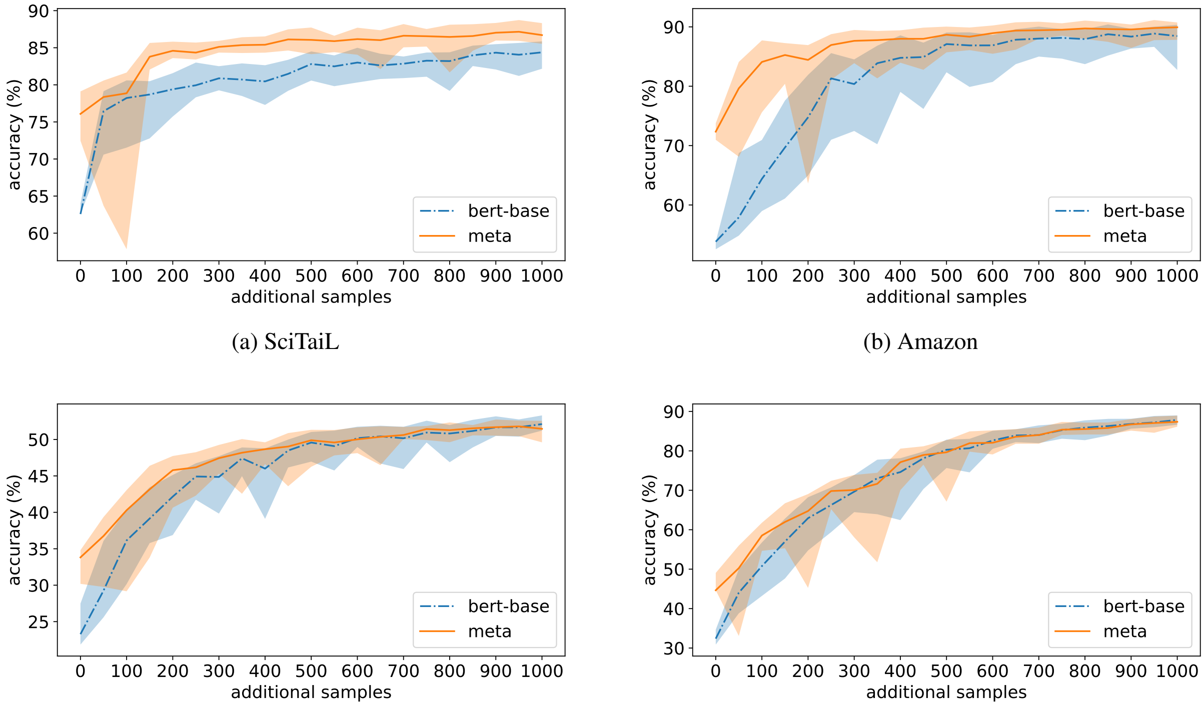
### C.2 Topic Classification

In Figure 10 and 11, we see that even starting with larger seed data that the meta-initialization $\theta^*_{\mathcal{T}_{Topic}}$ still outperforms the baseline as in Section 5.2. For AG News we observe that the performance becomes equal faster as the seed size becomes larger. However, the learning plateaus at around the same number of annotated samples. Indicating that the model is reaching its limit in performance or needs to learn with a larger $L$, e.g, $L = 2000, 4000$, to reach a significantly higher performance.

For TREC we observe in Figure 11 that the difference in performance stays constant for all seed sizes and even with a large amount of annotated samples. Showing that the meta-initialization $\theta^*_{\mathcal{T}_{Topic}}$ has gained knowledge on how to extract relevant keywords and phrases, therefore giving it a consistent performance advantage.

### C.3 Query Size

In Figure 12 we see that using a larger query size the meta-initialization $\theta^*_{\mathcal{T}}$ outperforms the baseline more consistently for SciTaiL and Emotion compared to in Figure 9. However, we do not see the same trend for Amazon and Yelp. For Yelp the reason might be that fine-grained sentiment classification task is too complex for the model to represent in its encoded space. $random$ and $ALPS$ often reach the highest performance, as shown in Table 4, 5 and 6. Showing that model-independent acquisition functions are preferable. This is an indication that the representations in the encoded space are unreliable. Therefore, the advantage of the meta-initialization the model at larger $q = 100$ does not materialize. For Amazon we do not observe a significant difference between the two scenarios, it might be that the task is too simple and the difference in the encoded space between meta-initialized active learner and the baseline are not noticeable when picking 50 or 100 samples.



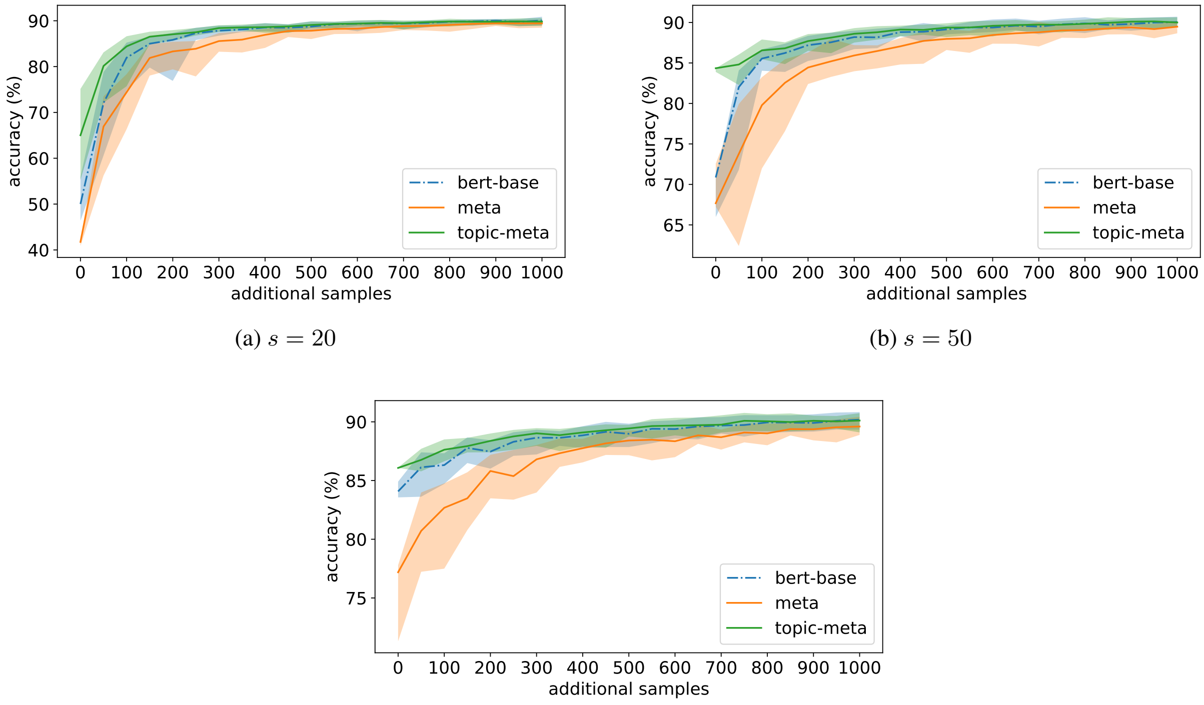
(a) SciTaiL

(b) Amazon

(c) Yelp

(d) Emotion

Figure 9: AL results on $s = 20$ and $q = 50$ showing the mean accuracy (%) across all acquisition functions. The meta-initialization in general outperforms the baseline, especially up to 300-400 additional annotated samples. For 400 or more the baseline and meta-initialization become or less equal, for Yelp and Emotion.

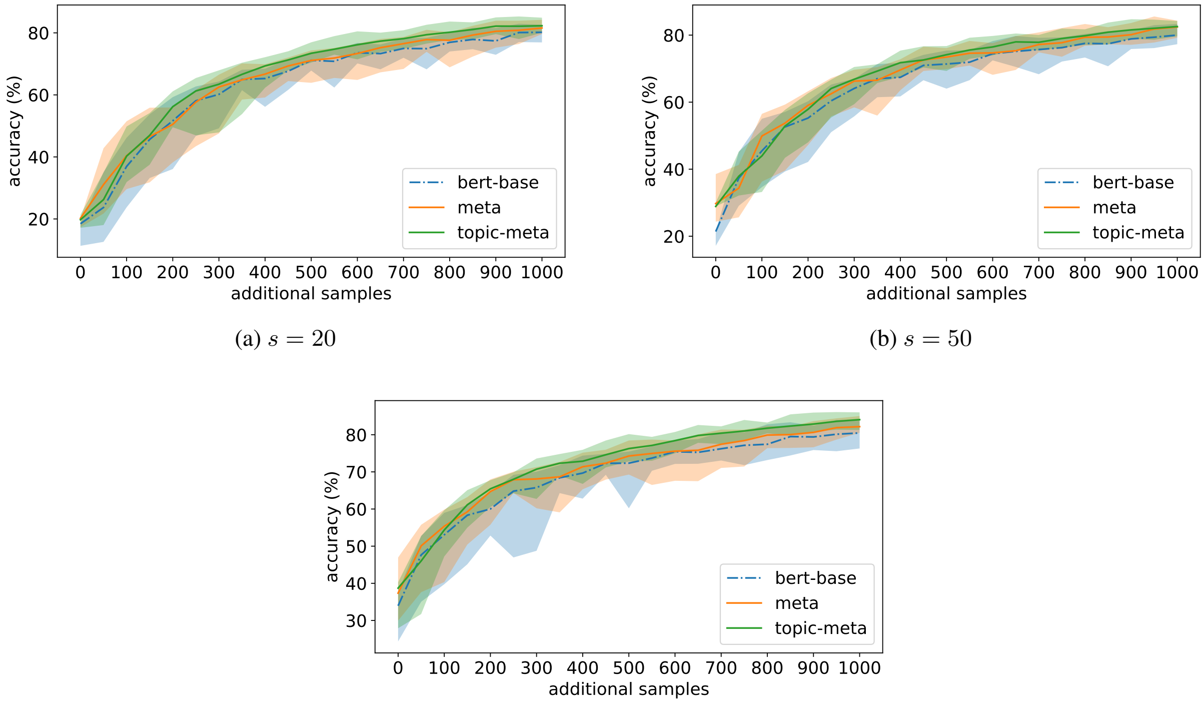


(a) $s = 20$

(b) $s = 50$

(c) $s = 100$

Figure 10: AL results for AG News with $q = 50$ and $s = 20, 50, 100$ showing the mean accuracy (%) across all acquisition functions. The meta-learned initialization with topic classification tasks outperforms the baseline, while without these tasks it performs worse.

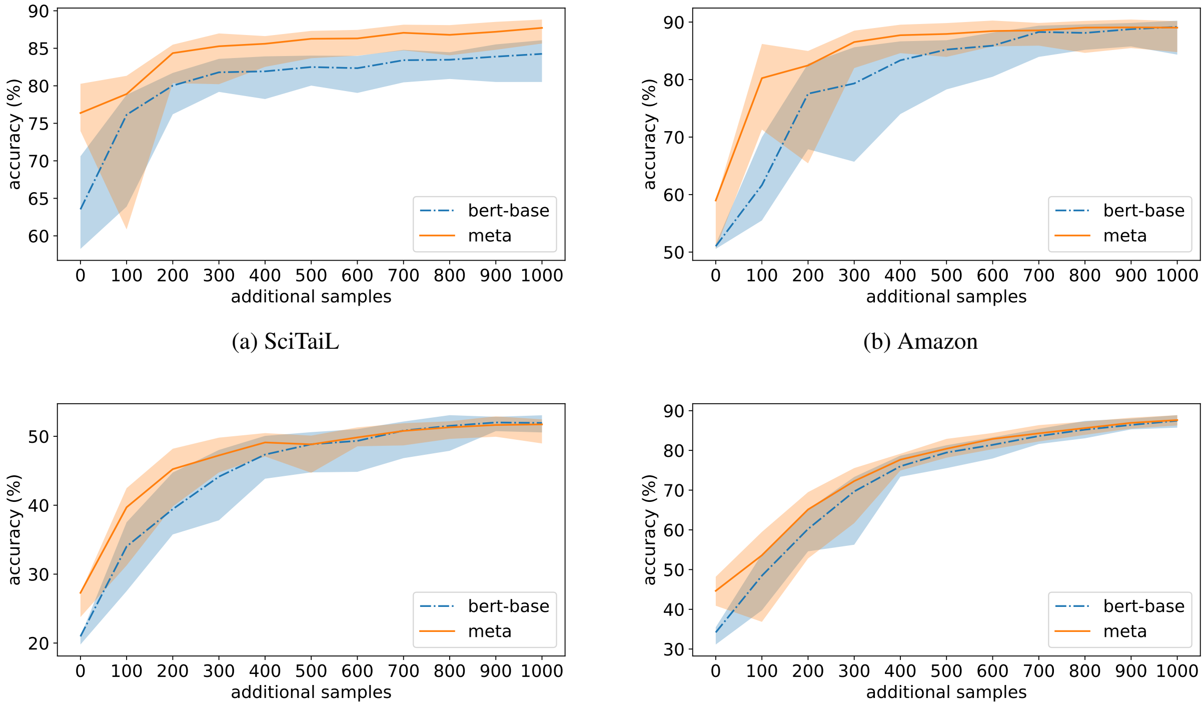


(a) $s = 20$

(b) $s = 50$

(c) $s = 100$

Figure 11: AL results for TREC with $q = 50$ and $s = 20, 50, 100$ showing the mean accuracy (%) across all acquisition functions. The meta-learned initialization with topic classification tasks outperforms the baseline, while without these tasks it performs worse.



(a) SciTaiL

(b) Amazon

(c) Yelp

(d) Emotion

Figure 12: AL results on $s = 20$ and $q = 100$ showing the mean accuracy (%) across all acquisition functions. The meta-initialization seems to have an additional advantage at larger queries.

|  |  |  | additional acquired samples |  |  |  |  |  |
|---|---|---|---|---|---|---|---|---|
| Task | Init | $a$ | 0 | 200 | 400 | 600 | 800 | 1000 |
| Scitail | BERT | R | 62.57$_{\pm 13.89}$ | 79.32$_{\pm 6.85}$ | 80.56$_{\pm 3.72}$ | 82.77$_{\pm 2.62}$ | 84.14$_{\pm 2.08}$ | 84.09$_{\pm 2.26}$ |
|  |  | E | 62.57$_{\pm 13.89}$ | 78.85$_{\pm 2.42}$ | 77.87$_{\pm 4.30}$ | 81.17$_{\pm 3.37}$ | 80.81$_{\pm 5.63}$ | 81.50$_{\pm 1.54}$ |
|  |  | B | 62.57$_{\pm 13.89}$ | 79.82$_{\pm 3.32}$ | 81.90$_{\pm 2.14}$ | 83.17$_{\pm 0.47}$ | 83.24$_{\pm 2.32}$ | 85.80$_{\pm 1.03}$ |
|  |  | A | 62.57$_{\pm 13.89}$ | 79.39$_{\pm 4.10}$ | 81.86$_{\pm 2.65}$ | 84.23$_{\pm 3.24}$ | 83.12$_{\pm 1.79}$ | 84.82$_{\pm 2.14}$ |
|  |  | C | 62.57$_{\pm 13.89}$ | 79.61$_{\pm 5.89}$ | 80.13$_{\pm 6.37}$ | 83.66$_{\pm 1.68}$ | 84.68$_{\pm 1.17}$ | 85.68$_{\pm 2.89}$ |
|  | meta | R | **76.08**$_{\pm 8.88}$ | 84.24$_{\pm 2.27}$ | 84.51$_{\pm 1.61}$ | 86.21$_{\pm 2.97}$ | 85.93$_{\pm 1.84}$ | 85.90$_{\pm 1.95}$ |
|  |  | E | **76.08**$_{\pm 8.88}$ | 84.80$_{\pm 1.04}$ | 85.68$_{\pm 0.97}$ | **86.69**$_{\pm 2.54}$ | 86.42$_{\pm 2.12}$ | 87.21$_{\pm 2.33}$ |
|  |  | B | **76.08**$_{\pm 8.88}$ | 83.85$_{\pm 1.35}$ | **85.76**$_{\pm 2.15}$ | 86.13$_{\pm 2.44}$ | **87.13**$_{\pm 1.26}$ | 87.14$_{\pm 1.49}$ |
|  |  | A | **76.08**$_{\pm 8.88}$ | 84.99$_{\pm 1.22}$ | 85.56$_{\pm 1.51}$ | 85.79$_{\pm 3.39}$ | 86.74$_{\pm 1.35}$ | 85.90$_{\pm 1.19}$ |
|  |  | C | **76.08**$_{\pm 8.88}$ | **85.08**$_{\pm 1.84}$ | 85.50$_{\pm 3.13}$ | 85.99$_{\pm 4.49}$ | 86.07$_{\pm 8.92}$ | **87.38**$_{\pm 2.23}$ |
| avg. diff. |  |  | +13.51 | +5.19 | +4.94 | +3.16 | +3.26 | +2.33 |
| Amazon | BERT | R | 53.82$_{\pm 6.38}$ | 71.94$_{\pm 18.18}$ | 78.88$_{\pm 5.23}$ | 84.33$_{\pm 3.63}$ | 86.89$_{\pm 1.34}$ | 87.85$_{\pm 4.27}$ |
|  |  | E | 53.82$_{\pm 6.38}$ | 79.28$_{\pm 5.38}$ | 87.87$_{\pm 4.86}$ | 89.02$_{\pm 2.25}$ | 87.90$_{\pm 4.03}$ | 89.91$_{\pm 2.31}$ |
|  |  | B | 53.82$_{\pm 6.38}$ | 80.16$_{\pm 8.95}$ | 88.91$_{\pm 2.68}$ | 88.78$_{\pm 1.67}$ | 89.21$_{\pm 0.62}$ | 90.08$_{\pm 3.28}$ |
|  |  | A | 53.82$_{\pm 6.38}$ | 64.06$_{\pm 5.89}$ | 83.68$_{\pm 3.33}$ | 83.39$_{\pm 9.64}$ | 85.59$_{\pm 5.74}$ | 84.52$_{\pm 6.46}$ |
|  |  | C | 53.82$_{\pm 6.38}$ | 78.35$_{\pm 8.31}$ | 84.57$_{\pm 1.68}$ | 88.90$_{\pm 0.66}$ | 90.08$_{\pm 1.49}$ | 89.86$_{\pm 2.11}$ |
|  | meta | R | **72.35**$_{\pm 15.27}$ | 85.66$_{\pm 6.67}$ | 87.69$_{\pm 5.33}$ | 88.83$_{\pm 1.30}$ | 89.15$_{\pm 1.41}$ | 89.41$_{\pm 2.82}$ |
|  |  | E | **72.35**$_{\pm 15.27}$ | 79.86$_{\pm 31.49}$ | 89.26$_{\pm 1.43}$ | **89.95**$_{\pm 1.73}$ | 90.16$_{\pm 0.62}$ | 90.16$_{\pm 1.25}$ |
|  |  | B | **72.35**$_{\pm 15.27}$ | 86.76$_{\pm 3.70}$ | **89.04**$_{\pm 1.89}$ | 89.64$_{\pm 0.86}$ | **90.40**$_{\pm 2.44}$ | 90.57$_{\pm 0.56}$ |
|  |  | A | **72.35**$_{\pm 15.27}$ | 83.01$_{\pm 6.09}$ | 85.26$_{\pm 5.37}$ | 86.55$_{\pm 3.92}$ | 89.33$_{\pm 1.25}$ | 88.84$_{\pm 2.52}$ |
|  |  | C | **72.35**$_{\pm 15.27}$ | **86.88**$_{\pm 2.23}$ | 88.69$_{\pm 2.32}$ | 89.75$_{\pm 1.80}$ | 89.60$_{\pm 3.56}$ | **90.68**$_{\pm 1.37}$ |
| avg. diff. |  |  | +18.54 | +9.68 | +3.21 | +2.06 | +1.79 | +1.49 |
| Yelp | BERT | R | 23.28$_{\pm 6.09}$ | 41.53$_{\pm 3.02}$ | 47.42$_{\pm 3.08}$ | 50.37$_{\pm 0.91}$ | 51.95$_{\pm 0.61}$ | 52.32$_{\pm 2.01}$ |
|  |  | E | 23.28$_{\pm 6.09}$ | 44.25$_{\pm 5.16}$ | 43.78$_{\pm 4.83}$ | 49.43$_{\pm 1.48}$ | 48.07$_{\pm 4.19}$ | 51.43$_{\pm 1.06}$ |
|  |  | B | 23.28$_{\pm 6.09}$ | 40.93$_{\pm 9.06}$ | 45.02$_{\pm 3.00}$ | 49.97$_{\pm 0.30}$ | **51.93**$_{\pm 1.67}$ | **52.49**$_{\pm 1.08}$ |
|  |  | A | 23.28$_{\pm 6.09}$ | 41.56$_{\pm 7.90}$ | 49.30$_{\pm 2.91}$ | 51.12$_{\pm 2.29}$ | 51.33$_{\pm 0.70}$ | 52.38$_{\pm 2.41}$ |
|  |  | C | 23.28$_{\pm 6.09}$ | 42.52$_{\pm 4.05}$ | 44.48$_{\pm 11.30}$ | 50.01$_{\pm 2.55}$ | 50.81$_{\pm 1.88}$ | 51.93$_{\pm 0.82}$ |
|  | meta | R | **33.82**$_{\pm 5.50}$ | 47.19$_{\pm 3.24}$ | 49.89$_{\pm 0.99}$ | 50.00$_{\pm 1.72}$ | 51.85$_{\pm 1.30}$ | 51.82$_{\pm 1.28}$ |
|  |  | E | **33.82**$_{\pm 5.50}$ | 43.15$_{\pm 8.19}$ | 48.49$_{\pm 3.10}$ | 49.03$_{\pm 1.91}$ | 50.66$_{\pm 3.02}$ | 50.99$_{\pm 2.86}$ |
|  |  | B | **33.82**$_{\pm 5.50}$ | 46.40$_{\pm 4.84}$ | **50.03**$_{\pm 0.53}$ | **51.68**$_{\pm 1.99}$ | 51.87$_{\pm 1.98}$ | 51.11$_{\pm 1.44}$ |
|  |  | A | **33.82**$_{\pm 5.50}$ | **47.16**$_{\pm 2.74}$ | 47.08$_{\pm 3.71}$ | 50.13$_{\pm 1.04}$ | 50.53$_{\pm 1.30}$ | 51.21$_{\pm 2.42}$ |
|  |  | C | **33.82**$_{\pm 5.50}$ | 45.02$_{\pm 3.01}$ | 47.83$_{\pm 3.44}$ | 49.23$_{\pm 3.62}$ | 51.49$_{\pm 1.54}$ | 52.17$_{\pm 1.24}$ |
| avg. diff. |  |  | +10.54 | +3.63 | +2.67 | -0.16 | +0.46 | -0.65 |
| Emotion | BERT | R | 32.38$_{\pm 5.36}$ | 62.84$_{\pm 3.36}$ | 77.12$_{\pm 2.03}$ | 81.27$_{\pm 2.90}$ | 84.92$_{\pm 1.74}$ | 86.59$_{\pm 1.04}$ |
|  |  | E | 32.38$_{\pm 5.36}$ | 63.21$_{\pm 2.81}$ | 77.21$_{\pm 3.76}$ | 82.71$_{\pm 2.17}$ | 84.36$_{\pm 5.03}$ | 88.57$_{\pm 1.34}$ |
|  |  | B | 32.38$_{\pm 5.36}$ | 66.72$_{\pm 7.64}$ | 74.05$_{\pm 4.92}$ | 84.94$_{\pm 2.51}$ | 87.41$_{\pm 0.43}$ | 88.80$_{\pm 0.71}$ |
|  |  | A | 32.38$_{\pm 5.36}$ | 63.21$_{\pm 11.16}$ | 77.56$_{\pm 1.42}$ | 81.48$_{\pm 1.86}$ | 85.34$_{\pm 0.90}$ | 86.18$_{\pm 0.98}$ |
|  |  | C | 32.38$_{\pm 5.36}$ | 58.67$_{\pm 12.06}$ | 67.14$_{\pm 18.84}$ | 82.77$_{\pm 4.38}$ | 87.45$_{\pm 2.49}$ | 89.13$_{\pm 1.59}$ |
|  | meta | R | **44.65**$_{\pm 15.80}$ | **70.06**$_{\pm 3.88}$ | 78.43$_{\pm 0.74}$ | 82.12$_{\pm 0.20}$ | 83.93$_{\pm 1.43}$ | 85.71$_{\pm 1.17}$ |
|  |  | E | **44.65**$_{\pm 15.80}$ | 60.10$_{\pm 17.44}$ | 76.74$_{\pm 2.20}$ | 80.35$_{\pm 5.07}$ | 84.84$_{\pm 1.55}$ | 87.11$_{\pm 3.12}$ |
|  |  | B | **44.65**$_{\pm 15.80}$ | 69.88$_{\pm 5.47}$ | **81.11**$_{\pm 3.59}$ | **85.17**$_{\pm 2.84}$ | **87.63**$_{\pm 1.36}$ | **89.48**$_{\pm 0.58}$ |
|  |  | A | **44.65**$_{\pm 15.80}$ | 67.49$_{\pm 6.55}$ | 76.36$_{\pm 3.84}$ | 81.08$_{\pm 0.97}$ | 83.82$_{\pm 0.96}$ | 85.39$_{\pm 1.04}$ |
|  |  | C | **44.65**$_{\pm 15.80}$ | 56.20$_{\pm 24.42}$ | 72.97$_{\pm 10.00}$ | 81.53$_{\pm 3.59}$ | 87.37$_{\pm 1.19}$ | 89.12$_{\pm 0.77}$ |
| avg. diff. |  |  | +12.27 | +1.82 | +2.51 | -0.58 | -0.38 | -0.49 |

Table 4: AL performance between the base and meta-learned initialization $\theta^*_{\mathcal{T}}$ across different tasks, seed size $\boldsymbol{s = 20}$ and acquisition size of $q = 50$. The table shows the accuracy at $0, 200, 400, 600, 800$ and $1000$ additionally acquired samples. Where $R = random, E = entropy, B = BADGE, A = ALPS$ and $C = CAL$.

| | | | additional acquired samples | | | | | |
|---|---|---|---|---|---|---|---|---|
| Task | Init | $a$ | 0 | 200 | 400 | 600 | 800 | 1000 |
| Scitail | BERT | R | 75.86$_{\pm4.15}$ | 79.11$_{\pm3.53}$ | 81.86$_{\pm1.24}$ | 81.91$_{\pm0.56}$ | 83.86$_{\pm0.51}$ | 83.79$_{\pm0.71}$ |
| | | E | 75.86$_{\pm4.15}$ | 79.83$_{\pm2.50}$ | 81.10$_{\pm2.79}$ | 82.13$_{\pm1.26}$ | 82.31$_{\pm0.89}$ | 81.95$_{\pm1.97}$ |
| | | B | 75.86$_{\pm4.15}$ | 80.30$_{\pm1.36}$ | 80.22$_{\pm2.43}$ | 84.40$_{\pm1.44}$ | 84.33$_{\pm0.45}$ | 84.66$_{\pm0.48}$ |
| | | A | 75.86$_{\pm4.15}$ | 81.45$_{\pm1.97}$ | 82.38$_{\pm1.35}$ | 82.98$_{\pm1.22}$ | 83.02$_{\pm2.35}$ | 85.38$_{\pm0.90}$ |
| | | C | 75.86$_{\pm4.15}$ | 78.17$_{\pm5.31}$ | 81.19$_{\pm1.07}$ | 84.02$_{\pm0.64}$ | 83.75$_{\pm0.87}$ | 85.21$_{\pm0.86}$ |
| | meta | R | **78.27**$_{\pm7.28}$ | 84.52$_{\pm1.53}$ | 85.47$_{\pm0.57}$ | 86.26$_{\pm0.34}$ | 87.14$_{\pm1.10}$ | 87.06$_{\pm0.79}$ |
| | | E | **78.27**$_{\pm7.28}$ | 80.37$_{\pm8.07}$ | 85.96$_{\pm0.30}$ | 86.17$_{\pm0.36}$ | 85.86$_{\pm0.50}$ | 86.87$_{\pm1.04}$ |
| | | B | **78.27**$_{\pm7.28}$ | 84.45$_{\pm1.47}$ | 86.40$_{\pm0.69}$ | **86.98**$_{\pm1.03}$ | 86.96$_{\pm0.75}$ | 86.96$_{\pm0.93}$ |
| | | A | **78.27**$_{\pm7.28}$ | **85.31**$_{\pm0.74}$ | **87.19**$_{\pm1.35}$ | 86.74$_{\pm0.40}$ | 86.91$_{\pm1.00}$ | 87.73$_{\pm1.46}$ |
| | | C | **78.27**$_{\pm7.28}$ | 74.57$_{\pm18.85}$ | 83.81$_{\pm3.42}$ | 86.83$_{\pm0.52}$ | **87.79**$_{\pm0.60}$ | **87.95**$_{\pm1.23}$ |
| avg. diff. | | | +2.41 | +2.07 | +4.42 | +3.51 | +3.48 | +3.12 |
| Amazon | BERT | R | 52.27$_{\pm0.88}$ | 76.35$_{\pm6.03}$ | 85.69$_{\pm1.73}$ | 88.12$_{\pm0.51}$ | 88.81$_{\pm1.13}$ | 89.55$_{\pm0.37}$ |
| | | E | 52.27$_{\pm0.88}$ | 85.86$_{\pm3.79}$ | 87.21$_{\pm1.67}$ | 89.01$_{\pm0.68}$ | 89.35$_{\pm0.91}$ | 89.82$_{\pm1.02}$ |
| | | B | 52.27$_{\pm0.88}$ | 82.89$_{\pm5.58}$ | 86.72$_{\pm1.58}$ | 88.25$_{\pm1.06}$ | 88.67$_{\pm1.66}$ | 90.12$_{\pm0.72}$ |
| | | A | 52.27$_{\pm0.88}$ | 65.38$_{\pm8.02}$ | 82.49$_{\pm6.23}$ | 85.83$_{\pm2.17}$ | 86.65$_{\pm1.80}$ | 87.11$_{\pm1.68}$ |
| | | C | 52.27$_{\pm0.88}$ | 81.78$_{\pm4.87}$ | 87.18$_{\pm2.13}$ | 89.21$_{\pm0.94}$ | 90.21$_{\pm0.38}$ | 89.65$_{\pm0.67}$ |
| | meta | R | **76.56**$_{\pm8.50}$ | 83.93$_{\pm6.39}$ | 88.14$_{\pm1.23}$ | 86.51$_{\pm3.32}$ | 88.66$_{\pm1.44}$ | 88.59$_{\pm0.87}$ |
| | | E | **76.56**$_{\pm8.50}$ | 87.86$_{\pm1.56}$ | 88.57$_{\pm0.58}$ | 89.84$_{\pm1.20}$ | 90.14$_{\pm1.03}$ | 89.69$_{\pm1.06}$ |
| | | B | **76.56**$_{\pm8.50}$ | 87.08$_{\pm2.00}$ | **88.96**$_{\pm0.85}$ | **89.86**$_{\pm0.84}$ | 90.59$_{\pm0.63}$ | 90.38$_{\pm0.96}$ |
| | | A | **76.56**$_{\pm8.50}$ | 84.22$_{\pm1.85}$ | 86.19$_{\pm1.94}$ | 87.60$_{\pm1.93}$ | 88.22$_{\pm1.09}$ | 89.19$_{\pm0.64}$ |
| | | C | **76.56**$_{\pm8.50}$ | **89.04**$_{\pm1.04}$ | 87.84$_{\pm1.68}$ | 88.90$_{\pm2.00}$ | **90.59**$_{\pm0.43}$ | **91.31**$_{\pm0.24}$ |
| avg. diff. | | | +24.30 | +7.97 | +2.08 | +0.46 | +0.90 | +0.58 |
| Yelp | BERT | R | 24.62$_{\pm1.11}$ | 43.07$_{\pm3.32}$ | 49.67$_{\pm1.18}$ | 48.31$_{\pm3.52}$ | 51.72$_{\pm0.63}$ | 51.13$_{\pm1.74}$ |
| | | E | 24.62$_{\pm1.11}$ | 40.47$_{\pm1.06}$ | 47.55$_{\pm1.83}$ | 47.81$_{\pm3.64}$ | 50.84$_{\pm1.22}$ | 49.79$_{\pm3.01}$ |
| | | B | 24.62$_{\pm1.11}$ | 42.72$_{\pm2.52}$ | 49.16$_{\pm0.75}$ | 50.81$_{\pm0.57}$ | 51.91$_{\pm1.04}$ | 52.01$_{\pm1.39}$ |
| | | A | 24.62$_{\pm1.11}$ | 43.60$_{\pm2.02}$ | 49.71$_{\pm0.17}$ | 50.65$_{\pm1.18}$ | 52.44$_{\pm0.42}$ | 51.96$_{\pm1.30}$ |
| | | C | 24.62$_{\pm1.11}$ | 40.95$_{\pm3.63}$ | 48.85$_{\pm1.64}$ | 49.38$_{\pm3.55}$ | 51.49$_{\pm1.11}$ | 51.88$_{\pm1.16}$ |
| | meta | R | **37.81**$_{\pm1.99}$ | 47.39$_{\pm2.27}$ | **50.23**$_{\pm0.87}$ | **50.99**$_{\pm0.74}$ | 51.96$_{\pm0.73}$ | 51.35$_{\pm0.60}$ |
| | | E | **37.81**$_{\pm1.99}$ | 42.79$_{\pm3.09}$ | 48.01$_{\pm2.48}$ | 49.06$_{\pm1.20}$ | 51.77$_{\pm0.97}$ | 51.17$_{\pm1.46}$ |
| | | B | **37.81**$_{\pm1.99}$ | 45.96$_{\pm1.22}$ | 48.86$_{\pm2.17}$ | 50.87$_{\pm1.40}$ | **52.06**$_{\pm0.76}$ | **52.63**$_{\pm0.95}$ |
| | | A | **37.81**$_{\pm1.99}$ | **48.21**$_{\pm0.76}$ | 49.72$_{\pm1.64}$ | 50.31$_{\pm0.98}$ | 51.66$_{\pm0.89}$ | 51.61$_{\pm0.35}$ |
| | | C | **37.81**$_{\pm1.99}$ | 45.87$_{\pm1.59}$ | 48.80$_{\pm2.76}$ | 49.33$_{\pm1.76}$ | 51.79$_{\pm0.98}$ | 51.98$_{\pm0.31}$ |
| avg. diff. | | | +13.18 | +3.88 | +0.13 | +0.72 | +0.17 | +0.40 |
| Emotion | BERT | R | 41.11$_{\pm3.23}$ | 64.36$_{\pm1.36}$ | 76.62$_{\pm1.60}$ | 83.03$_{\pm0.67}$ | 85.90$_{\pm0.50}$ | 87.14$_{\pm0.74}$ |
| | | E | 41.11$_{\pm3.23}$ | 68.55$_{\pm1.68}$ | 76.80$_{\pm3.07}$ | 83.65$_{\pm1.25}$ | 85.69$_{\pm1.07}$ | 87.54$_{\pm1.66}$ |
| | | B | 41.11$_{\pm3.23}$ | 63.88$_{\pm6.51}$ | 77.24$_{\pm2.14}$ | 84.03$_{\pm1.44}$ | 88.19$_{\pm0.43}$ | 89.12$_{\pm0.37}$ |
| | | A | 41.11$_{\pm3.23}$ | 63.13$_{\pm4.80}$ | 78.87$_{\pm1.92}$ | 83.15$_{\pm1.70}$ | 84.91$_{\pm0.83}$ | 86.49$_{\pm0.60}$ |
| | | C | 41.11$_{\pm3.23}$ | 66.26$_{\pm2.17}$ | 79.21$_{\pm1.33}$ | 84.76$_{\pm1.59}$ | 87.55$_{\pm0.63}$ | 88.97$_{\pm0.36}$ |
| | meta | R | **52.28**$_{\pm0.99}$ | 69.55$_{\pm1.62}$ | 78.07$_{\pm1.29}$ | 82.87$_{\pm1.34}$ | 85.10$_{\pm0.52}$ | 86.33$_{\pm0.86}$ |
| | | E | **52.28**$_{\pm0.99}$ | 68.64$_{\pm2.57}$ | 74.42$_{\pm4.15}$ | 80.27$_{\pm4.04}$ | 86.44$_{\pm1.40}$ | 89.05$_{\pm0.81}$ |
| | | B | **52.28**$_{\pm0.99}$ | **71.49**$_{\pm2.13}$ | 80.08$_{\pm1.71}$ | **85.07**$_{\pm1.27}$ | 87.45$_{\pm0.23}$ | 89.27$_{\pm0.44}$ |
| | | A | **52.28**$_{\pm0.99}$ | 71.42$_{\pm1.99}$ | 78.30$_{\pm0.88}$ | 82.56$_{\pm1.11}$ | 84.48$_{\pm0.48}$ | 86.57$_{\pm0.91}$ |
| | | C | **52.28**$_{\pm0.99}$ | 70.47$_{\pm3.00}$ | **81.63**$_{\pm0.79}$ | 84.68$_{\pm1.36}$ | **88.37**$_{\pm0.62}$ | **89.46**$_{\pm1.05}$ |
| avg. diff. | | | +11.17 | +5.08 | +0.75 | -0.63 | -0.08 | -0.29 |

Table 5: AL performance between the base and meta-learned initialization $\theta^*_{\mathcal{T}}$ across different tasks, seed size $\boldsymbol{s = 50}$ and acquisition size of $q = 50$. The table shows the accuracy at $0, 200, 400, 600, 800$ and $1000$ additionally acquired samples. Where $R = random, E = entropy, B = BADGE, A = ALPS$ and $C = CAL$.

|  |  |  | additional acquired samples |  |  |  |  |  |
|---|---|---|---|---|---|---|---|---|
| Task | Init | $a$ | 0 | 200 | 400 | 600 | 800 | 1000 |
| Scitail | BERT | R | 80.48$_{\pm 0.94}$ | 82.32$_{\pm 0.99}$ | 83.21$_{\pm 0.25}$ | 84.35$_{\pm 0.83}$ | 83.48$_{\pm 3.36}$ | 83.82$_{\pm 1.65}$ |
|  |  | E | 80.48$_{\pm 0.94}$ | 80.41$_{\pm 0.44}$ | 80.55$_{\pm 1.76}$ | 82.53$_{\pm 0.99}$ | 82.26$_{\pm 0.98}$ | 83.29$_{\pm 0.72}$ |
|  |  | B | 80.48$_{\pm 0.94}$ | 82.41$_{\pm 0.63}$ | 83.15$_{\pm 0.41}$ | 84.39$_{\pm 1.19}$ | 83.95$_{\pm 1.61}$ | 84.57$_{\pm 0.95}$ |
|  |  | A | 80.48$_{\pm 0.94}$ | 80.86$_{\pm 1.19}$ | 82.40$_{\pm 0.83}$ | 83.30$_{\pm 1.43}$ | 83.88$_{\pm 0.29}$ | 84.37$_{\pm 0.97}$ |
|  |  | C | 80.48$_{\pm 0.94}$ | 79.42$_{\pm 1.86}$ | 81.64$_{\pm 1.88}$ | 83.19$_{\pm 1.18}$ | 83.73$_{\pm 1.26}$ | 84.89$_{\pm 1.02}$ |
|  | meta | R | **83.71**$_{\pm 0.94}$ | 84.89$_{\pm 1.09}$ | 86.59$_{\pm 1.21}$ | 86.75$_{\pm 0.44}$ | **87.53**$_{\pm 0.83}$ | 86.84$_{\pm 0.77}$ |
|  |  | E | **83.71**$_{\pm 0.94}$ | 84.26$_{\pm 1.31}$ | 85.78$_{\pm 1.32}$ | 85.95$_{\pm 0.99}$ | 86.36$_{\pm 1.13}$ | 86.58$_{\pm 0.82}$ |
|  |  | B | **83.71**$_{\pm 0.94}$ | 84.86$_{\pm 1.99}$ | 86.66$_{\pm 0.85}$ | **86.92**$_{\pm 0.54}$ | 85.87$_{\pm 1.72}$ | 88.05$_{\pm 1.26}$ |
|  |  | A | **83.71**$_{\pm 0.94}$ | **86.46**$_{\pm 1.46}$ | **87.00**$_{\pm 0.61}$ | 86.53$_{\pm 0.90}$ | 87.41$_{\pm 0.29}$ | 87.72$_{\pm 0.74}$ |
|  |  | C | **83.71**$_{\pm 0.94}$ | 81.94$_{\pm 3.24}$ | 86.25$_{\pm 0.80}$ | 83.65$_{\pm 4.91}$ | 87.50$_{\pm 1.07}$ | **88.48**$_{\pm 0.98}$ |
| avg. diff. |  |  | +3.22 | +3.40 | +4.26 | +2.41 | +3.47 | +3.35 |
| Amazon | BERT | R | 64.60$_{\pm 12.01}$ | 81.20$_{\pm 5.13}$ | 86.19$_{\pm 2.28}$ | 85.56$_{\pm 2.49}$ | 87.69$_{\pm 1.57}$ | 88.82$_{\pm 0.89}$ |
|  |  | E | 64.60$_{\pm 12.01}$ | 85.68$_{\pm 1.16}$ | 85.82$_{\pm 2.14}$ | 88.26$_{\pm 1.29}$ | 91.00$_{\pm 0.45}$ | 89.76$_{\pm 0.61}$ |
|  |  | B | 64.60$_{\pm 12.01}$ | 84.46$_{\pm 2.05}$ | 86.63$_{\pm 2.31}$ | 88.44$_{\pm 2.84}$ | 90.15$_{\pm 0.25}$ | 89.32$_{\pm 0.84}$ |
|  |  | A | 64.60$_{\pm 12.01}$ | 73.95$_{\pm 10.25}$ | 83.73$_{\pm 4.22}$ | 86.55$_{\pm 2.30}$ | 87.46$_{\pm 1.57}$ | 88.37$_{\pm 1.89}$ |
|  |  | C | 64.60$_{\pm 12.01}$ | 80.69$_{\pm 4.46}$ | 86.01$_{\pm 2.70}$ | 89.39$_{\pm 0.84}$ | 90.15$_{\pm 0.45}$ | 90.44$_{\pm 0.80}$ |
|  | meta | R | **74.31**$_{\pm 8.23}$ | 88.09$_{\pm 1.75}$ | 87.36$_{\pm 1.34}$ | 89.13$_{\pm 0.58}$ | 89.66$_{\pm 0.29}$ | 89.76$_{\pm 0.80}$ |
|  |  | E | **74.31**$_{\pm 8.23}$ | 87.03$_{\pm 1.19}$ | **89.81**$_{\pm 0.94}$ | 88.81$_{\pm 1.34}$ | 90.23$_{\pm 0.72}$ | **90.84**$_{\pm 0.69}$ |
|  |  | B | **74.31**$_{\pm 8.23}$ | **88.89**$_{\pm 1.23}$ | 88.98$_{\pm 0.92}$ | **90.11**$_{\pm 0.37}$ | 90.13$_{\pm 0.85}$ | 90.04$_{\pm 0.62}$ |
|  |  | A | **74.31**$_{\pm 8.23}$ | 85.77$_{\pm 2.37}$ | 86.18$_{\pm 1.89}$ | 87.58$_{\pm 1.79}$ | 89.91$_{\pm 0.79}$ | 88.80$_{\pm 1.45}$ |
|  |  | C | **74.31**$_{\pm 8.23}$ | 88.15$_{\pm 0.99}$ | 89.13$_{\pm 1.46}$ | 89.85$_{\pm 0.74}$ | **90.56**$_{\pm 1.10}$ | 90.40$_{\pm 0.53}$ |
| avg. diff. |  |  | +9.71 | +6.39 | +2.61 | +1.46 | +0.86 | +0.63 |
| Yelp | BERT | R | 31.39$_{\pm 3.46}$ | 45.60$_{\pm 3.07}$ | 48.00$_{\pm 1.23}$ | **51.88**$_{\pm 0.87}$ | **52.04**$_{\pm 0.80}$ | 52.40$_{\pm 0.86}$ |
|  |  | E | 31.39$_{\pm 3.46}$ | 42.69$_{\pm 1.80}$ | 49.05$_{\pm 0.82}$ | 51.05$_{\pm 0.65}$ | 50.64$_{\pm 1.64}$ | 51.59$_{\pm 0.52}$ |
|  |  | B | 31.39$_{\pm 3.46}$ | 48.01$_{\pm 0.78}$ | 49.35$_{\pm 2.22}$ | 50.78$_{\pm 0.49}$ | 51.74$_{\pm 0.68}$ | 52.54$_{\pm 1.13}$ |
|  |  | A | 31.39$_{\pm 3.46}$ | 47.36$_{\pm 2.25}$ | 50.07$_{\pm 0.54}$ | 51.40$_{\pm 0.50}$ | 51.59$_{\pm 0.50}$ | 52.68$_{\pm 1.09}$ |
|  |  | C | 31.39$_{\pm 3.46}$ | 42.92$_{\pm 4.38}$ | 48.91$_{\pm 0.36}$ | 50.47$_{\pm 1.43}$ | 50.31$_{\pm 3.43}$ | 51.14$_{\pm 1.71}$ |
|  | meta | R | **39.43**$_{\pm 3.78}$ | 47.66$_{\pm 1.29}$ | 49.24$_{\pm 1.47}$ | 50.74$_{\pm 0.78}$ | 51.05$_{\pm 1.12}$ | **52.58**$_{\pm 1.00}$ |
|  |  | E | **39.43**$_{\pm 3.78}$ | 46.24$_{\pm 2.13}$ | 47.68$_{\pm 2.90}$ | 51.15$_{\pm 1.90}$ | 50.25$_{\pm 2.77}$ | 51.33$_{\pm 1.58}$ |
|  |  | B | **39.43**$_{\pm 3.78}$ | 45.26$_{\pm 4.15}$ | **50.82**$_{\pm 0.69}$ | 51.32$_{\pm 0.75}$ | 51.94$_{\pm 1.24}$ | 52.54$_{\pm 1.05}$ |
|  |  | A | **39.43**$_{\pm 3.78}$ | **47.44**$_{\pm 1.63}$ | 49.63$_{\pm 0.88}$ | 50.30$_{\pm 1.06}$ | 50.62$_{\pm 0.70}$ | 51.60$_{\pm 0.64}$ |
|  |  | C | **39.43**$_{\pm 3.78}$ | 44.09$_{\pm 3.50}$ | 49.13$_{\pm 1.75}$ | 50.36$_{\pm 1.17}$ | 51.98$_{\pm 0.61}$ | 51.96$_{\pm 0.55}$ |
| avg. diff. |  |  | +8.04 | +0.82 | +0.22 | -0.34 | -0.10 | -0.07 |
| Emotion | BERT | R | 47.53$_{\pm 5.22}$ | 69.46$_{\pm 1.89}$ | 78.78$_{\pm 1.03}$ | 82.83$_{\pm 1.51}$ | 85.55$_{\pm 0.70}$ | 86.73$_{\pm 0.94}$ |
|  |  | E | 47.53$_{\pm 5.22}$ | 69.87$_{\pm 2.96}$ | 75.81$_{\pm 5.00}$ | 84.08$_{\pm 0.80}$ | 86.70$_{\pm 1.23}$ | 89.30$_{\pm 0.28}$ |
|  |  | B | 47.53$_{\pm 5.22}$ | 68.37$_{\pm 0.77}$ | 80.98$_{\pm 1.50}$ | 85.67$_{\pm 1.09}$ | 88.62$_{\pm 1.18}$ | 89.30$_{\pm 0.47}$ |
|  |  | A | 47.53$_{\pm 5.22}$ | 72.22$_{\pm 3.09}$ | 78.73$_{\pm 2.20}$ | 84.27$_{\pm 0.55}$ | 85.32$_{\pm 0.84}$ | 86.84$_{\pm 0.36}$ |
|  |  | C | 47.53$_{\pm 5.22}$ | 70.05$_{\pm 2.92}$ | 80.79$_{\pm 1.17}$ | 85.17$_{\pm 1.07}$ | 87.58$_{\pm 0.66}$ | **89.88**$_{\pm 0.94}$ |
|  | meta | R | **56.99**$_{\pm 3.06}$ | 71.68$_{\pm 0.66}$ | 80.01$_{\pm 1.10}$ | 82.68$_{\pm 0.57}$ | 84.69$_{\pm 0.79}$ | 86.51$_{\pm 0.61}$ |
|  |  | E | **56.99**$_{\pm 3.06}$ | 67.59$_{\pm 6.82}$ | 80.82$_{\pm 1.48}$ | 82.84$_{\pm 1.39}$ | 85.25$_{\pm 3.50}$ | 88.44$_{\pm 1.50}$ |
|  |  | B | **56.99**$_{\pm 3.06}$ | 72.71$_{\pm 1.68}$ | **82.24**$_{\pm 1.15}$ | 85.69$_{\pm 0.96}$ | 88.08$_{\pm 0.34}$ | 89.58$_{\pm 0.51}$ |
|  |  | A | **56.99**$_{\pm 3.06}$ | **74.37**$_{\pm 1.13}$ | 80.02$_{\pm 1.29}$ | 83.26$_{\pm 1.15}$ | 85.74$_{\pm 0.46}$ | 86.81$_{\pm 0.32}$ |
|  |  | C | **56.99**$_{\pm 3.06}$ | 70.82$_{\pm 5.19}$ | 80.75$_{\pm 2.20}$ | **86.74**$_{\pm 1.16}$ | **88.32**$_{\pm 0.56}$ | 89.70$_{\pm 0.56}$ |
| avg. diff. |  |  | +9.46 | +1.44 | +1.75 | -0.16 | -0.34 | -0.20 |

Table 6: AL performance between the base and meta-learned initialization $\theta^*_{\mathcal{T}}$ across different tasks, seed size $\boldsymbol{s = 100}$ and acquisition size of $q = 50$. The table shows the accuracy at $0, 200, 400, 600, 800$ and $1000$ additionally acquired samples. Where $R = random, E = entropy, B = BADGE, A = ALPS$ and $C = CAL$.