Self-Training for Sample-Efficient Active Learning for Text Classification with Pre-Trained Language Models

Christopher Schröder[1,2,3] and **Gerhard Heyer**[1,3]

[1]Center for Scalable Data Analytics and Artificial Intelligence (ScaDS.AI), Dresden/Leipzig
[2]TUD Dresden University of Technology
[3]Leipzig University

Abstract

Active learning is an iterative labeling process that is used to obtain a small labeled subset, despite the absence of labeled data, thereby enabling to train a model for supervised tasks such as text classification. While active learning has made considerable progress in recent years due to improvements provided by pre-trained language models, there is untapped potential in the often neglected unlabeled portion of the data, although it is available in considerably larger quantities than the usually small set of labeled data. In this work, we investigate how self-training, a semi-supervised approach that uses a model to obtain pseudo-labels for unlabeled data, can be used to improve the efficiency of active learning for text classification. Building on a comprehensive reproduction of four previous self-training approaches, some of which are evaluated for the first time in the context of active learning or natural language processing, we introduce HAST, a new and effective self-training strategy, which is evaluated on four text classification benchmarks. Our results show that it outperforms the reproduced self-training approaches and reaches classification results comparable to previous experiments for three out of four datasets, using as little as 25% of the data. The code is publicly available at https://github.com/chschroeder/self-training-for-sample-efficient-active-learning.

1 Introduction

In supervised machine learning, a lack of labeled data is the main obstacle to real-world applications, since labeled data is usually non-existent, expensive to obtain, and sometimes even requires domain experts for annotations. One solution to create models despite the absence of labels, is *active learning*, where in an iterative process an oracle (usually realized through a human annotator) provides labels for unlabeled instances that have been deemed to be informative by a so-called *query strategy*. These labels are then used to train a model, which in turn is used by the query strategy during the next iteration. In this work, we investigate the combination of self-training and active learning to reduce the required amount of labeled data even further.

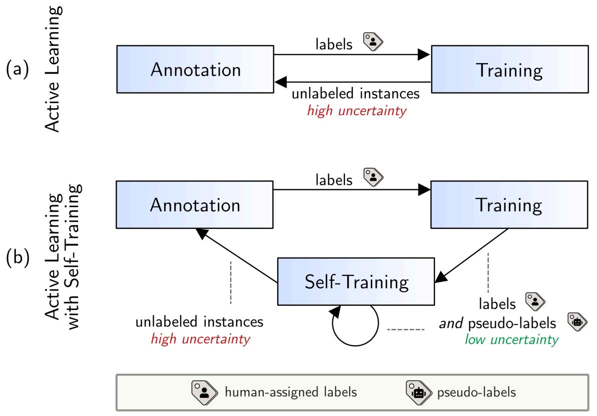

Figure 1: Active learning (a), and active learning with interleaved self-training (b). For active learning, the most uncertain samples are labeled by the human annotator, while for self-training pseudo-labels are obtained from the current model using the most certain samples.

During recent years, transformer-based pre-trained language models (Vaswani et al., 2017; Devlin et al., 2019) have successfully been applied for active-learning-based text classification, thereby considerably raising the state-of-the-art results (e.g., Margatina et al., 2021). The dominant paradigm here is pool-based active learning (Lewis and Gale, 1994) where the query strategy repeatedly selects batches of instances to be labeled next from the *pool*, the entirety of unlabeled data. While language models have successfully been adopted for active learning (e.g., by Ein-Dor et al. (2020), Yuan et al. (2020), and Margatina et al. (2021)), the total labeling effort, i.e. the number of queries and the number of instances per query, has remained similar to setups predating transformers. With regard to the size of queries, there are two prevailing setups: (1) Absolute query sizes (Yang et al.,

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 11987–12004
November 12-16, 2024 ©2024 Association for Computational Linguistics

2009; Sharma et al., 2015; Zhang et al., 2017; Ein-Dor et al., 2020; Yuan et al., 2020; Schröder et al., 2022; Tonneau et al., 2022), where a fixed number of instances are queried during each iteration, and (2) relative query sizes (Lowell et al., 2019; Prabhu et al., 2019; Margatina et al., 2021), in which the number of queried instances is a percentage of the unlabeled pool. We argue that those query sizes of both aforementioned experiment setups are needlessly large. Previous works query up to 1000 instances (Yang et al., 2009) or up to 25% percent of the unlabeled pool (Lowell et al., 2019), where the former is of considerable size and the latter is clearly infeasible in practice as soon as datasets reach average contemporary sizes or annotation costs are high. When using language models that have been trained on billions (Devlin et al., 2019) or even trillions (Touvron et al., 2023) of tokens, **there is no need to label hundreds or even thousands of instances.**

In this work, we introduce a sample-efficient active learning approach which incorporates self-training to reduce the amount of training data required for our key task of text classification. Our contributions are as follows: (1) We propose a simple yet highly effective self-training approach that complements high-quality active learning labels with high-quantity *pseudo-labels*. (2) We reproduce four existing self-training approaches, enabling a fair comparison among them despite strongly diverging settings and hyperparameter choices in the original works. (3) In extensive experiments, we compare the new approach to the four reproduced methods on four text classification benchmarks using two query strategies and a baseline. (4) Finally, we discuss possible implications for active learning that result from the observed effectiveness, as well as the trade-offs between favoring small versus large language models.

The new approach complements active learning with pseudo-labels obtained from the current model. Using as few as 130 instances, we achieve scores competitive with regard to the state of the art on three out of four datasets.

2 Related Work

In this work, we investigate the intersection of self-training and active learning for text classification.

Self-Training The idea of self-training (Scudder, 1965; Yarowsky, 1995) is to leverage unlabeled data in supervised tasks, by obtaining algorithmically-derived pseudo-labels that are subsequently used for training a model. In natural language processing (NLP), self-training is an established and well-studied semi-supervised approach (Clark et al., 2003; Mihalcea, 2004; Tomanek and Hahn, 2009; Ye et al., 2020) that provides additional data by generating soft or hard labels from unlabeled data. Similar to active learning, the selection of suitable unlabeled instances is essential. However, unlike active learning, there is no human in the loop, therefore, self-training aims to obtain pseudo-labels that are likely to be correct. Pseudo-labels, however, are not guaranteed to be correct, a central issue is pseudo-label regularization, which prevents overfitting on incorrect pseudo-labels.

The recently dominant class of pre-trained transformer models (Vaswani et al., 2017; Devlin et al., 2019) are well-known for their improved effectiveness which derives, among other things, from a high sample efficiency of the contextualized representation, and therefore each additional pseudo-labeled instance can be highly valuable for self-training a pre-trained model. Consequently, it is unsurpising that self-training has been investigated in NLP with recent model architectures (Meng et al., 2020; Mukherjee and Awadallah, 2020; Vu et al., 2021; Gera et al., 2022; Chen et al., 2022; Sosea and Caragea, 2022), however, it is still underresearched regarding active learning (Yu et al., 2022; Xu et al., 2023), where the additional labeled data could help to alleviate the data scarcity.

Active Learning and Self-Training Despite the recent performance gains achieved by transformer models, active learning still uses a considerable amount of data. Recent work in transformer-based active learning for NLP, however, focused on query strategies (Ein-Dor et al., 2020; Margatina et al., 2021; Zhang and Plank, 2021; Wertz et al., 2023; Zeng and Zubiaga, 2023), which often raised the state-of-the-art results given the same labeling budget, but mostly disregarded translating these improvements into reduced annotation efforts.

Despite the comparably slow adoption of self-training, a few recent works already started to investigate the use of unlabeled data in order to improve data efficiency (Siméoni et al., 2020; Gonsior et al., 2020; Gilhuber et al., 2022; Tsvigun et al., 2022). In the context of active learning for text classification, both Yu et al. (2022) and (Xu et al., 2023) use pre-trained language models for active learning for text classification, thereby outperforming regular

	Pseudo-Label Selection			Self-Training							Setting	
	Subs.	Unc.	Div.	Cls. Bal.	Weight.	Regularization					Data	Domain
Approach						D	P	E	T	N		
UST	✔	✔	(✔)	✔	✔	✔	✖	✖	✖	✖	Text	Few-Shot
AcTune	✖	✔	✔	✖	✔	✖	✔	✖	✔	✖	Text	Active Learning
VERIPS	✔	✔	✖	✖	✖	✖	✖	✔	✔	✖	Images	Active Learning
NeST	✖	✔	✖	✖	✖	✖	✔	✖	✔	✔	Text	Active Learning
HAST (ours)	✔	✔	(✔)	✔	✔	✖	✖	✖	✔	✔	Text	Active Learning

Table 1: Comparing the four most relevant self-training approaches in terms of pseudo-label selection, self-training, and experiment setting. Symbols: ✔: covered; (✔): implicitly covered; ✖: not covered. Abbreviations in the section pseudo-label selection: subsampling (Subs.), uncertainty (Unc.), diversity (Div.). Abbreviations in the section self-training: class balance (Cls. Bal.) and weighting (Weight.). Abbreviations in the section regularization: dropout (D), previous prediction (P), ensembling (E), thresholding (T), and embedding space neighborhood (N).

active learning. Their pseudo-label selection, however, relies on the prediction of previous rounds, which renders subsampling, a common method for handling large datasets or expensive models, impossible. The work of (Xu et al., 2023) is closest to our work due to an intersection of self-training and active learning, and text classification.

3 Active Learning and Self-Training

The goal for active learning is to minimize the annotation effort while maximizing performance regarding some task, such as text classification.

Problem Formulation In pool-based active learning, the training data $\mathcal{X} = \{(x_i)\}_{i=1}^{n}$ is partitioned into two disjoint sets: unlabeled pool $\mathcal{U}$ and labeled pool $\mathcal{L}$ (i.e., $\mathcal{U} \cap \mathcal{L} = \emptyset$). During the active learning loop, the query strategy selects the best ranked instances $\mathcal{X}_q \subseteq \mathcal{U}$, which are then removed from $\mathcal{U}$, labeled by the oracle, and subsequently added to $\mathcal{L}$. We refer to a model as M, and more specifically as M_t when it has been trained after query t. We denote the predicted class distribution of instance x during query t (using model M_{t-1}) as $P_t(y|x)$.

3.1 Incorporating Self-Training

While active learning aims to obtain a small labeled subset, during training it disregards the data in the unlabeled pool. Self-training is a semi-supervised approach that, in addition to the labeled pool's data, leverages (parts of) the unlabeled pool by assigning machine-generated pseudo-labels. Similar to active learning, it *queries* instances according to a criterion such as amongst others, uncertainty. In contrast to active learning, however, the selected instances are pseudo-labeled according to some heuristic instead of labeled by a human annotator.

While for active learning selecting the most uncertain instances has been shown to be effective (Lewis and Gale, 1994; Roy and McCallum, 2001; Schröder et al., 2022), using the most certain instances has been observed to be most beneficial for self-training across several NLP tasks (e.g., Mihalcea (2004); Tomanek and Hahn (2009); Mukherjee and Awadallah (2020)). While these approaches obviously contradict, they can complement each other, as shown in Figure 1: vanilla active learning selects by uncertainty, aiming to find instances that provide the most information to the model, while self-training selects instances by certainty, preferring instances whose pseudo-labels are likely to be correct to increase the set of labeled data.

3.2 Pseudo-Label Regularization

Pseudo-labels are usually derived from a previous model's predictions. As a consequence, pseudo-labels are not guaranteed to be correct, which introduces label noise to the self-training process. In the case of multiple subsequent self-training iterations, this error can propagate over the iterations, resulting in progressively higher levels of noise (Arazo et al., 2020; Yu et al., 2022). Therefore, a key issue for self-training is *pseudo-label regularization*, where methods carefully select or weight pseudo-labels to minimize the expected noise.

3.3 Previous Approaches

Similar to active learning, at the heart of each self-training approach is a strategy that decides which instances are selected—but in this case to be pseudo-labeled. In the following, we present the four most relevant self-training approaches.

UST Uncertainty-aware self-training (UST; Mukherjee and Awadallah (2020)) uses dropout-based stochastic sampling to obtain multiple confidence estimates for each instance. Aggregated scores are then obtained using the BALD measure (Houlsby et al., 2011), based on which instances are sampled in a class-balanced manner.

AcTune AcTune (Yu et al., 2022) aims to obtain a diverse set of instances by preceding the sampling step with weighted K-Means clustering. To overcome the noise of per-instance label variation during self-training iterations, it aggregates pseudo-labels over multiple iterations.

VERIPS The verified pseudo-label selection (VERIPS; (Gilhuber et al., 2022)) starts by selecting instances whose prediction confidence exceeds a fixed threshold. Pseudo-labels are then filtered by a verification step, retaining only the labels for which the current model's predictions match those of a model trained without pseudo-labels.

NeST Neighborhood-regularized self-training (NeST; (Xu et al., 2023)) leverages the embedding space to obtain pseudo-labels that closely match the predicted distribution of their k-nearest neighbors. The individual scores are averaged over multiple active learning iterations for additional stability.

In Table 1, we compare the distinguishing features of all presented approaches, including the approach proposed in Section 4. A striking common feature is that all sample selection mechanisms rely on uncertainty, which has been shown to be very effective both for active learning and self-training (Yu et al., 2022; Xu et al., 2023). The main difference is the pseudo-label selection and regularization.

3.4 Limitations of Previous Approaches

Apart from methodological similarities and differences, we also identified several conceptual shortcomings shared by several approaches, which limit the conclusiveness of existing evaluations.

Unrealistic Evaluation Settings The experiments of UST use validation sets matching the size of the training data, and those of AcTune select the best model based on validation sets of sizes 500 and 1000. Validation sets of these sizes are unrealistic for an active learning scenario, where even training sets of these sizes would exceed the amount of data that we deem to be necessary when evaluating on common text classification benchmarks. Moreover, the classification performance is also supported by an extensive hyperparameter optimization (Yu et al., 2022; Xu et al., 2023), which would not be possible without these validation sets.

Computational Efficiency Since transformer models are known to be computationally expensive, UST and VERIPS incorporate a subsampling mechanism before pseudo-label selection, thereby enabling the use of self-training even with computationally expensive models and large datasets. The pseudo-label selection of both AcTune and NeST, however, relies on predictions from previous self-training iterations. If the current subsample differs from previous ones, these previous predictions will not be available, rendering subsampling impractical for those methods and considerably constraining their computational efficiency.

Confidence Thresholds VERIPS, AcTune, and NeST apply a strict confidence threshold as part of their pseudo-label regularization, which has been shown to have considerable impact on the performance (Gilhuber et al., 2022; Yu et al., 2022). VERIPS and NeST keep it fixed at a high value, while AcTune performs a hyperparameter search for each dataset. The former relies on the availability of high confidence instances, and the latter is infeasible in real-world active learning scenarios, where no validation data exists.

The limitations illustrated above cast doubt on the generalizability of these findings. Furthermore, differences in task (text and image classification) and setting (few-shot and active learning), complicate cross-study comparisons. Therefore, active learning and self-training require further investigation, which we address through a reproduction study.

4 Hard-Label Regularized Self-Training

Based on a methodological analysis (Sections 3.3 and 3.4) and a reproduction study (Section 5), we present **ha**rd-label neighborhood-regularized **s**elf-**t**raining (HAST, pronounced *"haste"*), a novel self-training method that aims to complement active learning with large quantities of pseudo-labels.

The idea of our proposed approach is to rely on the generalization capabilities provided by *contrastive (representation) learning* and on the regularization provided by nearest neighbor relationships in the resulting embedding space. In contrastive learning, training is performed with n-tuples of instances, in the following assumed to be pairs. A pair can be formed between any two instances, and consequently each additional pseudo-labeled instance increases the number of possible pairings. Obviously, when combining this with self-training, this can considerably enhance the effective number of instances—at the risk of introducing noise due to incorrect pseudo-labels.

4.1 Contrastive Representation Learning

Commonly used representations that are obtained from a language model's layers, such as the `[cls]` token (Devlin et al., 2019), rely on the principle that semantically similar inputs will result in similar embedding vectors—but apart from testing for semantic similarity, distance metrics between vectors are often meaningless. Representation learning (Bengio et al., 2013) on the other hand, aims to learn a meaningful space in which the dimensions capture explanatory factors in the data (Bengio et al., 2013; Le-Khac et al., 2020) and distance metrics are rendered meaningful (Le-Khac et al., 2020). Moreover, in *contrastive* representation learning (Carreira-Perpiñán and Hinton, 2005), this is achieved by training on contrasting pairs of instances, where similar instances are pulled together and dissimilar instances are pushed apart in the embedding space. One recent approach is the fine-tuning paradigm SetFit (Tunstall et al., 2022), which uses a Siamese network to train embeddings that are then used as representations in downstream tasks. SetFit has shown incredible effectiveness in the few-shot setting (Tunstall et al., 2022), making it an obvious choice for active learning.

Algorithm 1 AL WITH SELF-TRAINING

Input: unlabeled pool $\mathcal{U}$; labeled pool $\mathcal{L}$; initial model M_0; number of queries Q; batch size B; self-training iterations T

```
1: for q ∈ {1, ..., $Q$} do
2:     $\mathcal{X}_q \leftarrow$ query batch of size $B$ from $\mathcal{U}$
3:     $\mathcal{Y}_q \leftarrow$ labels provided by oracle
4:     $\mathcal{L} \leftarrow \mathcal{L} \cup \{(x_{q,i}, y_{q,i}), ..., (x_{q,i+B}, y_{q,i+B})\}$
5:     $\mathcal{U} \leftarrow \mathcal{U} \setminus \{(x_{q,i}, y_{q,i}), ..., (x_{q,i+B}, y_{q,i+B})\}$
6:     $M_q \leftarrow$ TRAIN($\mathcal{L}$)
7:     $M^*_q \leftarrow$ SELFTRAIN($\mathcal{U}, \mathcal{L}, M_q$,T)
```

Output: Final model M^*_Q

4.2 Active Learning and Self-Training

We incorporate self-training into pool-based active learning by adding a subsequent self-training step after each training step as shown in Algorithm 1. After each query (line 2), a new model is trained (line 6), expressed through a generic TRAIN() function that takes a list of labeled instances, and optionally a list of weights. This is followed by a self-training step (line 7), which may be HAST or one of the previous approaches. The training in line 6 could be skipped after the first iteration, but this additional step "resets" a potentially degraded model, thereby counteracting model instability (Mosbach et al., 2021) and model collapse due to error propagation.

Algorithm 2 SELFTRAIN (HAST)

Input: unlabeled pool $\mathcal{U}$; labeled pool $\mathcal{L}$; current Model M_{t_0}, number of self-training iterations T

```
1: $\mathcal{L}_p = \mathcal{L}; \mathcal{U}_p = \mathcal{U}$
2: for t ∈ {1, ..., $T$} do
3:     $\mathcal{Y}_{q,t} \leftarrow M_{t_0}(\mathcal{U}_p)$
4:     $\mathcal{X}^*_{q,t} \leftarrow \{x_i | x_i \in \mathcal{U}_p$ and $\mathbb{1}_{PL}(x)\}$
5:     $\mathcal{L}_p \leftarrow \mathcal{L}_p \cup \{(x_{t,i}, y_{t,i}), ..., (x_{t,m}, y_{t,m})\}$
6:     $\mathcal{U}_p \leftarrow \mathcal{U}_p \setminus \{(x_{t,i}, y_{t,i}), ..., (x_{t,m}, y_{t,m})\}$
7:     $W_{q,t} \leftarrow$ Weights as described by Eq. 4.
8:     $M^*_{q,t} \leftarrow$ TRAIN($\mathcal{L}_p, W_{q,t}$)
```

Output: Self-trained model $M^*_{q,t}$

4.3 HAST: Pseudo-Labels and Weighting

The proposed approach is intended to exploit the current model to ideally provide larger amounts of pseudo-labels by leveraging the embedding space. Instead of relying on label distributions, we use hard labels, which are obtained by a majority vote of the instance's k nearest neighbors (KNN). The proposed approach is shown in Algorithm 2. Our pseudo-label selection (line 4) takes all instances from the unlabeled pool $\mathcal{U}_p$, where the most confident label crosses the binary decision threshold of 0.5 and the predicted label $\hat{y}_i$ agrees with the k nearest neighbors' majority vote:

$$\mathbb{1}_{PL}(x) = \begin{cases} 1 & \text{if } s_i > 0.5 \wedge \hat{y}_i^{knn} = \hat{y}_i \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where $s_i = P(y_i = \hat{y}_i | x) \in (0, 1]$ is the confidence of the most confident predicted label $\hat{y}_i$, and $\hat{y}^{knn}$ is the label given by a KNN majority vote. Since the predicted label $\hat{y}_i = \text{argmax}\, P(y|x)$ is obtained from the class with highest confidence, this strategy implicitly selects instances with high certainty.

Weighting With the proposed pseudo-label selection strategy, we can obtain a potentially large number of pseudo-labels. This can introduce both

(1) a class imbalance (Henning et al., 2023) among the pseudo-labels and (2) an imbalance between the pseudo-labels and human-annotated labels.

To overcome these issues, we first introduce a weighting term to adjust for class imbalance:

$$z = \frac{N/C - h_c}{max(1, h_c)} \tag{2}$$

where N is the number of all pseudo-labels in $\mathcal{Y}_{q,t}$, C is the number of classes, N/C is the expected number of instances for a balanced class distribution, and h_c is the count of class c in the histogram of $\mathcal{Y}_{q,t}$ over c bins. A max operator in the denominator makes the function well-defined for $h_c \in \mathbb{N}_0$.

This yields a term that is inversely proportional to the current class imbalance. Since the resulting values are unbounded and can potentially grow very large, we apply a sigmoid function to squash the values into the interval $(0, 10)$:

$$\alpha_c = \frac{10}{1 + e^{-z}} \tag{3}$$

To reduce the effect of a possibly excessive number of pseudo-labels and retain some weight on the human-annotated labels, we introduce another term $\beta \in (0, 1]$, which is the labeled-to-unlabeled ratio weight that penalizes pseudo-label weights iff $\beta < 1$. The final weights are then given by:

$$W_i = \alpha_{\hat{y}_i} \cdot \beta \tag{4}$$

Human-annotated instances have a weight of $W_i = 1.0$. Finally, the resulting weights are L^1-normalized, i.e. $\sum_i |W_i| = 1$, and are then element-wise multiplied with the per-instance loss.

5 Experiments

In the experiments, we evaluate the proposed self-training method HAST. Moreover, in an extensive reproduction study, we re-implement the four most relevant previous self-training approaches, evaluating them on active learning for text classification.

Dataset Name (ID)	Type	Classes	Training	Test	Metric
AG's News (AGN)	N	4	120,000	7,600	Acc.
DBPedia-14 (DBP)	T	14	560,000	70,000	F$_1$
IMDB (IMDB)	S	2	25,000	25,000	Acc.
TREC-6 (TREC-6)	Q	6	5,500	500	F$_1$

Table 2: Key information about the examined datasets. Abbreviations: N (News), S (Sentiment), Q (Questions).

5.1 Experiment Setup

The key task in this work is active learning for single-label text classification. Using only 130 instances, the experiments are designed to be both challenging and data efficient. We compare HAST against the reproductions of four previous approaches (UST, AcTune, NeST, and VERIPS) which are compared under equivalent conditions.

Data We evaluate on four established text classification benchmarks, whose key characteristics are displayed in Table 2. AGN and IMDB exhibit a balanced class distribution, and DBP and TREC-6 exhibit an imbalanced class distribution. IMDB is a binary classification problem, while the other datasets are multi-class problems.

Evaluation Following Kirk et al. (2022), we report the classification performance in accuracy for balanced and in macro-F$_1$ for imbalanced datasets.

Classification We evaluate two different models: (1) the paraphrase-mpnet-base SBERT (Reimers and Gurevych, 2019) model, which is fine-tuned using SetFit (Tunstall et al., 2022), and, in order to verify its effectiveness in the non-contrastive setting, (2) a BERT-base model (Devlin et al., 2019) that is trained using vanilla fine-tuning. Both of these models consist of 110M trainable parameters.

Active Learning Models are initialized with 30 instances. Active learning is performed over 10 iterations during each of which 10 more instances are labeled. Following (Hu et al., 2019) and (Yu et al., 2022), the model is trained from scratch after each active learning and self-training iteration. While we do not directly investigate query strategies in this work, they are paramount to active learning and need to be considered. To assess their effect on the self-training process, we evaluate all configurations using two query strategies and a baseline: breaking ties (Scheffer et al., 2001; Luo et al., 2005), contrastive predictions[1] (Margatina et al., 2021), and random sampling.

Self-Training For HAST, we use $k = 5$ and $\beta = 0.1$. For all other strategies, we use the best hyperparameters as reported in the respective publications. A subsample of 16384 instances is drawn

[1] This query strategy is originally called *contrastive active learning*, referring to contrastive differences between predictive distributions of embedding similar instances. In this work, we refer to this strategy as *contrastive predictions* to avoid confusion with contrastive representation learning.

			Datasets			
Query Strategy	**Classifier**	**Self-Training**	**AGN**	**DBP**	**IMDB**	**TREC**
Breaking Ties	BERT	No Self-Training	0.763 0.057	0.619 0.129	0.745 0.030	0.341 0.130
		UST	0.798 0.016	0.645 0.042	0.764 0.100	0.333 0.121
		AcTune	0.806 0.021	0.651 0.054	0.795 0.050	0.434 0.063
		VERIPS	0.834 0.012	0.907 0.047	0.816 0.050	0.540 0.110
		NeST	**0.840 0.013**	**0.918 0.006**	0.783 0.041	**0.580 0.090**
		HAST	0.762 0.055	0.605 0.071	**0.806 0.097**	0.424 0.193
	SetFit	No Self-Training	0.853 0.011	0.973 0.004	0.872 0.009	0.691 0.029
		UST	0.658 0.030	0.483 0.033	0.851 0.028	0.491 0.042
		AcTune	0.863 0.006	0.980 0.003	0.896 0.024	0.642 0.030
		VERIPS	0.859 0.005	0.981 0.002	0.857 0.024	0.730 0.021
		NeST	0.878 0.005	0.981 0.001	**0.927 0.005**	**0.781 0.034**
		HAST	**0.886 0.007**	**0.984 0.001**	0.882 0.040	0.773 0.024
Contrastive Predictions	BERT	No Self-Training	0.635 0.087	0.366 0.034	0.670 0.065	0.210 0.089
		UST	0.712 0.153	0.203 0.108	0.765 0.056	0.311 0.108
		AcTune	0.678 0.178	0.353 0.165	0.684 0.079	0.136 0.076
		Verips	**0.823 0.018**	**0.655 0.093**	0.750 0.067	0.449 0.108
		NeST	0.701 0.085	0.606 0.083	0.772 0.079	**0.517 0.166**
		HAST	0.718 0.071	0.327 0.122	**0.810 0.045**	0.288 0.091
	SetFit	No Self-Training	0.798 0.003	0.678 0.055	0.890 0.016	0.560 0.048
		UST	0.597 0.025	0.400 0.028	0.842 0.015	0.420 0.069
		AcTune	0.811 0.012	0.704 0.052	0.903 0.018	0.622 0.061
		Verips	0.814 0.011	0.776 0.056	0.893 0.013	0.638 0.059
		NeST	0.842 0.006	0.786 0.094	**0.919 0.005**	0.605 0.055
		HAST	**0.849 0.013**	**0.815 0.087**	0.916 0.009	**0.773 0.016**
Random	BERT	No Self-Training	0.760 0.040	0.534 0.061	0.740 0.046	0.276 0.100
		UST	0.797 0.039	0.693 0.083	0.794 0.014	0.298 0.065
		AcTune	0.791 0.050	0.559 0.082	0.801 0.045	0.386 0.123
		VERIPS	0.812 0.023	0.850 0.063	0.813 0.029	0.551 0.074
		NeST	**0.819 0.039**	**0.865 0.037**	0.782 0.022	**0.553 0.071**
		HAST	0.677 0.114	0.650 0.053	**0.831 0.051**	0.514 0.169
	SetFit	No Self-Training	0.848 0.005	0.939 0.031	0.907 0.007	0.676 0.031
		UST	0.659 0.038	0.476 0.039	0.871 0.031	0.491 0.061
		AcTune	0.847 0.014	0.970 0.008	0.918 0.004	0.651 0.025
		VERIPS	0.854 0.010	0.968 0.008	0.921 0.002	0.726 0.017
		NeST	0.860 0.011	0.965 0.004	0.923 0.008	**0.797 0.024**
		HAST	**0.885 0.002**	**0.974 0.006**	**0.926 0.004**	0.738 0.020

Table 3: Classification performance after the final iteration (in accuracy or macro-F_1), broken down per query strategy, classifier, and self-training approach. The reported numbers represent the average over five runs, with the standard deviations shown to the right of each value.

before obtaining pseudo-labels for all strategies supporting subsampling. To minimize the effect of error propagation (as shown in Section 3.2), but also for reasons of computational feasibility, we refrain from consecutive self-training iterations.

5.2 Results

The final classification performance of each configuration is shown in Table 3. We observe self-training to be highly effective, with improvements of up to 29 percentage points compared to active learning without self-training. The best results (in bold) are always achieved by either NeST or HAST. The former wins for most BERT configurations, while the latter wins for most SetFit configurations.

Besides the final performance, it is also crucial to investigate the performance after each active learning iteration, which can be seen in the learning curves depicted in Figure 2. HAST self-training is a strong contender in most settings, especially with SetFit, where it sometimes reaches a performance close to the final value already during the first few iterations. Besides the learning curves, a horizontal line at the top represents the model performance when training on the full train set, showing that several configurations achieve remarkable results at only 130 instances. Moreover, the SetFit models are still slightly superior in this case, except for TREC-6, indicating that larger class imbalances could be a problem (at least for the hyperparame-

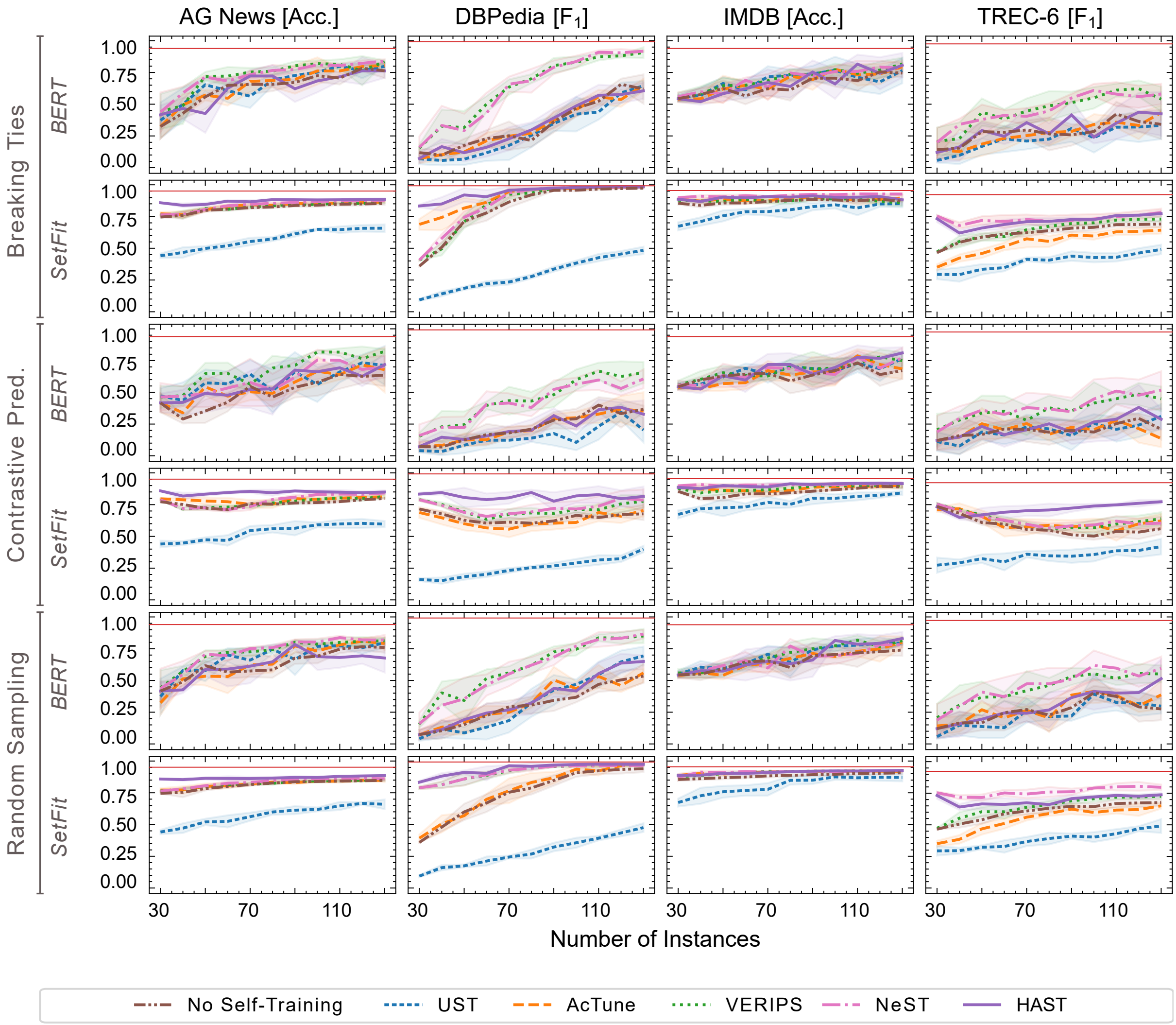

Figure 2: Learning curves per model, query strategy, and dataset, showing the classification performance on the test set. The x-axis shows the number of instances, while the y-axis indicates classification performance. The horizontal (red) line represents the performance of the respective model trained on 100% of the data (without active learning).

Dataset	Approach (Parameters)	N	Score
AGN	ReGen[1] (125M)	0	0.850
[Acc.]	BERT[3] (336M)	525	0.904
	HAST (110M)	130	0.886
DBP	DeBERTa[4] (355M)	0	0.945
[F$_1$]	UST[2] (110M)	420	0.986
	HAST (110M)	130	0.984
IMDB	RoBERTa (355M)[4]	0	0.925
[Acc.]	UST[2] (110M)	60	0.900
	HAST (110M)	130	0.927
TREC-6	GPT3.5 Turbo & RoBERTa[5]	0	0.914
[F$_1$]	BERT[3] (336M)	525	0.968
	HAST (110M)	130	0.773

Table 4: Comparison with previous works that have investigated low-resource methods: [1](Yu et al., 2023), [2](Mukherjee and Awadallah, 2020), [3](Schröder et al., 2022), [4](Gera et al., 2022), [5](Xiao et al., 2023). Column N represents the number of traning instances.

ters that we used). The corresponding area under curve values can be found in Appendix Table 7.

When comparing the results across query strategies, their impact seems to be minimal. The contrastive predictions strategy does not achieve superior performance in any configuration, so we focus on the comparison between breaking ties and random sampling. Breaking ties reaches slightly higher final scores at the expense of marginally lower area under curve.

In Table 4, we compare the best result per dataset to results from literature for sample-efficient methods, including zero-shot, few-shot, and active learning. Except for TREC-6, HAST achieves results close to the state of the art, despite using only very few instances and a comparably small model. Notably, on AGN and DBP, HAST achieves results comparable to methods that used 525 and 420 instances, respectively, while using only 25% and 30% of those instances. HAST outperforms UST, AcTune, and VERIPS, while being on par with NeST. In combination with SetFit models, it even achieves slightly higher accuracy and F_1 scores.

6 Discussion

The experiments have shown that HAST is a highly effective self-training strategy that is able to leverage a large number of pseudo-labels. When combined with SetFit models, HAST outperforms most other approaches and is on par with NeST in classification performance and area under curve. Further investigation revealed that these gains can likely be attributed to the large number of pseudo-labels (as shown in Appendix Table 9), whose acquisition is facilitated by the semantically meaningful embedding space. The large number of pseudo-labels, however, is likely to increase the level of label noise, but HAST demonstrates robustness against this, tolerating up to 20% incorrect labels, particularly when paired with SetFit (see Appendix E.1).

Through an extensive reproduction, we have also investigated the relative strength of UST, AcTune, VERIPS, and NeST in the context of active learning for text classification. The strongest contender is NeST, which is on par with HAST but does not outperform, despite using a computationally more expensive pseudo-label acquisition. The primary impediment of previous approaches appears to be an overreliance on confidence thresholds, which are difficult to optimize in active learning scenarios.

Why did the experiments not incorporate the most recent large language models of 1B or more parameters? With a total runtime of 2600 hours, the experiments are already computationally expensive—despite using models that are considered small by today's standards—and would be infeasible with larger model sizes. Moreover, research has demonstrated that smaller models can outperform larger ones (Hsieh et al., 2023) when properly fine-tuned or distilled. Therefore, we prioritize model efficiency in our active learning research, which ultimately aims to support real-world annotation where smaller models provide a more accessible and practical solution.

Why did the experiments use only a single self-training iteration? While increasing the number of self-training iterations *may* further increase the classification performance, this also runs the risk of degradation (Gera et al., 2022; Xu et al., 2023). For this reason, by using only a single self-training iteration we minimize the risk of degradation, thereby using self-training not to replace but to complement active learning, and in favor of real-world settings at only little additional computational costs.

For which real-world use cases is the proposed method a good fit? As previously stated, when self-training is combined with active learning it introduces another source of error. The most favorable is the transductive setting (Tong and Koller, 2001; Kottke et al., 2023), where the models do not necessarily need to generalize on future unseen data e.g., when active learning is used for labeling corpora in social sciences (Romberg and Escher, 2022) or biomedicine (Nachtegael et al., 2024). For datasets, whose class balance is heavily skewed, it might not be optimal yet, but this remains to be investigated in future research.

Will the reduced need in labeled data ultimately render active learning obsolete? Although our work has shown that strong models can be trained using very few samples, this was conducted on established benchmark datasets that are relatively small in size and less challenging compared to real-world datasets, which may have hundreds of multi-label classes, hierarchies, or highly skewed class distributions. While simpler tasks, such as two-class sentiment analysis, might be solvable with zero-shot learning, more complex problems will still benefit from active learning. Should text classification become able to tackle the majority of problems through zero-shot or few-shot, active learning will remain valuable for refining class definitions by providing high-quality labels for instances where the current model exhibits high uncertainty.

7 Conclusions

In this work, we devise and evaluate HAST, a new self-training approach that is tailored to contrastive learning and aims to generate a large number of pseudo-labels to enhance the efficiency of contrastive training. We reproduce four existing self-training approaches and evaluate all approaches on the task of active learning for text classification. Using only small language models of 110M parameters and 25% of instances used in previous work, the proposed approach achieves results close to the state of the art on three out of four datasets.

Limitations

This study is a not a replication, but a reproduction with slight deviations that provide comparable conditions. While this makes previous approaches comparable for the first time, this also introduces the risk of deviations or errors in the code.

While the overall approach has shown to be highly effective, for an active learning study, it is unfortunate that this seems to be largely caused by data-efficient models leveraging the additional pseudo-labels, and only to a minor degree by the instances selected by the query strategy. Nevertheless, this was previously unknown and motivates further research on finding a query strategy that might be more beneficial for self-training.

Finally, the proposed approach is targeted at single-label classification. Our heuristic for hard-label decisions is not applicable to the multi-label settings and would require a different heuristic.

Ethical Considerations

This work presents a method that reduces annotation efforts and could be used for good or bad—similar to most methods. In either scenario, our method would help to reduce the annotation efforts, however, all of this could be achieved through extensive labeling efforts, without our method.

Moreover, since self-training relies on algorithmically assigned pseudo-labels, the obtained pseudo-label distribution is dependent on the unknown true distribution of the dataset, which could be biased towards certain classes. In this case, self-training might not only be prone to error propagation, but also might propagate class biases.

Acknowledgments

The authors acknowledge the financial support by the Federal Ministry of Education and Research of Germany and by Sächsische Staatsministerium für Wissenschaft, Kultur und Tourismus in the programme Center of Excellence for AI-research "Center for Scalable Data Analytics and Artificial Intelligence Dresden/Leipzig", project identification number: ScaDS.AI.

We would like to thank the Webis Group and the Leipzig Corpora Collection for providing GPU resources. We are especially grateful to the anonymous reviewers for their highly constructive and valuable feedback.

References

Eric Arazo, Diego Ortego, Paul Albert, Noel E. O'Connor, and Kevin McGuinness. 2020. Pseudo-labeling and confirmation bias in deep semi-supervised learning. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

Yoshua Bengio, Aaron C. Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(8):1798–1828.

Miguel Á. Carreira-Perpiñán and Geoffrey Hinton. 2005. On contrastive divergence learning. In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*, volume R5 of *Proceedings of Machine Learning Research*, pages 33–40. PMLR. Reissued by PMLR on 30 March 2021.

Hui Chen, Wei Han, and Soujanya Poria. 2022. SAT: Improving semi-supervised text classification with simple instance-adaptive self-training. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 6141–6146, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Stephen Clark, James Curran, and Miles Osborne. 2003. Bootstrapping POS-taggers using unlabelled data. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 49–55.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Liat Ein-Dor, Alon Halfon, Ariel Gera, Eyal Shnarch, Lena Dankin, Leshem Choshen, Marina Danilevsky, Ranit Aharonov, Yoav Katz, and Noam Slonim. 2020. Active Learning for BERT: An Empirical Study. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7949–7962, Online. Association for Computational Linguistics.

Ariel Gera, Alon Halfon, Eyal Shnarch, Yotam Perlitz, Liat Ein-Dor, and Noam Slonim. 2022. Zero-shot text classification with self-training. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1107–1119, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Sandra Gilhuber, Philipp Jahn, Yunpu Ma, and Thomas Seidl. 2022. VERIPS: verified pseudo-label selection for deep active learning. In *IEEE International Conference on Data Mining, ICDM 2022, Orlando, FL, USA*, pages 951–956. IEEE.

Julius Gonsior, Maik Thiele, and Wolfgang Lehner. 2020. Weakal: Combining active learning and weak supervision. In *Discovery Science*, pages 34–49, Cham. Springer International Publishing.

Sophie Henning, William Beluch, Alexander Fraser, and Annemarie Friedrich. 2023. A survey of methods for addressing class imbalance in deep-learning based natural language processing. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 523–540, Dubrovnik, Croatia. Association for Computational Linguistics.

Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. 2011. Bayesian active learning for classification and preference learning. *arXiv preprint arXiv:1112.5745*.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alex Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. 2023. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 8003–8017, Toronto, Canada. Association for Computational Linguistics.

Peiyun Hu, Zack Lipton, Anima Anandkumar, and Deva Ramanan. 2019. Active learning with partial feedback. In *Proceedings of the 7th International Conference on Learning Representations (ICLR)*.

Hannah Kirk, Bertie Vidgen, and Scott Hale. 2022. Is more data better? Re-thinking the importance of efficiency in abusive language detection with transformers-based active learning. In *Proceedings of the Third Workshop on Threat, Aggression and Cyberbullying (TRAC 2022)*, pages 52–61, Gyeongju, Republic of Korea. Association for Computational Linguistics.

Daniel Kottke, Christoph Sandrock, Georg Krempl, and Bernhard Sick. 2023. A stopping criterion forÂ transductive active learning. In *Machine Learning and Knowledge Discovery in Databases*, pages 468–484, Cham. Springer Nature Switzerland.

Phuc H. Le-Khac, Graham Healy, and Alan F. Smeaton. 2020. Contrastive representation learning: A framework and review. *IEEE Access*, 8:193907–193934.

David D. Lewis and William A. Gale. 1994. A sequential algorithm for training text classifiers. In *Proceedings of the 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 3–12. Springer, ACM/Springer.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING)*, volume 1 of *COLING '02*, pages 1–7, USA. Association for Computational Linguistics.

David Lowell, Zachary C. Lipton, and Byron C. Wallace. 2019. Practical obstacles to deploying active learning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 21–30, Hong Kong, China. Association for Computational Linguistics.

Tong Luo, Kurt Kramer, Dmitry B. Goldgof, Lawrence O. Hall, Scott Samson, Andrew Remsen, and Thomas Hopkins. 2005. Active learning to recognize multiple types of plankton. *Journal of Machine Learning Research (JMLR)*, 6:589–613.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Katerina Margatina, Loic Barrault, and Nikolaos Aletras. 2022. On the importance of effectively adapting pretrained language models for active learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 825–836, Dublin, Ireland. Association for Computational Linguistics.

Katerina Margatina, Giorgos Vernikos, Loïc Barrault, and Nikolaos Aletras. 2021. Active learning by acquiring contrastive examples. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 650–663, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yu Meng, Yunyi Zhang, Jiaxin Huang, Chenyan Xiong, Heng Ji, Chao Zhang, and Jiawei Han. 2020. Text classification using label names only: A language model self-training approach. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9006–9017, Online. Association for Computational Linguistics.

Rada Mihalcea. 2004. Co-training and self-training for word sense disambiguation. In *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004*, pages 33–40, Boston, Massachusetts, USA. Association for Computational Linguistics.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2021. On the stability of fine-tuning BERT: Misconceptions, explanations, and strong baselines. In *International Conference on Learning Representations*.

Subhabrata Mukherjee and Ahmed Awadallah. 2020. Uncertainty-aware self-training for few-shot text classification. In *Advances in Neural Information Processing Systems*, volume 33, pages 21199–21212. Curran Associates, Inc.

Charlotte Nachtegael, Jacopo De Stefani, Anthony Cnudde, and Tom Lenaerts. 2024. DUVEL: an active-learning annotated biomedical corpus for the recognition of oligogenic combinations. *Database*, 2024:baae039.

Barbara Plank. 2022. The "problem" of human label variation: On ground truth in data, modeling and evaluation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 10671–10682, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Ameya Prabhu, Charles Dognin, and Maneesh Singh. 2019. Sampling bias in deep active classification: An empirical study. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4058–4068, Hong Kong, China. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Julia Romberg and Tobias Escher. 2022. Automated topic categorisation of citizens' contributions: Reducing manual labelling efforts through active learning. In *Electronic Government*, pages 369–385, Cham. Springer International Publishing.

Nicholas Roy and Andrew McCallum. 2001. Toward optimal active learning through sampling estimation of error reduction. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML)*, ICML'01, pages 441–448. Morgan Kaufmann Publishers Inc.

Tobias Scheffer, Christian Decomain, and Stefan Wrobel. 2001. Active hidden markov models for information extraction. In *Proceedings of the 4th International Conference on Advances in Intelligent Data Analysis (IDA)*, IDA '01, pages 309–318, Berlin, Heidelberg. Springer-Verlag.

Christopher Schröder, Lydia Müller, Andreas Niekler, and Martin Potthast. 2023. Small-text: Active learning for text classification in python. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics (EACL): System Demonstrations*, pages 84–95, Dubrovnik, Croatia. Association for Computational Linguistics.

Christopher Schröder, Andreas Niekler, and Martin Potthast. 2022. Revisiting uncertainty-based query strategies for active learning with transformers. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2194–2203, Dublin, Ireland. Association for Computational Linguistics.

Henry Scudder. 1965. Probability of error of some adaptive pattern-recognition machines. *IEEE Transactions on Information Theory*, 11(3):363–371.

Manali Sharma, Di Zhuang, and Mustafa Bilgic. 2015. Active learning with rationales for text classification. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 441–451, Denver, Colorado. Association for Computational Linguistics.

Oriane Siméoni, Mateusz Budnik, Yannis Avrithis, and Guillaume Gravier. 2020. Rethinking deep active learning: Using unlabeled data at model training. In *25th International Conference on Pattern Recognition, ICPR 2020, Virtual Event / Milan, Italy, January 10-15, 2021*, pages 1220–1227. IEEE.

Tiberiu Sosea and Cornelia Caragea. 2022. Leveraging training dynamics and self-training for text classification. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 4750–4762, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Katrin Tomanek and Udo Hahn. 2009. Semi-supervised active learning for sequence labeling. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1039–1047, Suntec, Singapore. Association for Computational Linguistics.

Simon Tong and Daphne Koller. 2001. Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research*, 2:45–66.

Manuel Tonneau, Dhaval Adjodah, Joao Palotti, Nir Grinberg, and Samuel Fraiberger. 2022. Multilingual detection of personal employment status on Twitter. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6564–6587, Dublin, Ireland. Association for Computational Linguistics.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *Preprint*, arXiv:2302.13971.

Akim Tsvigun, Artem Shelmanov, Gleb Kuzmin, Leonid Sanochkin, Daniil Larionov, Gleb Gusev, Manvel Avetisian, and Leonid Zhukov. 2022. Towards computationally feasible deep active learning. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1198–1218, Seattle, United States. Association for Computational Linguistics.

Lewis Tunstall, Nils Reimers, Unso Eun Seo Jo, Luke Bates, Daniel Korat, Moshe Wasserblat, and Oren Pereg. 2022. Efficient few-shot learning without prompts. *arXiv preprint arXiv:2209.11055*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Proceedings of the Advances in Neural Information Processing Systems 30 (NeurIPS)*, pages 5998–6008. Curran Associates, Inc., Long Beach, CA, USA.

Tu Vu, Minh-Thang Luong, Quoc Le, Grady Simon, and Mohit Iyyer. 2021. STraTA: Self-training with task augmentation for better few-shot learning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5715–5731, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Lukas Wertz, Jasmina Bogojeska, Katsiaryna Mirylenka, and Jonas Kuhn. 2023. Reinforced active learning for low-resource, domain-specific, multi-label text classification. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 10959–10977, Toronto, Canada. Association for Computational Linguistics.

Ruixuan Xiao, Yiwen Dong, Junbo Zhao, Runze Wu, Minmin Lin, Gang Chen, and Haobo Wang. 2023. FreeAL: Towards human-free active learning in the era of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14520–14535, Singapore. Association for Computational Linguistics.

Ran Xu, Yue Yu, Hejie Cui, Xuan Kan, Yanqiao Zhu, Joyce Ho, Chao Zhang, and Carl Yang. 2023. Neighborhood-regularized self-training for learning with few labels. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(9):10611–10619.

Bishan Yang, Jian-Tao Sun, Tengjiao Wang, and Zheng Chen. 2009. Effective multi-label active learning for text classification. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, pages 917–926, New York, NY, USA. Association for Computing Machinery.

David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *33rd Annual Meeting of the Association for Computational Linguistics*, pages 189–196, Cambridge, Massachusetts, USA. Association for Computational Linguistics.

Zhiquan Ye, Yuxia Geng, Jiaoyan Chen, Jingmin Chen, Xiaoxiao Xu, SuHang Zheng, Feng Wang, Jun Zhang, and Huajun Chen. 2020. Zero-shot text classification via reinforced self-training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3014–3024, Online. Association for Computational Linguistics.

Yue Yu, Lingkai Kong, Jieyu Zhang, Rongzhi Zhang, and Chao Zhang. 2022. AcTune: Uncertainty-based active self-training for active fine-tuning of pretrained language models. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1422–1436, Seattle, United States. Association for Computational Linguistics.

Yue Yu, Yuchen Zhuang, Rongzhi Zhang, Yu Meng, Jiaming Shen, and Chao Zhang. 2023. ReGen: Zero-shot text classification via training data generation with progressive dense retrieval. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 11782–11805, Toronto, Canada. Association for Computational Linguistics.

Michelle Yuan, Hsuan-Tien Lin, and Jordan Boyd-Graber. 2020. Cold-start active learning through self-supervised language modeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7935–7948, Online. Association for Computational Linguistics.

Xia Zeng and Arkaitz Zubiaga. 2023. Active PETs: Active data annotation prioritisation for few-shot claim verification with pattern exploiting training. In *Findings of the Association for Computational Linguistics: EACL 2023*, pages 190–204, Dubrovnik, Croatia. Association for Computational Linguistics.

Mike Zhang and Barbara Plank. 2021. Cartography active learning. In *Findings of the Association for Computational Linguistics (EMNLP Findings)*, pages 395–406, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Proceedings of the Advances in Neural Information Processing Systems 28 (NIPS)*, pages 649–657. Curran Associates, Inc., Montreal, Quebec, Canada.

Ye Zhang, Matthew Lease, and Byron C. Wallace. 2017. Active discriminative text representation learning. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, pages 3386–3392, San Francisco, California, USA. AAAI Press.

Supplementary Material

In the following, we provide details for reproduction (Sections A–D), supplementary analyses (Section E), and an extended discussion (Section F).

A Environment

The experiments were conducted using CUDA 11.2 and a single NVIDIA A100 GPU per run. Experiment code is written in Python and executed in a Python 3.8 environment. All experiment code has been published on Github: https://github.com/chschroeder/self-training-for-sample-efficient-active-learning.

B Software

Our experiments leverage tried and test machine learning libraries: PyTorch (2.2.1), transformers (4.29.2), scikit-learn (1.4.1.post1), setfit (0.7.0), small-text (2.0.0-dev, commit f9be17a0), scipy (1.12.0), numpy (1.26.4). A full list, including transitive dependencies, is included in the Github repository.

The experiment code extends a previous code base (Schröder et al., 2022), which is built around the small-text library (Schröder et al., 2023). We extend this setup with self-training functionality, including the four reproduced strategies.

C Datasets

The experiments used text classification benchmarks that are well-known and also widely used: AG's News (AGN; Zhang et al., 2015), DBPedia (DBP; Zhang et al., 2015), IMDB (Maas et al., 2011), and TREC-6. (Li and Roth, 2002) The raw texts were obtained via the huggingface datasets library. Following (Margatina et al., 2022), we subsampled DBP to 10K instances per class (140K in total) to render the computational efforts (which are outlined in Section E) feasible.

Dataset	Batch Size	Max. Seq. Length
AGN	40	64
DBP	24	128
IMDB	14	512
TREC	40	64

Table 5: Hyperparameter settings for the maximum sequence length (as number of tokens) per dataset.

D Hyperparameters

Self-Training For VERIPS, we used the margin-based variant, which has been shown to be superior to the entropy-based variant (Gilhuber et al., 2022).

Maximum Sequence Length We set the maximum sequence length to the minimum multiple

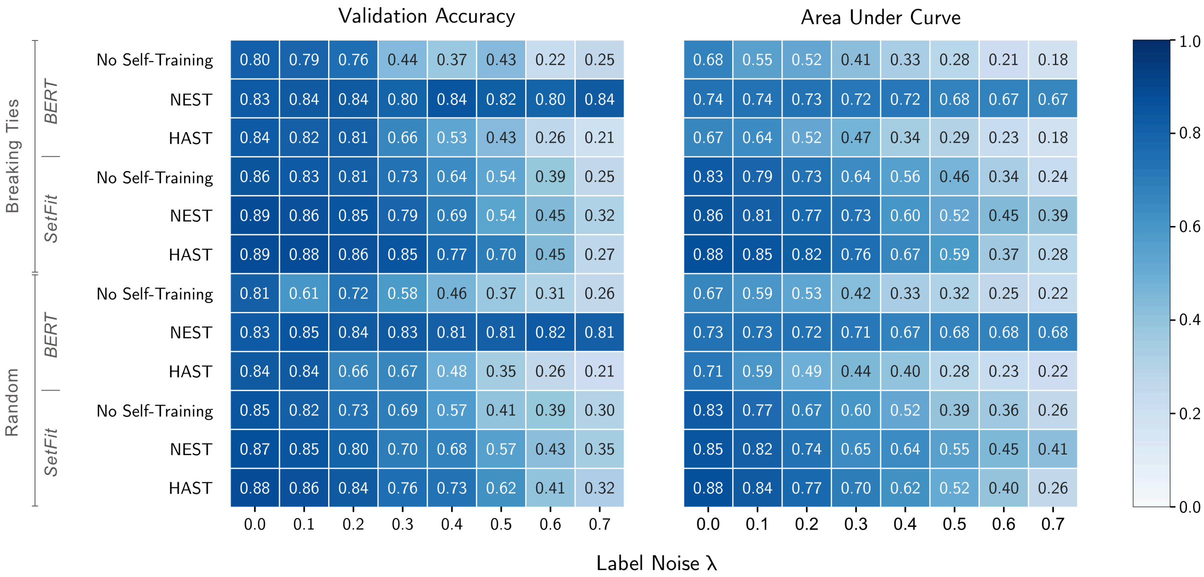

Figure 3: The effect of label noise for NeST and HAST on AGN. Each label is replaced by an incorrect random label with probability λ. The left side shows validation accuracy after the final active learning iteration. The right side shows the respective area under the learning curve for all 10 queries.

power of two for which 95% of the dataset's sentences contain less than or an equal number of tokens, capped at 512 which is an architectural restriction of employed models (see Table 5).

D.1 Classification

BERT We fine-tuned each model from bert-base-uncased using a learning rate of η = 2e−5 and training the model for 15 epochs.

SetFit We follow the original publication (Tunstall et al., 2022) and first train the embeddings, which are subsequently used as features in combination with a logistic regression classification head. The original implementation was extended to support the per-instance weighting.

We use a learning rate of η = 2e−5 and train for 1 epoch, which in the SetFit implementation is defined in iterations through the data. During each iteration, two pairs (one positive and one negative) are formed per labeled instance, which can lead to a steep increase in training data. For the sake of computational efficiency, we scale this parameter inversely to the number of pseudo-labels down to a minimum of 1 iteration.

E Experiments

In Table 7, we provide the final area under curve values for the learning curves shown in Section 2.

Overall Runtime and GPU hours The total runtime of all experiments configuration is 2600 hours.

Individual Computational Costs The average runtime of the training step (including self-training) is shown in Table 8.

Evaluation Metrics We adhere to established active learning evaluation protocols and evaluate both the final classification performance and area under curve. The former is measured in accuracy for balanced datasets and in F_1 for imbalanced datasets. For both metrics, tried and tested implementations from scikit-learn were used.

E.1 Impact of Label Noise

In the simulated active learning experiments, the annotation is realized by a simple lookup of the true labels. In real-world settings, however, answers provided by annotators may be wrong, either due to human label variation (Plank, 2022) or annotators making mistakes. Pseudo-labels are an imperfect heuristic, and especially in combination with self-training, those labels may be wrong—even disregarding human annotation errors, which may introduce additional noise.

For this reason, we investigate the effect of erroneous labeling in the annotation step and introduce a label noise λ, which represents the probability of a label to be wrong, i.e. replaced by random label other than the true label. We investigate the two strongest self-training approaches from Section 5: HAST and NeST. In Figure 3, we present validation accuracy and AUC, broken down by increasing label noise. We find that up to a noise level of $\lambda = 0.2$, HAST is only affected to smaller degree in AUC, while accuracy is only slightly lower. In the two rows where NeST is applied in combination with BERT, self-training fails since NeST is not able to find pseudo-labels, which is why the results are considerably better. This also shows the potential risk of self-training—especially when facing high label noise.

E.2 Instance Weighting Ablation

In Table 6, we present an ablation study over the weighting terms introduced in Section 4.3. Here we use HAST with the best performing query strategy, breaking ties, and ablate (1) class weights (by setting $\alpha = 1$), (2) pseudo-label weights (by setting $\beta = 1$), (3) class and pseudo-label weights (by setting $\alpha = \beta = 1$). Surprisingly, using both weightings simultaneously, does not yield the best results. Pseudo-label down-weighting seems to have more impact in general. Most importantly, it seems that weighting has a larger impact on BERT, while SetFit results are often close—except for the highly imbalanced dataset TREC. Since, as reported in Section 5, even the fully supervised SetFit models seem to perform subpar on TREC-6, this is likely already a problem at the level of the classifier.

F Extended Discussion

Why does HAST introduce a confidence threshold, despite the paper criticized previous methods for this? At some point in every self-training algorithm, a decision on how to assign pseudo-labels is required. We use a threshold of $s_i > 0.5$, the well-known tried and tested binary decision threshold, which is used in many classification settings. Being the middle of the $[0, 1]$ confidence interval, this is the weakest decision criterion possible, and more importantly, it not optimized on the datasets (and not intended to).

Classifier	Self-Training	Datasets			
		AGN	DBP	IMDB	TREC
	Final Accuracy/F$_1$				
BERT	HAST	0.766 0.063	0.604 0.067	0.807 0.109	0.405 0.192
	w/o class weighting ($\alpha = 1.0$)	0.845 0.008	**0.794 0.048**	0.799 0.097	0.589 0.160
	w/o pseudo-label down-weighting ($\beta = 1.0$)	0.855 0.017	**0.794 0.048**	0.695 0.139	**0.601 0.112**
	w/o class weighting and down-weighting ($\alpha = \beta = 1.0$)	**0.859 0.019**	**0.794 0.048**	**0.849 0.016**	0.536 0.091
SetFit	HAST	0.889 0.006	0.984 0.001	0.881 0.045	0.691 0.012
	w/o class weighting ($\alpha = 1.0$)	0.886 0.003	**0.985 0.003**	**0.924 0.004**	0.763 0.015
	w/o pseudo-label down-weighting ($\beta = 1.0$)	**0.889 0.002**	0.983 0.004	0.914 0.009	0.761 0.009
	w/o class weighting and down-weighting ($\alpha = \beta = 1.0$)	0.889 0.002	0.985 0.001	0.902 0.031	**0.785 0.019**
	Area under Curve				
BERT	HAST	0.634 0.034	0.332 0.018	0.670 0.037	0.278 0.029
	w/o class weighting ($\alpha = 1.0$)	0.683 0.012	**0.484 0.037**	**0.711 0.032**	**0.393 0.046**
	w/o pseudo-label down-weighting ($\beta = 1.0$)	0.651 0.042	**0.484 0.037**	0.690 0.029	0.381 0.041
	w/o class weighting and down-weighting ($\alpha = \beta = 1.0$)	**0.691 0.009**	**0.484 0.037**	0.700 0.026	0.392 0.036
SetFit	HAST	**0.873 0.004**	0.942 0.015	**0.898 0.004**	0.636 0.023
	w/o class weighting ($\alpha = 1.0$)	0.871 0.005	**0.951 0.009**	0.891 0.008	**0.737 0.012**
	w/o pseudo-label down-weighting ($\beta = 1.0$)	0.870 0.003	0.937 0.017	0.897 0.008	0.714 0.014
	w/o class weighting and down-weighting ($\alpha = \beta = 1.0$)	0.869 0.005	0.940 0.009	0.897 0.008	0.721 0.020

Table 6: Ablation analysis: final classification performance (top) in accuracy or macro-F$_1$ and area under curve (bottom) when removing different components from the instance weighting (see Section 4.3). Breaking ties was employed as query strategy for all runs and the reported numbers are the average over five runs. The reported numbers represent the average over five runs, with the standard deviations shown to the right of each value.

Strategy	Classifier	Self-Training	Datasets			
			AGN	DBP	IMDB	TREC
Breaking Ties	BERT	No Self-Training	0.638 0.030	0.335 0.021	0.650 0.019	0.285 0.032
		UST	0.664 0.017	0.283 0.030	0.684 0.009	0.231 0.077
		AcTune	0.656 0.028	0.324 0.015	0.676 0.030	0.268 0.033
		VERIPS	0.728 0.019	**0.640 0.026**	**0.701 0.010**	0.465 0.063
		NeST	**0.733 0.011**	0.638 0.047	0.695 0.025	**0.467 0.028**
		HAST	0.634 0.030	0.333 0.018	0.668 0.033	0.304 0.032
	SetFit	No Self-Training	0.818 0.009	0.830 0.016	0.868 0.003	0.628 0.021
		UST	0.573 0.006	0.295 0.012	0.794 0.008	0.394 0.020
		AcTune	0.831 0.006	0.906 0.005	0.886 0.007	0.548 0.021
		VERIPS	0.825 0.006	0.851 0.022	0.877 0.008	0.650 0.017
		NeST	0.840 0.006	0.865 0.016	**0.917 0.002**	**0.728 0.031**
		HAST	**0.871 0.002**	**0.942 0.015**	0.898 0.004	0.711 0.010
Contrastive Predictions	BERT	No Self-Training	0.498 0.036	0.228 0.037	0.640 0.021	0.203 0.026
		UST	0.594 0.041	0.148 0.035	0.665 0.014	0.195 0.046
		AcTune	0.550 0.043	0.222 0.032	0.642 0.015	0.208 0.026
		Verips	**0.678 0.021**	**0.451 0.042**	**0.676 0.026**	0.366 0.058
		NeST	0.598 0.060	0.424 0.037	0.661 0.023	**0.386 0.073**
		HAST	0.566 0.052	0.232 0.021	0.675 0.023	0.233 0.082
	SetFit	No Self-Training	0.757 0.003	0.643 0.014	0.849 0.009	0.573 0.019
		UST	0.537 0.017	0.250 0.014	0.769 0.018	0.345 0.031
		AcTune	0.785 0.011	0.623 0.020	0.877 0.010	0.608 0.046
		Verips	0.765 0.009	0.708 0.022	0.880 0.011	0.621 0.026
		NeST	0.780 0.002	0.722 0.036	**0.907 0.004**	0.622 0.017
		HAST	**0.845 0.007**	**0.814 0.046**	0.904 0.005	**0.714 0.016**

(Continued on next page.)

(Continued from previous page.)

			Datasets			
Strategy	**Classifier**	**Self-Training**	**AGN**	**DBP**	**IMDB**	**TREC**
Random	BERT	No Self-Training	0.630 0.013	0.307 0.028	0.653 0.024	0.263 0.038
		UST	0.681 0.030	0.330 0.034	0.694 0.010	0.233 0.021
		AcTune	0.648 0.023	0.335 0.027	0.658 0.026	0.284 0.031
		VERIPS	0.727 0.013	**0.608 0.051**	**0.697 0.033**	**0.446 0.049**
		NeST	**0.735 0.022**	0.592 0.034	0.676 0.024	0.428 0.056
		HAST	0.625 0.042	0.354 0.027	0.679 0.039	0.301 0.041
	SetFit	No Self-Training	0.814 0.009	0.755 0.038	0.885 0.006	0.608 0.023
		UST	0.577 0.008	0.285 0.020	0.811 0.020	0.381 0.026
		AcTune	0.820 0.012	0.774 0.028	0.912 0.001	0.547 0.018
		VERIPS	0.823 0.007	0.916 0.015	0.912 0.002	0.649 0.021
		NeST	0.834 0.007	0.916 0.012	**0.914 0.005**	**0.763 0.008**
		HAST	**0.868 0.006**	**0.941 0.010**	0.911 0.007	0.693 0.021

Table 7: Area under curve per query strategy, classifier, self-training method, and dataset. For AGN and IMDB the area under the accuracy curve is listed, for DBP and TREC the area under the macro-F_1 curve. The reported numbers represent the average over five runs, with the standard deviations shown to the right of each value.

			Datasets			
Strategy	**Classifier**	**Self-Training**	**AGN**	**DBP**	**IMDB**	**TREC**
Breaking Ties	BERT	UST	352.24 1.11	970.27 3.81	1879.50 3.14	98.25 1.36
		AcTune	301.39 13.73	692.87 5.08	1013.13 42.33	36.61 0.83
		VERIPS	252.08 2.94	723.46 3.57	837.57 1.79	62.17 2.13
		NeST	244.58 1.86	723.03 3.26	846.71 4.91	63.70 0.99
		HAST	240.71 2.12	690.35 3.71	1897.03 89.95	41.48 3.48
	SetFit	UST	716.14 2.34	1312.27 3.68	2655.04 7.94	42.71 0.24
		AcTune	673.86 1.07	1475.65 2.91	1578.38 4.40	17.68 0.48
		VERIPS	453.75 1.66	1021.13 2.43	1238.13 2.12	13.67 0.14
		NeST	465.65 2.61	1011.67 3.52	1242.64 3.68	16.03 0.30
		HAST	677.57 5.97	1123.51 3.71	2579.80 10.94	25.27 0.33
Contrastive Predictions	BERT	UST	353.84 4.89	948.33 7.66	1882.40 5.53	97.87 3.00
		AcTune	353.40 3.46	691.46 1.41	1059.80 8.37	37.00 0.74
		Verips	245.26 1.14	730.82 3.11	834.88 4.35	64.69 0.58
		NeST	250.04 2.00	739.44 1.82	841.60 5.00	64.84 0.59
		HAST	244.35 1.78	694.14 1.68	1967.57 46.02	39.69 0.44
	SetFit	UST	710.34 4.07	1290.36 1.92	2653.14 15.72	41.31 0.75
		AcTune	667.61 1.65	1463.93 1.88	1564.77 5.86	15.07 0.11
		Verips	454.07 1.20	994.89 1.71	1238.64 6.81	13.20 0.16
		NeST	471.62 1.01	1010.48 3.86	1236.69 6.40	14.95 0.18
		HAST	660.91 5.54	1040.48 8.59	2577.87 12.03	23.56 0.63
Random	BERT	UST	350.81 2.66	947.98 1.36	1881.25 3.22	96.76 2.98
		AcTune	320.59 18.36	696.55 4.64	1015.37 31.33	36.29 1.22
		VERIPS	261.08 5.14	726.11 5.94	836.54 2.41	94.99 5.26
		NeST	265.89 5.44	726.38 4.50	847.49 4.60	62.09 0.98
		HAST	240.68 2.14	691.40 0.62	1840.03 62.39	42.86 3.26
	SetFit	UST	720.45 5.89	1304.55 1.78	2663.02 4.80	40.90 0.76
		AcTune	685.87 6.17	1492.17 2.82	1573.35 1.39	18.43 0.28
		VERIPS	457.66 0.43	993.81 3.14	1234.64 2.01	13.15 0.07
		NeST	471.96 1.48	1011.17 2.69	1357.29 8.12	15.29 0.32
		HAST	703.01 7.19	1150.31 7.58	2676.58 14.91	27.17 0.46

Table 8: Mean average training runtime (in seconds) over all iterations. A failed effort to obtain pseudo-labels is counted as zero seconds and therefore reduces the runtime. The reported numbers represent the average over five runs, with the standard deviations shown to the right of each value.

			Datasets			
Strategy	**Classifier**	**Self-Training**	**AGN**	**DBP**	**IMDB**	**TREC**
Breaking Ties	BERT	UST	14.82 38.19	215.92 127.58	6.50 25.40	168.96 45.49
		AcTune	5.23 11.84	0.00 0.00	4.40 13.32	0.00 0.00
		VERIPS	0.00 0.00	0.00 0.00	0.00 0.00	0.00 0.00
		NeST	0.00 0.00	0.00 0.00	6.96 6.91	0.00 0.00
		HAST	620.80 1136.78	0.00 0.00	1172.96 4169.12	138.00 167.35
	SetFit	UST	1.06 0.04	24.77 29.16	1.08 0.06	61.04 6.81
		AcTune	1.72 0.34	2.89 0.64	1.20 0.20	7.20 4.95
		VERIPS	0.00 0.00	0.00 0.00	0.00 0.00	0.00 0.00
		NeST	3.15 1.47	3.76 1.39	2.03 1.06	7.36 4.19
		HAST	1.49 0.45	90.91 116.36	1.14 0.15	292.69 222.55
Contrastive Predictions	BERT	UST	120.0 0.0	420.0 0.0	60.0 0.0	180.0 0.0
		AcTune	24.9 0.2	0.0 0.0	25.0 0.0	0.0 0.0
		Verips	0.0 0.0	0.0 0.0	0.0 0.0	0.0 0.0
		NeST	0.0 0.0	0.0 0.0	536.5 443.3	0.0 0.0
		HAST	1391.0 1085.6	0.0 0.0	13741.9 1268.1	90.0 116.8
	SetFit	UST	120.0 0.0	420.0 0.0	60.0 0.0	180.0 0.0
		AcTune	25.0 0.0	25.0 0.3	25.0 0.0	25.0 0.0
		Verips	0.0 0.0	0.0 0.0	0.0 0.0	0.0 0.0
		NeST	81.6 78.3	19.8 22.7	155.9 102.2	22.8 14.0
		HAST	9178.5 1000.0	1964.9 1312.3	14355.7 306.4	1507.5 337.0
Random	BERT	UST	12.13 33.69	164.45 106.32	5.84 22.03	165.05 46.27
		AcTune	2.90 4.07	0.00 0.00	2.54 7.91	0.00 0.00
		VERIPS	0.00 0.00	0.00 0.00	1.05 0.08	0.00 0.00
		NeST	0.00 0.00	0.00 0.00	6.72 6.12	0.00 0.00
		HAST	469.14 750.86	0.00 0.00	2105.59 5268.35	72.53 79.30
	SetFit	UST	1.06 0.04	23.12 30.56	1.09 0.06	61.51 6.92
		AcTune	1.54 0.30	4.86 2.59	1.23 0.25	10.05 7.46
		VERIPS	0.00 0.00	0.00 0.00	1.18 0.00	0.00 0.00
		NeST	5.07 3.22	13.25 10.74	3.69 3.06	9.46 5.95
		HAST	1.63 0.40	86.84 142.18	1.07 0.06	333.54 255.17

Table 9: Mean average number of pseudo labels over all iterations, broken down per query strategy, classifier, and self-training approach. A value of zero indicates that no pseudo-labels could be selected, mostly due to not exceeding the confidence threshold. The reported numbers represent the average over five runs, with the standard deviations shown to the right of each value.