Neural Networks Against (and For) Self-Training: Classification with Small Labeled and Large Unlabeled Sets

Payam Karisani[1,2]

[1]University of Illinois at Urbana-Champaign
[2]Emory University
payam.karisani@emory.edu

Abstract

We propose a semi-supervised text classifier based on self-training using one positive and one negative property of neural networks. One of the weaknesses of self-training is the semantic drift problem, where noisy pseudo-labels accumulate over iterations and consequently the error rate soars. In order to tackle this challenge, we reshape the role of pseudo-labels and create a hierarchical order of information. In addition, a crucial step in self-training is to use the classifier confidence prediction to select the best candidate pseudo-labels. This step cannot be efficiently done by neural networks, because it is known that their output is poorly calibrated. To overcome this challenge, we propose a hybrid metric to replace the plain confidence measurement. Our metric takes into account the prediction uncertainty via a subsampling technique. We evaluate our model in a set of five standard benchmarks, and show that it significantly outperforms a set of ten diverse baseline models. Furthermore, we show that the improvement achieved by our model is additive to language model pretraining, which is a widely used technique for using unlabeled documents. Our code is available at https://github.com/p-karisani/RST.

1 Introduction

Text classification has achieved tremendous success in the past decade thanks to the advancement in deep neural networks. Even though the introduction of contextual word embeddings and language model pretraining (Peters et al., 2018; Devlin et al., 2019) has greatly reduced the reliance on large manually annotated datasets, the current over-parametrized models are still prone to overfitting. To further reduce this reliance one can use unlabeled data (Abney, 2007; Chapelle et al., 2006). In this article, we use the properties of neural networks and develop a self-training model, termed Robust Self-Training (RST), for low-data regime text classification. Self-training (also known as pseudo-labeling) is iterative (Scudder, 1965; Lee, 2013), and in each iteration unlabeled documents are automatically annotated and augmented with labeled documents.

Previous studies (Carlson et al., 2010; Chen et al., 2013) report that self-training suffers from the semantic drift problem. That is, as the iterations are carried on, spurious pseudo-labels are generated and added to the labeled documents. This eventually distorts the class boundaries and drifts the original class centroids. To address this problem, inspired by the catastrophic forgetting phenomenon in neural networks (McCloskey and Cohen, 1989), we propose a novel procedure to reshape the role of pseudo-labels in the algorithm. We also aim to overcome a weakness of neural networks in this algorithm. Self-training relies on prediction confidence to select the best candidate documents. In this framework, the classifier output is interpreted as prediction confidence (Ruder and Plank, 2018). Self-training performance deteriorates in the settings that the used classifier is unable to accurately estimate the prediction confidence (Rizve et al., 2021). Neural networks suffer from such a problem, because their outputs are mis-calibrated (Guo et al., 2017). To address this problem, we propose a novel metric to replace the plain confidence measurement. Our metric takes into account the prediction uncertainty via a subsampling algorithm.

We use a set of five standard benchmarks to evaluate our model. The selected datasets cover a wide spectrum of documents, ranging from formal news documents to highly informal social media documents. We also compare our model with a set of ten methods, including approaches that use variants of self-training, use multiple classifiers, use multi-view learning, and use various uncertainty metrics. The experiments signify to the superiority of our model. Additionally, we analyze our model and demonstrate that the improvement achieved by our model is additive to the performance of domain

Findings of the Association for Computational Linguistics: ACL 2023, pages 12148–12162
July 9-14, 2023

specific language model pretraining.

The contributions of our work are as follows: **1)** We mitigate the semantic drift problem in self-training by reshaping the role of pseudo-labeled documents and creating a hierarchical order of information. **2)** We enhance the pseudo-label selection in self-training by proposing a novel selection metric to replace the plain confidence measurement. Our metric is particularly advantageous when neural networks are used as the underlying classifier, because these classifiers are overconfident in their predictions (Guo et al., 2017). **3)** Through an extensive set of experiments with five datasets and ten baselines we show that our model is highly resistant to noisy pseudo-labels, yields an additive improvement to domain specific language model pretraining, and outperforms the state of the art.

2 Related Work

Neural networks in self-training. Self-training or pseudo-labeling (Scudder, 1965; Lee, 2013) is a semi-supervised learning algorithm. Previous studies investigate various aspects of this algorithm and aim for filling the niches. For instance, Arazo et al. (2019) integrate MixUp (Zhang et al., 2018a) with the oversampling of labeled documents, Amiri (2019) proposes a new document sampling strategy, Xie et al. (2020b) and He et al. (2020) report that adding noise to pseudo-labels and the hidden layers of a network enhances the model performance–the latter for the sequence generation task, and Zoph et al. (2020) contrast self-training and pretraining and conclude that under certain conditions the former outperforms the latter. Karisani et al. (2020) propose a multi-view self-training model to incorporate domain-knowledge, Pham et al. (2021) propose a feedback loop across self-training iterations, Karamanolakis et al. (2021) propose a model to incorporate weakly supervised domain specific rules, Vu et al. (2021) report that pre-training a model with an auxiliary NLI task enhances self-training, and Li et al. (2021) reduces the variance of pseudo-labels within each class using an angular loss.

As opposed to our research, none of these studies propose a model to maintain a balance between the set of pseudo-labels and the set of manual labels. Additionally, they don't analyze the deterioration of performance during the self-training iterations, and consequently have no defense against this fundamental weakness.

Uncertainty measurement in NLP. Confidence in model prediction, is the amount of trust in the predicted class label compared to the other class labels (Guo et al., 2017). Uncertainty in model prediction, is the amount of trust in the entire prediction regardless of the predicted label (Kendall and Gal, 2017). The research on the efficacy of uncertainty in semi-supervised learning is scarce. Mukherjee and Awadallah (2020) propose to filter out uncertain predictions before the candidate selection step, Rizve et al. (2021) apply a set of thresholds to filter out uncertain and unconfident predictions, and Xu et al. (2021) experiment with various uncertainty metrics and report that uncovering the Heteroscedastic uncertainty (the intrinsic data uncertainty) (Kendall and Gal, 2017) is the best strategy on average.

As opposed to our work, none of these studies propose an integrated metric for selecting pseudo-labels. Additionally, after selecting the pseudo-labels, they don't propose any strategy to restrain the noisy labels from polluting the labeled set.

Ensemble and multi-view models. There exist models that use multiple classifiers, examples include variants of Tri-training (Søgaard, 2010; Ruder and Plank, 2018), variants of co-training (Blum and Mitchell, 1998; Sindhwani et al., 2005; Wu et al., 2018; Karisani et al., 2020), and other ad hoc ensemble models (Li and Zhou, 2007; Hady and Schwenker, 2008; Zhang et al., 2018b).

As opposed to our work, these models rely only on the confidence of classifiers. No coherent uncertainty interpretation has been proposed for them. Additionally, they use ensembling in the prediction stage, whereas, we employ only one classifier for this purpose, which is more resource efficient.

Semantic drift in self-training. Semantic drift in self-training (Curran et al., 2007; Carlson et al., 2010) occurs when spurious pseudo-labels accumulate over time and distort the distribution of labeled documents. In the context of neural networks, the research in this area is sparse. One approach is to avoid pseudo-labels altogether, and use unlabeled documents differently (Gururangan et al., 2019; Xie et al., 2020a; Chen et al., 2020a; Gururangan et al., 2020). Nonetheless, these alternative methods don't necessarily compete with self-training and can co-exist with it inside a framework. To address semantic drift directly, existing approaches mainly aim for explicitly adjusting pseudo-labels. Li et al. (2021) use an angular loss function to project pseudo-labels. Karisani and Karisani (2021)

assume pseudo-labels evolve in a stochastic process and normalize their values. In terms of the role of pseudo-labels in self-training, our algorithm can be taken as the generalized form of the algorithm proposed by Karisani and Karisani (2021).

Connections to consistency regularization. There are two distinctions between our model and consistency training methods (Chen et al., 2020b; Xie et al., 2020a), one in the methodology and another in the objective. In consistency-based regularization methods, data points are manipulated and new data points are generated. As opposed to these methods we don't manipulate data, instead we revise the training steps. Additionally, in consistency based regularization methods the goal is to create a smooth loss surface, so that the class boundaries are easier to adjust and expand to unlabeled data. Our objective is different, we aim to address the model overconfidence, which is why we don't use this step during the training of the classifier (consistency regularization is done during the training), we use it only during the candidate selection. This means that our method doesn't compete with consistency training, and can co-exist with it in a single framework.

3 Proposed Method

In a typical self-training model (Yarowsky, 1995), there is a set L of labeled data, and a set U of unlabeled data. A predictive model is trained on L and is used to probabilistically label U. Then, given a hyper-parameter θ, as the minimum confidence threshold, the confidently labeled documents in U and their associated *pseudo-labels* are selected and added to L. This procedure is iterative. In this framework, there is no constraint on the choice of the underlying model, except that it is required to assign a confidence score to each pseudo-label.

There are two challenges to face in this setting: 1) Self-training suffers from the semantic drift problem (Curran et al., 2007; Carlson et al., 2010). That is, as we increase the number of iterations, the error rate accumulates and the class boundaries are distorted. 2) Neural networks are overconfident in their predictions (Guo et al., 2017; Hendrycks and Gimpel, 2017), and as we discuss in Section 3.2, this shortcoming deteriorates the quality of the pseudo-labels in each iteration.

To address these challenges, we present Robust Self-Training (RST). Algorithm 1 provides an overview of RST , with two classifiers, in Structured English. Lines 12 to 18 demonstrate one iteration of the algorithm, which is repeated till the set of unlabeled documents U is exhausted. The iteration begins by initializing the classifiers C_1 and C_2. Then, it continues by sampling from the set of pseudo-labels S and distilling it (Hinton et al., 2015) into the classifier C_1.[1] Then, another sample from the set of labeled documents L is taken to further train C_1 using Equation 1 (see Section 3.1). These steps are re-taken to train the second classifier C_2. Finally, C_1 and C_2 are used in Equation 2 (see Section 3.2) to label and score the documents in the set U. The top documents are removed from U and are added to the set S. Since we have multiple classifiers labeling each document (in this case two classifiers), we store the average of the outputs in S. On Line 19, the entire set S is used to pretrain the final classifier C, and on Line 20, the set L is used to finetune C using Equation 1. In the following two sections we discuss how our algorithm can address the two aforementioned challenges.

Algorithm 1 Overview of RST

```
1:  procedure RST
2:    Given:
3:      L : Set of labeled documents
4:      U : Set of unlabeled documents
5:    Return:
6:      Trained classifier on L and U
7:    Execute:
8:      Set K to 100 // hyper-parameter (step size)
9:      Set R to 70 // hyper-parameter (sample ratio)
10:     Set S to EMPTY // the set of pseudo-labels
11:     while U is not EMPTY do
12:       Initialize the classifiers C1 and C2
13:       Sample R% of S, order the data as described in
                Section 3.1, and use it to train C1
14:       Sample R% of L and use in Equation 1 for C1
15:       Sample R% of S, order the data as described in
                Section 3.1, and use it to train C2
16:       Sample R% of L and use in Equation 1 for C2
17:       Use C1 and C2 to label U and then score the
                documents using Equation 2
18:       Remove the top K documents from U and add
                them to S
19:     Order the set S as described in Section 3.1, and
            use it to train the classifier C
20:     Use the set L in Equation 1 to further train C
21:     Return C
```

3.1 Overcoming Semantic Drift

An inherent pitfall of the self-training algorithm is the semantic drift problem (Curran et al., 2007; Carlson et al., 2010; Chen et al., 2013), where adding new pseudo-labels ultimately impacts the

[1] In the first iteration the set S is empty, therefore, no distillation is done.

properties of the classes in the set of labeled documents. To mitigate this problem, one solution is to order the training data based on the deemed noise in the labels.[2] Thus, we seek to re-design self-training to undergo such a modification.

Catastrophic forgetting is a problem in neural networks (McCloskey and Cohen, 1989; Kirkpatrick et al., 2017). This problem arises in the continual learning settings, when a series of tasks are sequentially used to train a neural network. It stems from the fact that the weights of the neural network update to serve the objective of the current task, therefore, the information about the current task replaces the information about the previous tasks. We use this property of neural networks to construct a natural hierarchical order of information. Because the pseudo-labels in each iteration are obtained by the model of the previous iteration, it is reasonable to assume that they are noisier than the pseudo-labels in the previous iterations. Based on this argument, we propose to order the pseudo-labels according to the reverse iteration number, and then, use them to train the network of the current iteration. Because it is assumed that the labeled data is noiseless, this set is used at the end of the training to finetune the network. One can assume that the pseudo-labels in this algorithm are used to initialize the network, and the labeled data is used to finetune the network.

To be able to initialize and train the network in each iteration, we store the iteration number that each pseudo-label was added to the pool. We call the set of pseudo-labels the set S, and the set of initial labeled documents the set L. At the beginning of each iteration, we order the pseudo-labels in S and use them to train the network, i.e., Task 1. We store–and use–the last layer logits of the network in classifying the documents in S to be used with a high temperature for initialization. Thus, we essentially distill the knowledge of the previous iterations into the network (Hinton et al., 2015). Additionally, because randomness in creating the batches is an essential ingredient of stochastic gradient descent, while training the network by the pseudo-labels of each iteration, we randomly select a percentage of pseudo-labels from other iterations. Finally, we use the documents in L and minimize the following objective function to further train the network, i.e., Task 2:

$$\mathcal{L}=(1-\lambda)(-\sum_{i=1}^{N}[y_i\log a_i+(1-y_i)\log(1-a_i)])+ \lambda(-\sum_{i=1}^{N}[q_i\log a'_i+(1-q_i)\log(1-a'_i)]), \quad (1)$$

where N is the number of the documents in the set L, y_i is the binary ground truth label of the document d_i, a_i is the output of the network after the softmax layer, a'_i is the output of the network after the softmax layer with a high temperature (Hinton et al., 2015), and q_i is the output of the network with the same temperature as the previous case right before the current task begins. Note that a'_i and q_i are different, the former refers to the current output of the network, while the weights are still being updated to meet the objective of the current task, and the latter refers to the output of the network after it was trained by the documents in the set S and before it was trained by the documents in the set L. λ is a penalty term ($0 \leq \lambda \leq 1$).

The first term in the loss function is the regular cross entropy between the ground truth labels and the output class distributions. The second term is the cross entropy between the current output class distributions and the class distributions that were obtained after training the network by the pseudo-labels. Intuitively, the goal of the second term is to prevent the first term from fully erasing the knowledge already stored in the network, i.e., the knowledge obtained from the pseudo-labels.

One advantage of employing pseudo-labels to initialize the network, as we described in this section, is that if during the self-training iterations due to the growing size of the set S the newly added pseudo-labels become highly noisy, the first term in the objective function yields stronger gradients and will automatically dampen the effect of these examples. In fact, we show that given this mechanism, there is no need to validate the number of self-training iterations anymore, and one can label and use the entire set U. Whereas, doing so in the regular self-training causes semantic drift.

3.2 Addressing Overconfidence

The performance in each self-training iteration heavily depends on the quality of the pseudo-labels added to the training data in the previous iterations. Neural networks are prone to assigning high posterior probabilities even to the out of distribution

[2] One can also reduce the importance of the pseudo-labels, which we use as a baseline.

data points (Hendrycks and Gimpel, 2017). This means, with a high probability, the mislabeled documents can be confidently assigned to the opposite class and can be selected as the best candidate documents. The effect of these spurious labels can accumulate over iterations and eventually deteriorate the performance.[3]

To address this issue, in this section we propose a novel selection criterion to replace the plain confidence metric. Our criterion takes into account the uncertainty in the classifier output. The core idea of our algorithm is to determine whether the output class distributions of a candidate document under multiple different subsamples of the set L are consistent.[4] A small divergence–while having distinctly different training sets–indicates that there are strong similarities between the candidate document and the set L. A high confidence that occurs due to the poor calibration of neural network outputs, and not because of the qualities of data, is less likely to re-occur under multiple sample sets.

To implement this idea, we note that the selection criterion must be proportional to model confidence and disproportional to output uncertainty. Below we propose a metric that follows our desired criteria:

$$Score(d) = \frac{\prod_{i=1}^{m}(1-\hat{H}(P_{a_i}))+\alpha}{GJS(P_{a_1},\ldots,P_{a_m})+\alpha}, \qquad (2)$$

where d is the candidate document; P_{a_i} is the output distribution of the classifier C_i trained on the *i-th* subsample; $\hat{H}(a_i)$ is the normalized entropy of the class distribution; GJS is the generalized Jensen-Shannon distance between the class distributions $P_{a_1}, \ldots, P_{a_m}$; m is the number of subsamples—in Algorithm 1 m equals 2—; and α is a smoothing factor—we set it to 1×10^{-4} in all the experiments. Depending on the value of α, the equation results in $Score(d) \in (0, +\infty)$.

The normalized entropy (Hassibi and Shadbakht, 2007) of a random variable is the entropy of the random variable divided by its maximum entropy:

$$\hat{H}(X) = -\sum^{n} p(X)\frac{\log p(X)}{\log n},$$

where n is the number of classes. We use the normalized variant instead of the regular Shannon entropy to scale the quantity between 0 and 1. The generalized Jensen-Shannon distance (Lin, 1991) measures the diversity between a set of distributions, and is calculated as follows:

$$GJS(P_{a_1},\ldots,P_{a_m}) = H(\overline{P}) - \frac{1}{m}\sum_{i=1}^{m} H(P_{a_i}),$$

where $H(\bullet)$ is the Shannon entropy, and $\overline{P}$ is the mean of the distributions. The mean is calculated as follows:

$$\overline{P} = \frac{1}{m}\sum_{i=1}^{m} P_{a_i}.$$

The numerator in Equation 2 represents the confidence of the classifiers. Higher confidence in the classification yields lower entropy in the class predictions, and hence, results in a higher score. The denominator in Equation 2 represents the output uncertainty. Using Equation 2 we can score the documents in the set U, and select the top documents and their associated pseudo-labels to be added to the set L–we assume all classifiers agree on the labels of the top candidate documents.[5]

So far we discussed binary classification problems. Extending our method to multi-class tasks is trivial. To do so, we only need to replace the binomial cross entropy in Equation 1 with a multinomial cross entropy. Note that Equation 2 remains intact, because it is agnostic to the number of classes.

3.3 Computational Complexity

During the experiments, we observed that even with two subsamples our model outperforms existing baselines. Therefore, we used only two classifiers in all the experiments. In terms of implementation, our model has two variants: a sequential variation and a parallel variation. In the sequential setting, the classifier C_1 is trained on the sets S and L, and then it is used to label the set U. The pseudo-labels are stored and the classifier is removed from the memory. This process is repeated for the classifier C_2 to obtain the second set of pseudo-labels. The two sets of pseudo-labels are processed using Equation 1, and the sets S and U

[3]Throttling is used to reduce this effect (Abney, 2007), which we use in the experiments. Our solution specifically reduces the noise introduced by an overconfident model.

[4]In terms of runtime and memory consumption our model is comparable to existing semi-supervised learning models. In fact, three of our baselines consist of model ensembling (they use two or more classifiers), which is a common practice in semi-supervised learning.

[5]We did not observe an example that violates this assumption in the experiments. Nonetheless, such an example can be taken as noise and can be discarded.

are updated. In this setting, the memory footprint is identical to that of the regular self-training and the run-time is $2\times$ slower; because each iteration involves training both networks. In the parallel setting, both classifiers C_1 and C_2 can be trained at the same time to obtain the sets of pseudo-labels. In this case, our model has $2\times$ more parameters, because both networks should be stored in memory. Since in the parallel case the two networks do not communicate, the run-time is significantly shorter than the sequential case–it is easily comparable to that of the regular self-training.

4 Experimental Setup

In the current and in the next sections we describe our experimental setup and our results.

4.1 Datasets

We evaluate our model in the sentiment analysis task, in the news classification task, in detecting the reports of medical drug side-effects (the ADR task), and in detecting the reports of product consumption.

In the sentiment analysis task, we use the Amazon dataset (Blitzer et al., 2007) and the Yelp dataset (Zhang et al., 2015a). In the news classification task, we use the AG-News dataset (Zhang et al., 2015b) which is a multi-class classification task with four classes. In the ADR task, we use the dataset introduced by Weissenbacher and Gonzalez-Hernandez (2019) prepared for an ACL 2019 Shared Task. In the product consumption task, we use the dataset introduced by Huang et al. (2017). We specifically use a diverse set of datasets in the experiments to comprehensively evaluate our model. The datasets cover short and long documents. They also cover balanced, imbalanced, and extremely imbalanced tasks. They contain a multi-class task. They also contain social media classification tasks, which reportedly suffer from noisy content (Karisani and Karisani, 2020; Karisani et al., 2022).

The Amazon dataset is accompanied by a set of unlabeled documents. In Yelp and AG-News datasets (for each one separately) we take a set of 10K unused training documents as unlabeled data. For ADR and Product datasets (for each one separately) we used the Twitter API and collected 10K in-domain documents[6] to be used by the models as unlabeled data.

4.2 Baselines

We compare our model with a set of ten diverse models.

Baseline (2019). We include the pretrained BERT model (base version) followed by one layer fully connected network, and a softmax layer (Devlin et al., 2019; Wolf et al., 2019). We follow the settings suggested in the reference to set-up the model. This baseline is finetuned on the training set and evaluated on the test set.

Self-train (1995, 2018). We include the neural self-training model (Yarowsky, 1995; Ruder and Plank, 2018). Based on the confidence of the classifier the top candidate pseudo-labels are selected and added to the labeled data–see the next section for the details. We use one instance of *Baseline* as the classifier in this model.

Tri-train+ (2010, 2018). We include the model introduced by Søgaard (2010) called tri-training with disagreement. This model is the enhanced variant of tri-training model (Zhi-Hua Zhou and Ming Li, 2005), and was shown to be more efficient (Ruder and Plank, 2018). We use three instantiations of *Baseline* with different initializations in this model.

Mutual-learn (2018). We include the model introduced by Zhang et al. (2018b). This model is based on the idea of raising the entropy of neural predictions to improve generalization (Pereyra et al., 2017). We use two instantiations of *Baseline* with different initializations in this model.

Spaced-rep (2019). We include the model introduced by Amiri (2019). This model is based on the Leitner learning system. In each iteration it selects the easiest and most informative documents.

Co-Decomp (2020). We include the model introduced by Karisani et al. (2020). In this model, which is a multi-view semi-supervised method, the task is decomposed into a set of sub-tasks, and then, their results are aggregated. We use two instantiations of *Baseline* in this model.

HAU (2021). Xu et al. (2021) experiment with various uncertainty and confidence measurement methods in two tasks, and report that on average Aleatoric Heteroscedastic Uncertainty metric outperforms other measurement methods. We include this method in our experiments.

[6]We used a set of related keywords to collect the documents. Depending on the subject, collecting this number of documents may take between a few days to a few weeks. It took us about 10 days to collect 10K dissimilar related documents.

UPS (2021). We include the model proposed by Rizve et al. (2021). This model uses a gating mechanism using thresholds to filter out uncertain and unconfident pseudo-labels. Then uses the regular cross entropy for the most confident data points, and another loss called negative cross entropy for the least confident data points.

BDD (2021). We include the model introduced by Li et al. (2021). This model uses an angular loss function to reduce the variance of label angels by transforming the values of pseudo-labels. Their hypothesis is that reducing the variance of model predictions should enhance model performance.

Sel-Reg (2022). We include the method by Kim and Lee (2022). They propose a regularizer to reduce the confirmation bias in successive pseudo-labeling iterations. Their core idea is to diversify the selection of pseudo-labels using an entropy-based loss term.

4.3 Experimental Details

In all the models we use pretrained BERT (the base variant) as the underlying classifier. This setting, which is realistic, makes any improvement over the naive baseline very difficult, because BERT already performs well with small labeled data (Devlin et al., 2019). On the other hand, because all the models have an identical pretrained network their comparison is completely fair.

All the models employ throttling (Abney, 2007) with confidence thresholding–minimum of 0.9 as the cutoff. We also use a model similar to linear growth sampling (Saito et al., 2017) for augmenting the labeled data with unlabeled data, i.e., in each iteration, we sample at most 10% of the current set of labeled data. We use the optimizer suggested by Devlin et al. (2019) with the batch size of 32–Adam with a linear scheduler.

Augmenting the entire set of unlabeled data with labeled data causes semantic drift in *self-training*. Karisani et al. (2020) show that *Co-Decomp* suffers from the same problem. Thus, we treated the number of pseudo-labels as the hyper-parameter in these models and in each experiment used 20% of the training set as the validation set to find the best value. We tuned all of the models for the F1 measure. We found that the optimal values depend on the task and the training sets. *Tri-training+* has an internal stopping criterion, and *Mutual-learn* uses the entire set of unlabeled data to regulate the confidences of the two classifiers. *Spaced-rep* and *BDD* rely on a validation set for candidate selection. Thus, we allocated 20% of the labeled set for this purpose. The rest of the settings are identical to what is suggested by Amiri (2019) and Li et al. (2021).

There are four hyper-parameters in our model: the value of softmax temperature in the distillation processes, the ratio of sampling, the value of λ in the objective function (Equation 1), and the number of classifiers. We set the values of the temperature and the sample size to 2 and 70% respectively across all the experiments. We tuned the value of λ in Product training set, and fixed it across all the experiments–$\lambda \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$. The optimal value of λ is 0.3, which assigns a higher weight to the first term in our loss function. Unless otherwise stated, in all the experiments we use two classifiers in our model.

To evaluate the models in a semi-supervised setting we adopt the standard practice in the literature (Nigam et al., 2000), thus, we use the stratified random sampling to sample a small set from the original training data to be used as the training set for the models. We repeat all the experiments 3 times with different random seeds, and report the average of the results.

Evaluation metrics. Amazon and Yelp datasets are balanced benchmarks, we report accuracy in these datasets. AG-News dataset is a multi-class task, following Gururangan et al. (2020) we report macro-F1 in this dataset. ADR and Product datasets are imbalanced. Following the argument made by Mccreadie et al. (2019) about imbalanced datasets, we report the F1 measure in the minority (the positive) class to account for both the quality and the coverage of the models.

5 Results and Analysis

5.1 Main Results

Table 1 reports the results of RST and the baselines in all the datasets. We observe that RST in all the cases is either the best or on a par with the best model. We particularly see that the improvement is substantial in ADR dataset. This is, in part, due to the skewed class distributions in this dataset. Our model efficiently utilizes the entire set of unlabeled documents resulting in a higher recall, and at the same time, maintaining a high precision. We also inspected the documents in ADR task and observed that they are significantly more diverse than the ones in the other four tasks. This quality of ADR

# Doc	Model	Amaz. Acc	Yelp Acc	AG-N. F1	ADR F1	Prod. F1
300	*Baseline*	0.815	0.891	0.863	0.238	0.728
	Self-train	0.833	0.883	0.871	0.303	0.731
	Tri-train+	0.867	0.914	0.873	0.306	0.734
	Mut-learn	0.851	0.908	0.877	0.024	0.753
	Space-rep	0.860	0.899	0.872	0.258	0.727
	Co-Deco.	-	-	-	0.310	0.754
	HAU	0.867	0.912	0.873	0.309	0.753
	UPS	0.870	0.910	0.877	0.323	0.755
	BDD	0.845	0.892	0.876	0.291	0.734
	Sel-Reg	0.867	0.912	0.886	0.116	0.750
	RST	**0.881**	**0.926**	**0.888**	**0.386**	**0.767**
500	*Baseline*	0.859	0.917	0.883	0.312	0.740
	Self-train	0.865	0.916	0.885	0.335	0.741
	Tri-train+	0.880	0.923	0.888	0.365	0.758
	Mut-learn	0.880	0.920	0.889	0.108	0.767
	Space-rep	0.862	0.917	0.888	0.295	0.737
	Co-Deco.	-	-	-	0.345	0.766
	HAU	0.879	0.917	0.882	0.349	0.767
	UPS	0.878	0.918	0.888	0.334	0.771
	BDD	0.859	0.891	0.878	0.312	0.741
	Sel-Reg	0.876	0.912	**0.892**	0.178	0.770
	RST	**0.891**	**0.928**	0.891	**0.421**	**0.783**

Table 1: Performance of RST compared to the baselines in all the datasets. We follow previous studies and report Accuracy in Amazon and Yelp datasets; and report F1 in AG-News, ADR, and Product datasets. The models were trained on 300 and 500 labeled documents.

makes it specifically susceptible to the number of training examples. We also note that *Mutual-learn* completely fails to learn in this dataset. Our investigations revealed that the extreme class imbalance is the underlying reason.[7]

5.2 Empirical Analysis

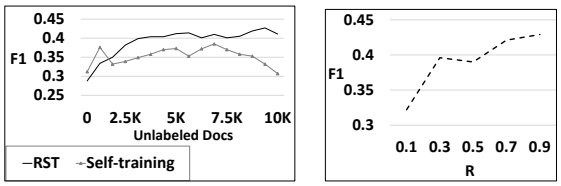
In this section, we contrast RST with domain specific language model pretraining, analyze the resistance of it to semantic drift, report an ablation study on the efficacy of the individual modules, examine the pretraining mechanism in RST, analyze the hyper-parameter sensitivity, and analyze the convergence performance.

We begin by validating our claim that our model can be complementary to language model pretraining (see Section 1). We compare RST to domain specific language model pretraining (Gururangan et al., 2020). Thus, we use the unlabeled data described in Section 4 to pretrain *Baseline* model using the masked language model and the next sentence prediction tasks (Devlin et al., 2019). Table 2 reports the results of this experiment. We observe that the combination of RST and pretraining yields an additional improvement. This experiment and the next ones require running models for multiple times. We carried them out in the ADR dataset with 500 initial labeled documents.

[7]We subsampled from the positive set in Product dataset and constructed a highly imbalanced dataset, this model yielded the same results in this case too.

Model	F1	Precision	Recall
DS-pretraining	0.352	0.330	0.421
RST	0.421	0.344	0.548
Combined	0.443	0.397	0.507

Table 2: Results of domain specific language model pretraining (*DS-pretraining*), RST, and their combination.



(a) RST vs self-training (b) F1 vs sample-ratio

Figure 1: **1a)** F1 of RST and *Self-pretraining* at varying unlabeled set sizes. **1b)** The sensitivity of RST to the sample ratio.

To demonstrate the robustness of RST against semantic drift we report the performance of RST at varying number of added unlabeled documents during the bootstrapping iterations. The results are shown in Figure 1a. We observe that in this regard our model is more robust compared to *Self-training* baseline. We also see that our model reaches a plateau at about 3,500 unlabeled documents. Given that 10K unlabeled documents, used in our experiments, is a relatively large set for unsupervised text classification experiments (Ruder and Plank, 2018), this demonstrates that RST is also data efficient.[8]

Next, we report an ablation study on the efficacy of subsampling and pretraining steps. To do so, we replace subsampling with the regular confidence thresholding, and in another experiment, replace pretraining with the regular data augmentation. Table 3 reports the results. We see that both strategies are effective, although pretraining makes a greater contribution. A fundamental question to answer is whether the effect of pretraining can be achieved by assigning a lower weight to pseudo-labels and augmenting them with labeled data. Table 4 reports the results of this experiment when we replace pretraining with *weighted augmentation* in RST –we assigned the weight of 0.5 to the pseudo-labels.[9] We see that the performance substantially deteriorates, signifying the efficacy of pretraining strategy.

[8]The run-time of our model with 10,000 unlabeled documents was less than 3 hours using NVIDIA Titan RTX GPUs. We implemented the sequential variation of our model.

[9]Lower or higher weights does not yield an improvement.

Model	F1	Precision	Recall
RST	0.421	0.344	0.548
RST w/o subsampling	0.394	0.289	0.624
RST w/o pretraining	0.357	0.292	0.498

Table 3: Ablation study on the efficacy of subsampling and pretraining techniques.

Model	F1	Precision	Recall
RST	0.421	0.344	0.548
Weighted augmentation	0.365	0.320	0.470

Table 4: F1, Precision, and Recall of RST when pretraining is replaced with weighted data augmentation.

We now focus on hyper-parameter sensitivity. Figure 1b reports the sensitivity of our model to the sampling ratio in the subsampling stage. We see that after a certain threshold the performance reaches a plateau and the increase in performance is negligible. Figure 3a reports the performance of RST at varying values of λ in the objective function–Equation 1. This coefficient governs the impact of pseudo-labels. We see that as the value of λ decreases, and a higher weight is assigned to the first term, the performance improves and ultimately drops again. This signifies the efficacy of our loss function, and verifies our argument in Section 3.1.

As we stated earlier, in all the experiments we used two classifiers in our model. To demonstrate the sensitivity of our model to the number of classifiers, we report the performance of RST with varying number of classifiers. Figure 2 illustrates the results. We see that by adding one more classifier our model can achieve slightly better results, however, after this cut-off the performance doesn't further improve.

Our loss function (Equation 1) has two terms. The second term in the loss function ties the current training stage (using labeled data) to the training in the previous stage (using pseudo-labels). This raises the question whether this dependency makes the convergence speed slower. To answer this question, we replaced the entire objective with the regular cross entropy on labeled data. Figure 3b reports the results. We see that in terms of convergence, RST is faster and more stable. This is perhaps due to catastrophic forgetting. Training on labeled data interferes with the already stored knowledge in the network and results in the fluctuations that we see in the new learning curve.

Table 1 compares our model with multiple baselines including several ensemble models, e.g., *Tri-train+*, *Mut-learn*, *Co-Deco.*, and *HAU*. As a reference point, one may still like to see how our model compares with an ensemble model armed with an entropy selection metric. Table 5 reports the results of this experiment. We see that RST outperforms such a model, verifying our claims.

Model	F1	Precision	Recall
RST	0.421	0.344	0.548
Tri-training with entropy	0.351	0.324	0.383

Table 5: F1, Precision, and Recall of RST compared to Tri-training. The Tri-training selection criterion is to select the pseudo-labels that have the least entropy.

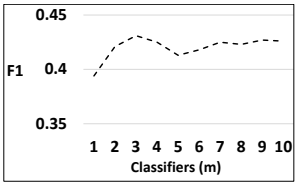
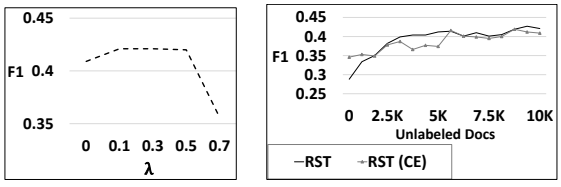


Figure 2: The sensitivity of RST to the number of classifiers. We see that our model reaches the highest performance when three classifiers are used.

In summary, we evaluated our model in five standard datasets under two settings and compared with ten strong baselines. We showed that in all the cases our model is either the best or on a par with the best model. We plan to investigate the applicability of our model in cross-lingual settings.

6 Conclusions

In this paper we proposed a semi-supervised text classifier. Our model is based on the self-training paradigm and employs neural network properties to enhance the bootstrapping procedure. Specifically, we use a subsampling technique to overcome the poor calibration of neural networks and to improve the candidate selection. Then, we exploit the catastrophic forgetting phenomenon in neural networks to alleviate the semantic drift problem. We evaluated our model in five public datasets and showed that it outperforms ten baselines.



(a) F1 vs λ (b) RST vs RST (CE)

Figure 3: **3a)** The sensitivity of RST to the penalty term λ. **3b)** The convergence rate of RST when we use the regular cross entropy instead of our loss function. The modified method is denoted by RST (CE).

Limitations

Our model is evaluated in standard English datasets for classification. As we stated earlier we plan to investigate the cross lingual setting in the next step.

The iterative nature of self-training imposes a high cost on the experiments. This has led to a few common practices. Most existing studies (including all the studies that we used as baselines) employ one underlying classifier to carry out the experiments–i.e., BERT or RNNs. This practice albeit limiting, is justified by the argument that if an algorithm does not make any assumption about the underlying structure of the classifier, then one can safely select the best available classifier and use it in the experiments. We used BERT in our experiments.

Another limitation is that, which is again stemmed from the high cost of self-training, one is typically forced to select a few sample sizes as labeled sets to carry out the experiments–e.g., 100 or 300. This is in contrast to similar research areas, such as Active Learning, when one can usually afford to report a learning curve to illustrate the performance with a few training examples all the way to using the full labeled dataset. Given that we have 10 baselines, we reported the performance with 300 and 500 labeled examples.

References

Steven Abney. 2007. *Semisupervised Learning for Computational Linguistics*, 1st edition. Chapman & Hall/CRC.

Hadi Amiri. 2019. Neural self-training through spaced repetition. In *Proceedings of the 2019 Conference of NAACL*, pages 21–31, Minneapolis, Minnesota.

Eric Arazo, Diego Ortego, Paul Albert, Noel E. O'Connor, and Kevin McGuinness. 2019. Pseudo-labeling and confirmation bias in deep semi-supervised learning. *CoRR*, abs/1908.02983.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *ACL 2007, Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics, June 23-30, 2007, Prague, Czech Republic*. The Association for Computational Linguistics.

Avrim Blum and Tom M. Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory, COLT 1998, Madison, Wisconsin, USA, July 24-26, 1998.*, pages 92–100.

Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka, and Tom M. Mitchell. 2010. Toward an architecture for never-ending language learning. In *Proceedings of the Twenty-Fourth AAAI*, page 1306–1313.

Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien, editors. 2006. *Semi-Supervised Learning*. The MIT Press.

Jiaao Chen, Zichao Yang, and Diyi Yang. 2020a. Mixtext: Linguistically-informed interpolation of hidden space for semi-supervised text classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 2147–2157. Association for Computational Linguistics.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. 2020b. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 1597–1607. PMLR.

Xinlei Chen, Abhinav Shrivastava, and Abhinav Gupta. 2013. Neil: Extracting visual knowledge from web data. In *The IEEE International Conference on Computer Vision (ICCV)*.

James R Curran, Tara Murphy, and Bernhard Scholz. 2007. Minimising semantic drift with mutual exclusion bootstrapping. In *Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics*, volume 6, pages 172–180. Bali.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proc of the 2019 NAACL*, pages 4171–4186.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 1321–1330. JMLR.org.

Suchin Gururangan, Tam Dang, Dallas Card, and Noah A. Smith. 2019. Variational pretraining for semi-supervised text classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5880–5894, Florence, Italy. Association for Computational Linguistics.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of ACL*.

Mohamed Farouk Abdel Hady and Friedhelm Schwenker. 2008. Co-training by committee: A generalized framework for semi-supervised learning with committees. *Int. J. Software and Informatics*, 2(2):95–124.

Babak Hassibi and Sormeh Shadbakht. 2007. Normalized entropy vectors, network information theory and convex optimization. In *2007 IEEE Information Theory Workshop on Information Theory for Wireless Networks*, pages 1–5. IEEE.

Junxian He, Jiatao Gu, Jiajun Shen, and Marc'Aurelio Ranzato. 2020. Revisiting self-training for neural sequence generation. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Dan Hendrycks and Kevin Gimpel. 2017. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Xiaolei Huang, Michael C Smith, Michael J Paul, Dmytro Ryzhkov, Sandra C Quinn, David A Broniatowski, and Mark Dredze. 2017. Examining patterns of influenza vaccination in social media. In *Workshops at the 31st AAAI*.

Giannis Karamanolakis, Subhabrata Mukherjee, Guoqing Zheng, and Ahmed Hassan Awadallah. 2021. Self-training with weak supervision. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 845–863. Association for Computational Linguistics.

Negin Karisani and Payam Karisani. 2020. Mining coronavirus (covid-19) posts in social media. *arXiv preprint arXiv:2004.06778*.

Payam Karisani, Joyce Ho, and Eugene Agichtein. 2020. Domain-guided task decomposition with self-training for detecting personal events in social media. In *Proceedings of The Web Conference 2020*, WWW '20, page 2411–2420. Association for Computing Machinery.

Payam Karisani and Negin Karisani. 2021. Semi-supervised text classification via self-pretraining. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, WSDM '21, page 40–48. Association for Computing Machinery.

Payam Karisani, Negin Karisani, and Li Xiong. 2022. Multi-view active learning for short text classification in user-generated data. In *Findings of the Association for Computational Linguistics: EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 6441–6453. Association for Computational Linguistics.

Alex Kendall and Yarin Gal. 2017. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5574–5584.

Noo-Ri Kim and Jee-Hyong Lee. 2022. Propagation regularizer for semi-supervised learning with extremely scarce labeled samples. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 14381–14390. IEEE.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526.

Dong-Hyun Lee. 2013. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, page 2.

Changchun Li, Ximing Li, and Jihong Ouyang. 2021. Semi-supervised text classification with balanced deep representation distributions. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics, ACL 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 5044–5053. Association for Computational Linguistics.

Ming Li and Zhi-Hua Zhou. 2007. Improve computer-aided diagnosis with machine learning techniques using undiagnosed samples. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 37(6):1088–1098.

Jianhua Lin. 1991. Divergence measures based on the shannon entropy. *IEEE Trans. Inf. Theory*, 37(1):145–151.

Michael McCloskey and Neal J. Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. volume 24 of *Psychology of Learning and Motivation*, pages 109 – 165. Academic Press.

Richard Mccreadie, Cody Buntain, and Ian Soboroff. 2019. Trec incident streams: Finding actionable information on social media. In *Proceedings of the 16th International Conference on Information Systems for Crisis Response and Management (ISCRAM), 2019*.

Subhabrata Mukherjee and Ahmed Hassan Awadallah. 2020. Uncertainty-aware self-training for few-shot text classification. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Kamal Nigam, Andrew McCallum, Sebastian Thrun, and Tom M. Mitchell. 2000. Text classification from labeled and unlabeled documents using EM. *Mach. Learn.*, 39(2/3):103–134.

Gabriel Pereyra, George Tucker, Jan Chorowski, Lukasz Kaiser, and Geoffrey E. Hinton. 2017. Regularizing neural networks by penalizing confident output distributions. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of NAACL: Human Language Technologies*, pages 2227–2237.

Hieu Pham, Zihang Dai, Qizhe Xie, and Quoc V. Le. 2021. Meta pseudo labels. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 11557–11568. Computer Vision Foundation / IEEE.

Mamshad Nayeem Rizve, Kevin Duarte, Yogesh Singh Rawat, and Mubarak Shah. 2021. In defense of pseudo-labeling: An uncertainty-aware pseudo-label selection framework for semi-supervised learning. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Sebastian Ruder and Barbara Plank. 2018. Strong baselines for neural semi-supervised learning under domain shift. In *Proceedings of the 56th Annual Meeting of ACL (Volume 1: Long Papers)*, pages 1044–1054. Association for Computational Linguistics.

Kuniaki Saito, Yoshitaka Ushiku, and Tatsuya Harada. 2017. Asymmetric tri-training for unsupervised domain adaptation. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 2988–2997.

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

Vikas Sindhwani, Partha Niyogi, and Mikhail Belkin. 2005. A co-regularization approach to semi-supervised learning with multiple views. In *Proceedings of ICML workshop on learning with multiple views*, volume 2005, pages 74–79.

Anders Søgaard. 2010. Simple semi-supervised training of part-of-speech taggers. In *Proceedings of the ACL 2010 Conference Short Papers*, ACLShort '10, page 205–208, USA. Association for Computational Linguistics.

Tu Vu, Minh-Thang Luong, Quoc V. 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 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 5715–5731. Association for Computational Linguistics.

Davy Weissenbacher and Graciela Gonzalez-Hernandez, editors. 2019. *Proceedings of the Fourth Social Media Mining for Health Applications (#SMM4H) Workshop & Shared Task*. Association for Computational Linguistics, Florence, Italy.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Jiawei Wu, Lei Li, and William Yang Wang. 2018. Reinforced co-training. In *Proceedings of the 2018 NAACL*, pages 1252–1262, New Orleans, Louisiana.

Qizhe Xie, Zihang Dai, Eduard H. Hovy, Thang Luong, and Quoc Le. 2020a. Unsupervised data augmentation for consistency training. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. 2020b. Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10687–10698.

Liyan Xu, Xuchao Zhang, Xujiang Zhao, Haifeng Chen, Feng Chen, and Jinho D. Choi. 2021. Boosting cross-lingual transfer via self-learning with uncertainty estimation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 6716–6723. Association for Computational Linguistics.

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.

Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. 2018a. mixup: Beyond empirical risk minimization. In *6th ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015a. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 649–657.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015b. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 649–657.

Ying Zhang, Tao Xiang, Timothy M. Hospedales, and Huchuan Lu. 2018b. Deep mutual learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zhi-Hua Zhou and Ming Li. 2005. Tri-training: exploiting unlabeled data using three classifiers. *IEEE Transactions on Knowledge and Data Engineering*, 17(11):1529–1541.

Barret Zoph, Golnaz Ghiasi, Tsung-Yi Lin, Yin Cui, Hanxiao Liu, Ekin D Cubuk, and Quoc V Le. 2020. Rethinking pre-training and self-training. *arXiv preprint arXiv:2006.06882*.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
It is a section after the conclusion section

☑ A2. Did you discuss any potential risks of your work?
It is a section after the limitation seciton

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Left blank.

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

Appendices B and C

☑ B1. Did you cite the creators of artifacts you used?
Left blank.

☒ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Left blank.

☑ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Discussed this in the ethics section

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Not applicable. Left blank.

☑ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Appendix B

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Appendix B

C ☑ Did you run computational experiments?

Section 5

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Appendix A also in the last section after the conclusion section

---

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Appendix D

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Section 5.1

☐ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
No response.

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.