Semi-Supervised Text Classification via Self-Pretraining

Payam Karisani
Emory University
payam.karisani@emory.edu

Negin Karisani
Purdue University
nkarisan@purdue.edu

ABSTRACT

We present a neural semi-supervised learning model termed Self-Pretraining. Our model is inspired by the classic self-training algorithm. However, as opposed to self-training, Self-Pretraining is threshold-free, it can potentially update its belief about previously labeled documents, and can cope with the semantic drift problem. Self-Pretraining is iterative and consists of two classifiers. In each iteration, one classifier draws a random set of unlabeled documents and labels them. This set is used to initialize the second classifier, to be further trained by the set of labeled documents. The algorithm proceeds to the next iteration and the classifiers' roles are reversed. To improve the flow of information across the iterations and also to cope with the semantic drift problem, Self-Pretraining employs an iterative distillation process, transfers hypotheses across the iterations, utilizes a two-stage training model, uses an efficient learning rate schedule, and employs a pseudo-label transformation heuristic. We have evaluated our model in three publicly available social media datasets. Our experiments show that Self-Pretraining outperforms the existing state-of-the-art semi-supervised classifiers across multiple settings. Our code is available at https://github.com/p-karisani/self-pretraining.

CCS CONCEPTS

• **Information systems → Search results deduplication**; **Social networks**; **Document filtering**; **Information extraction**; **Clustering and classification**; *Nearest-neighbor search.*

KEYWORDS

classification, semi-supervised learning, social media mining

ACM Reference Format:
Payam Karisani and Negin Karisani. 2021. Semi-Supervised Text Classification via Self-Pretraining. In *Proceedings of the Fourteenth ACM International Conference on Web Search and Data Mining (WSDM '21), March 8–12, 2021, Virtual Event, Israel.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3437963.3441814

1 INTRODUCTION

Semi-supervised text classifiers have achieved remarkable success in the past few years due to the high capacity of neural networks in generalization. Even though modern classifiers usually rely on large training sets, the introduction of contextual word embeddings and language model pretraining [18, 41, 43] has tremendously reduced the need for manual data annotation. However, the state-of-the-art neural models are still prone to overfitting, particularly in the areas with sparse and specialized language models. These areas include, but are not limited to: legal domain [26], medical domain [34], and social media domain [3].

Depending on the task at hand, one solution to address this issue is to automatically construct a large–and perhaps noisy–dataset [20], however, this is not always feasible [59]. A more methodical approach is to employ the techniques that improve generalization. These techniques include exploiting neural word embeddings [29], data augmentation [6], and domain adaptation [2]. Exploiting unlabeled data [9, 58] is also a complementary approach. In this study, we add to the body of literature on semi-supervised learning by employing the properties of neural networks and proposing a novel way to utilize unlabeled data. We focus on one of the areas that reportedly suffers from the lack of enough training data, i.e., the social media mining. In addition to the lack of training data, the progress in this domain is further hindered by the short document lengths, informal language model, and typically ambiguous choice of vocabularies. These qualities make the social media tasks a suitable test bed for evaluating semi-supervised learning algorithms.

Our algorithm, termed Self-Pretraining, is inspired by the self-training paradigm [58]. Similar to self-training, our algorithm is iterative and in each iteration selects a set of unlabeled documents to label. However, as opposed to self-training, our algorithm is threshold-free. Thus, it does not rank the unlabeled documents based on their prediction confidences. This makes our algorithm particularly suitable for the neural network models due to their poorly calibrated outputs [35]. Additionally, our algorithm is able to cope with the semantic drift problem [12]. That is, it is resilient to the noise in the *pseudo-labels* as the number of iterations increases and the error rate of the underlying classifier rises. Furthermore, Self-Pretraining is able to potentially revise the labels of the previously labeled documents. To achieve these, our model employs an iterative distillation process, i.e., in each iteration, the information obtained in the previous iterations is distilled into the classifier. It transfers a hypothesis across iterations, and utilizes a two-stage learning model, where the set of pseudo-labels is used to initialize the classifier, and the set of labeled documents is used to finetune the classifier. Additionally, Self-Pretraining adapts a novel learning rate schedule to efficiently integrate the two sets of noisy and noise-free training examples. Finally, in order to further mitigate the impact of noisy pseudo-labels in every iteration, our model transforms the distribution of pseudo-labels such that it reflects the distribution of the labels in the previous iterations.

Our experiments in three publicly available Twitter datasets show that Self-Pretraining outperforms the state of the art in multiple settings where only a few hundred labeled documents are

WSDM '21, March 8–12, 2021, Virtual Event, Israel

ACM ISBN 978-1-4503-8297-7/21/03…$15.00
https://doi.org/10.1145/3437963.3441814

available. This is significant, considering that the underlying classifier of our algorithm and all the baseline models is BERT [18] which already uses the language model pretraining, and therefore, makes any improvement over the baselines very challenging. We also carry out a comprehensive set of experiments to better understand the qualities of Self-Pretraining. Particularly, we demonstrate the robustness of our model against the noise in the pseudo-labels. The contributions of our study are as follows: **1)** We propose a novel semi-supervised learning framework termed Self-Pretraining. Our model is based on the self-training paradigm, however, it is threshold-free, it can cope with the semantic drift problem, and can also revise the previously labeled documents. To our knowledge, Self-Pretraining is the first model that addresses these drawbacks in a unified framework. **2)** We propose a novel learning rate schedule to effectively integrate the optimization procedure with our two-stage semi-supervised learning process. **3)** In order to further mitigate the semantic drift problem, we model the class distribution of the pseudo-labels as a stochastic process across the bootstrapping iterations, and propose a novel approach to transform the class distributions. **4)** We carry out a comprehensive set of experiments across three publicly available Twitter datasets, and demonstrate that our model outperforms several state-of-the-art baselines in multiple settings.

Our research clearly pushes the state of the art in semi-supervised text classification. We believe the ideas presented in our paper can be applied to other domains, e.g., image classification. Future work may explore this direction. In the next section, we provide an overview of the related studies and highlight the qualities of Self-Pretraining.

2 RELATED WORK

Unlabeled data in semi-supervised learning. Unlabeled data can be exploited in multiple ways. It can be used as a meta-source of information [21], it can be used as a regularizer [56], or it can be used in a domain adaptation setting to correlate the source and target data [51]. A more recent interest in literature is *self-supervision*, where a self-contained task is defined such that no manual annotation is required. Instances of such tasks are language model pretraining [18, 41] in NLP, and contrastive learning in image processing [15, 46]. From a different perspective, self-supervision studies can be categorized into task-agnostic [10] and task-specific [22] approaches. This has given rise to the notion of "pretrain, then finetune" the model. We integrate this paradigm into the self-training algorithm.

Bootstrapping in semi-supervised learning. Self-training is the oldest approach to semi-supervised learning [14] dating back to 1965 [48]. This idea re-emerged in the seminal work of Yarowsky [58] for NLP tasks in 1995, and also once more in the computer vision community in 2013 as *pseudo-labeling* [33]. This algorithm is a wrapper that repeatedly uses a supervised algorithm as the underlying model. There are multiple assumptions under which self-training–and in general semi-supervised learning–is expected to perform well. For instance, the *smoothness assumption* that states if the two data points x_1 and x_2 are close, then their predictions y_1 and y_2 should be also close–this assumption has been the basis of algorithms such as MixUp [60] and MixMatch [8]. As we discuss in the next section, one unsatisfactory aspect of self-training is that it relies on the properties of the underlying predictive model, e.g., the model output distributions. There have been attempts to address this drawback. For instance, throttling [1] can be used to dampen the effect of noisy candidates, or in the context of transductive learning, the density of the unlabeled data points can be incorporated to mitigate this issue [49].

In the past few years, studies have explored the efficacy of the neural networks as the underlying predictive model in self-training. A neural network variant of co-training [9] is proposed in [42]. In [30], the authors propose a framework to integrate human knowledge with co-training. In [55], a reinforcement learning variant of co-training is proposed. In [44], a neural network variant of tri-training with disagreement [50] is presented, and it is shown that the combination is a surprisingly strong baseline in the domain adaptation setting. The authors in [13] propose to use percentile scores instead of the confidence scores to select the best pseudo-labels; and the authors in [39] employ Bayesian neural networks to select the most and the least confident pseudo-labels in every iteration. In [4], a new document sampling strategy for self-training is proposed. The model, in addition to the classifier confidence, employs the training epochs in which the unlabeled documents are approximately correctly labeled. In [5], the authors propose to integrate MixUp [60] with the oversampling of the labeled training examples. They show that self-training is indeed a very strong baseline comparing to the common regularization and data augmentation techniques. In comparison to these studies, Self-Pretraining is the first model that employs model distillation [25] along a hypothesis to transfer information across iterations, enabling it to potentially revise the pseudo-labels. It integrates the pretraining/finetuning paradigm with self-training, utilizes an efficient optimization procedure along a perturbation technique to mitigate the negative impact of noisy pseudo-labels.

Other closely related studies. In addition to the studies above, Self-Pretraining is also related to the studies on model distillation [25] and temporal ensembling [32]. Model distillation was proposed in [11, 25] to transfer the knowledge from one model to another model. In [16], the authors show that transferring the knowledge of a big network, trained by a self-supervised task, to a small network improves generalization. Their main contribution is to show that big models are trained easier, and therefore, can be used as a proxy to train small networks. Their model is not iterative, and does not explore the unlabeled data to extract new information. Born-again networks were proposed in [19], the authors show that simply distilling a neural network into itself improves performance. Their model is not a semi-supervised algorithm, and is not proposed to exploit unlabeled data. The authors in [57] show that the regular neural self-training algorithm can be improved by adding noise to the model. Similar to our work, they allow the pseudo-labels evolve over iterations. Beyond this step, they don't propose any modification to the self-training algorithm. Additionally, the efficacy of their model is not explored in the semi-supervised setting. A very close approach to this study is presented in [23], where the authors again show that adding noise to the inner representation of the model enhances the self-training performance. Temporal ensembling was proposed in [32]. The authors propose to maintain the per-sample prediction average of the unlabeled data across the epochs and constrain the prediction variance. Their model is not

based on self-training, has no strategy to separate labeled from unlabeled data, and becomes unwieldy when using large datasets. The authors in [52] resolve the high complexity of temporal ensembling by updating the weights of the model across the epochs, instead of storing the predictions.

3 SEMI-SUPERVISED LEARNING VIA SELF-PRETRAINING

We begin this section by providing an overview of Self-Pretraining and highlighting its differences from the self-training algorithm. Then we introduce a series of strategies to overcome the drawbacks of the vanilla Self-Pretraining[1].

In the self-training algorithm [58], a small set of labeled documents L and a large set of unlabeled documents U are available for training. The algorithm is iterative and in each iteration the predictive model M is trained on the current set L, and is used to probabilistically label the current set U. Given the hyper-parameter θ as the minimum confidence threshold, the most confidently labeled documents in U and their associated *pseudo-labels* are selected to be augmented with the set L. This procedure is repeated till a certain criterion is met. There are three drawbacks with this algorithm: 1) The semantic drift problem [12], where the increasingly negative impact of noisy pseudo-labels overshadows the benefit of incorporating unlabeled data. 2) Reliance on the model calibration. If the underlying classifier is unable to accurately model the class distributions, then, it will fail to properly rank the candidate documents, e.g., in the case of neural networks [35]. 3) Being unable to revise the pseudo-labels once they are assigned to the unlabeled documents and augmented with the set of labeled documents. Even though there exist techniques to address these challenges under certain conditions, e.g., throttling [1] for the poor model calibration or mutual exclusive bootstrapping [17] for the semantic drift, to our knowledge, Self-Pretraining is the first unified framework to address all three.

Our algorithm is iterative and utilizes two neural networks as the underlying classifiers. Algorithm 1 illustrates Self-Pretraining in its basic form. Initially, the set L is used to train the network M_1 (Line 2), then the parameters of M_1 are copied to the network M_2 (Line 5). In the next step, a set of unlabeled documents are randomly drawn from U (Line 7). This set is labeled by M_2 and used along the set L to retrain[2] M_1 (Line 8). The role of the two networks is reversed in the next iteration. In each iteration, the sample size is increased by k (Line 6), and the algorithm stops when the sample set covers the entire set U. Finally, the ensemble of M_1 and M_2 can be used to label the unseen documents–we used the mean of their class predictions.

Algorithm 1 has two advantages: 1) To select the pseudo-labels the class distribution is not taken into account, therefore, there is no constraint on the classifier capacity in ranking the unlabeled documents. Additionally, this prevents the model from repeatedly selecting a fixed set of unlabeled documents in every iteration–i.e., the set of highly confident pseudo-labels. 2) The information that is transferred across the iterations is in the form of a hypothesis rather than a set of fixed pseudo-labels. Therefore, the model belief about the pseudo-labels can evolve over time–the pseudo-labels are not augmented with the set of labeled documents. On the other hand, this algorithm has one substantial disadvantage, and that is the problem of semantic drift. In fact, randomly sampling from the set of unlabeled documents exacerbates this problem by introducing noisy labels and pushing the transferred hypothesis towards a suboptimal point. In the following, we exploit the neural network properties and introduce a series of strategies to cope with this problem and also to enhance the flow of information across the iterations.

Algorithm 1 Overview of Vanilla Self-Pretraining

```
1:  function SELF_PRETRAINING(L, U, k)
2:    M1 ← train_model(L)
3:    sample_size ← 0
4:    repeat
5:      M2 ← copy_model(M1)
6:      sample_size ← sample_size + k
7:      C ← random_sample(U, sample_size)
8:      M1 ← train_model({(C, M2(C)) ∪ L})
9:    until sample_size < |U|
10:   return M1, M2
```

3.1 Hypothesis Transfer and Iterative Distillation

Self-Pretraining transfers a hypothesis–a learned function–from one iteration to the next iteration. In each iteration, this hypothesis is used to form a new one by creating a set of pseudo-labels and augmenting them with the set of labeled documents. Even though the ultimate criterion is maximizing the model utility, the short term goal in each iteration is not necessarily making accurate predictions but to carefully transfer the knowledge from one model to another. These two processes are not necessarily in accordance with each other, since the former may rely on the learner outcome and the latter may rely on the learning procedure itself. Thus, the classifier labels, even though informative, are not expressive enough to transfer the entire knowledge from one iteration to the next one.

The authors in [11, 25] propose an algorithm called model distillation to transfer the knowledge from a large model (called *teacher*) to a small model (called *student*). Model distillation is based on the argument that the class distribution carries a significant amount of information regarding the classifier decision boundary. For instance, given a document d that is labeled positive, it is nontrivial information to know that if the class prediction was 95% positive or 65% positive. The authors in [25] use model distillation to transfer knowledge from one network to another by modifying the softmax layer as follows:

$$a_i = \frac{\exp^{\frac{z_i}{T}}}{\sum_j \exp^{\frac{z_j}{T}}} \tag{1}$$

where z_i is the last layer *i-th* logit, j is the number of classes, and a_i is the class prediction. The hyper-parameter T is called temperature and is introduced to smooth the class predictions. A higher temperature results in a higher entropy in predictions. This is particularly desirable, since neural networks are known to have a low entropy in their predictions [35].

[1] We focus on the binary classification problems.

[2] Note that by definition, the neural self-training requires reinitialization and retraining in every iteration [44], thus our algorithm is comparable to other self-training models in terms of runtime.

Given the argument above, we employ model distillation in Self-Pretraining, and effectively distill the previous iterations into the student network M_1. Thus, in each iteration, instead of using the teacher–M_2–hard predictions on unlabeled documents, we use the soft predictions along the set L to train the student network–Algorithm 1, Line 8.

3.2 Two-Stage Semi-Supervised Learning

As we mentioned earlier, self-training suffers from the semantic drift problem. This problem occurs when the errors primarily caused by the pseudo-labels accumulate across iterations and ultimately distort the classifier boundary. Even though the minimum confidence threshold θ can potentially prevent spurious pseudo-labels from entering the training set, as the set L grows in size the probability of mislabeling documents increases correspondingly. This problem is even severer in our model, since it is threshold-free. One naive solution is to assign a lower weight to the pseudo-labels, however, we observed in our experiments that this approach is not effective enough to resolve the underlying problem.

To mitigate this problem, one solution is to process the set of pseudo-labels and decouple the information that contradicts the information stored in the set L. Erasing this section of the pseudo-labels can lower the error rate and subsequently improve the hypothesis in the current iteration. To accomplish this, we exploit the catastrophic forgetting phenomenon in the neural networks [31, 36]. Catastrophic forgetting occurs in the continual learning setting where a network is trained on a series of tasks. Each training procedure updates the parameters of the model to meet the requirements of the objective function, and the updates in the current task may contradict and erase the information related to the previous tasks. This effect is typically undesirable, however, in the context of Self-Pretraining, we use this mechanism as a proxy to build a hierarchy of information in the network. Therefore, we make a small modification in Algorithm 1. Instead of aggregating the set of pseudo-labels with the set of labeled documents–Line 8–we first use the set of pseudo-labels to initialize–train–the current network M_1, and then further train it using the set of labeled documents.

Decomposing the training procedure into two stages introduces a new challenge, and that is the possibility of completely updating the network parameters in order to learn the regularities in the set of labeled documents. To avoid this, we propose to use the following objective function while training the model M_1 using the set L:

$$\begin{aligned}\mathcal{L}=(1-\lambda)(-\sum_{i=1}^{N}[y_i\log a_i+(1-y_i)\log(1-a_i)])+\\ \lambda(-\sum_{i=1}^{N}[q_i\log a'_i+(1-q_i)\log(1-a'_i)])\end{aligned} \tag{2}$$

where N is the number of the documents in the set L, y_i is the true label of the document d_i, a_i is the class prediction of M_1 for d_i, a'_i is the class prediction of M_1 for d_i with a high temperature as described in Section 3.1, and q_i is the class prediction of M_2 for d_i with the same temperature as that of M_1. λ is a hyper-parameter to govern the relative weight of the two terms ($0 \leq \lambda \leq 1$). Since the gradients of the second term in Equation 2 scale by $\frac{1}{T^2}$, in order to balance the impact of the two terms in back-propagation, we multiply these gradients by T^2–see Equation 1.

The first term in Equation 2 is the cross entropy between the ground truth labels and the class probabilities of M_1. The second term is the cross entropy between the class probabilities of M_2 and M_1. This objective function is an effort to keep a balance between the information that is transferred from the previous iterations and the information that is extracted from the set of labeled documents L.

In Section 5 we demonstrate that the ideas proposed in this section greatly mounts the resistance of Self-Pretraining to the noise in the pseudo-labels. These ideas are related to two categories of studies: 1) The studies on pretraining neural networks [24, 27]. 2) The studies on curriculum learning [7]. Researchers [24, 27] in both NLP and the vision community have shown that *pretraining* a neural network with out-of-domain data and then *finetuning* it with the target data can significantly contribute to the performance. These two steps are analogous to the two stages that we described in this section. Additionally, our work is also closely related to the idea of curriculum learning [7], where it is shown that a learner can leverage the order of the training examples to learn more efficiently. Even though Self-Pretraining employs this mechanism, the criterion to determine the order of the training examples is not based on the properties of the data points but is based on the source of the labels.

3.3 Right Trapezoidal Learning Rates

In the previous section, we employed an approach to mitigate the semantic drift problem by exploiting the catastrophic forgetting phenomenon. This two-stage strategy creates a suitable opportunity for enhancing the optimization process. Since the pseudo-labels are potentially noisy, we propose to use this set to explore the hypothesis space and detect the region that contains a better local-optima. Thereafter, the set of labeled documents, which are noise-free, can be used to detect the target local-optima.

Given the argument above, we propose to use a *right trapezoidal* learning rate–illustrated in Figure 1–as follows:

$$\eta_t=\begin{cases} R & batch_t \subset C \\ R - R\,\frac{t-b_C}{b_L} & batch_t \subset L \end{cases}$$

where t denotes the current time step, and $batch_t$ is the current batch of documents being processed. η_t is the current learning rate, R is the initial learning rate, C is the set of pseudo-labels, L is the set of labeled documents, b_C is the number of pseudo-label batches, and b_L is the number of labeled batches.

Our proposed learning rate is composed of two phases: 1) A fixed learning rate–the dashed line in Figure 1–where the pseudo-labels are used to train the model M_1–see Algorithm 1. In this stage, the network parameters can freely update, and therefore, the learner can essentially explore the hypothesis space. 2) A gradually decreasing learning rate–the solid slanted line in Figure 1–where the labeled documents are used to further train the network. In this stage, the optimizer settles down, therefore, we use the noise-free labeled documents, since even a small perturbation in the data may cause a significant loss. Having a two-phase learning rate also organically integrates with our two-stage semi-supervised learning procedure. Since the gradual reduction in the learning rate, prevents the objective of the second task from completely erasing the knowledge transferred from the previous iterations.

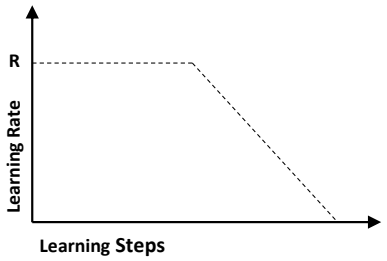

Figure 1: The Self-Pretraining learning rate schedule. The dashed horizontal line is the learning rate of the network during the training by the pseudo-labels, and the slanted line is the learning rate during the training by the labeled documents.

3.4 Inertial Class Distributions

Semi-supervised learning models rely on unlabeled data as their primary source of information. While these methods have obtained promising results, they are inherently prone to overfitting on the irregularities in the unlabeled data. Introducing an inductive bias [38] into the semi-supervised learning algorithms is a common approach to increase their robustness. For instance metric regularization [47] or temporal ensembling [32] are a few examples. While these techniques can be integrated into Self-Pretraining, in this section, we opt to explore a new direction.

We hypothesize that the class probability distribution of the randomly selected set of unlabeled documents–Algorithm 1, Line 8–should evolve slowly and avoid abrupt transitions across iterations. This is a harsh assumption, since this probability distribution also depends on the drawn samples. However, we argue that an abrupt change in this distribution can be the sign of an influx of noisy pseudo-labels in the previous iterations. Thus, we aim to prohibit such changes. To achieve this, we assume the distribution of the class probabilities is a random process dynamically evolving across the iterations, and the class probability distribution of the selected unlabeled documents in every iteration–$M_2(C)$ in Algorithm 1–is a sample from the underlying random variables.

For simplicity, we assume the process consists of only a family of two Gaussian random variables S^+ and S^-, where S^+ is the state of the positive pseudo-labels, and S^- is the state of the negative pseudo-labels. The sample mean and variance of S^+ in the iteration t (i.e., S^+_t) are given by:

$$\mu_t = \frac{\sum_{i=1}^{n} p_i^t}{n}$$
$$\sigma_t^2 = \frac{\sum_{i=1}^{n} (p_i^t - \mu_t)^2}{n}$$

where n is the number of positive pseudo-labels in the iteration t, and p_i^t is the probability of the *i-th* positive pseudo-label belonging to the positive class–it is clear that $0.5 \leq p_i^t$, because the sample is positive. Correspondingly, the sample mean and variance of S^- in the iteration t (i.e., S^-_t) are given by:

$$\gamma_t = \frac{\sum_{i=1}^{m} q_i^t}{m}$$
$$\varphi_t^2 = \frac{\sum_{i=1}^{m} (q_i^t - \gamma_t)^2}{m}$$

where m is the number of negative pseudo-labels in the iteration t, and q_i^t is the probability of the *i-th* negative pseudo-label belonging to the negative class–note that $0.5 \leq q_i^t$ and also note that for every pseudo-label $p_i^t + q_i^t = 1$.

In the iteration $t+1$, the sample distributions of the random variables S^+ and S^- proceed to $S^+_{t+1} \sim \mathcal{N}(\mu_{t+1}, \sigma^2_{t+1})$ and $S^-_{t+1} \sim \mathcal{N}(\gamma_{t+1}, \varphi^2_{t+1})$. These updates can be due to the randomness in the model initialization, the randomness in the selected set of unlabeled documents in the iteration t, or partially due to the noisy pseudo-labels introduced in the iteration t. More specifically, the misclassifications of the model M_2 in the iteration t–see Algorithm 1–which were subsequently used to pretrain the model M_1, and ultimately distorted the class distribution of the set of pseudo-labels in the iteration $t+1$. To dampen the impact of this noise, we define two Gaussian distributions $\hat{S}^+_{t+1}$ and $\hat{S}^-_{t+1}$ as the linear combination of the class distributions in the iterations t and $t+1$, and project[3] the pseudo-labels in S^+_{t+1} into $\hat{S}^+_{t+1}$, and the pseudo-labels in S^-_{t+1} into $\hat{S}^-_{t+1}$. Thus:

$$\begin{aligned} \hat{S}^+_{t+1} &= \alpha\ S^+_t + (1-\alpha)\ S^+_{t+1} \\ \hat{S}^-_{t+1} &= \alpha\ S^-_t + (1-\alpha)\ S^-_{t+1} \end{aligned} \tag{3}$$

where α is a hyper-parameter to govern the rate at which the probability distributions can evolve in every iteration. The new distributions $\hat{S}^+_{t+1}$ and $\hat{S}^-_{t+1}$ are defined between the class distributions in the iteration t and $t+1$. The hyper-parameter α determines the degree at which the pseudo-labels in the iteration $t+1$ are perturbed to resemble the pseudo-labels in the iteration t. By employing this mechanism, the sudden abrupt changes in the distribution of pseudo-labels are avoided. We perform this step after we generate the pseudo-labels using the model M_2, and before using this set to pretrain the model M_1–Algorithm 1, Line 8.

In Section 5, we show that Self-Pretraining algorithm, along the techniques that we introduced in the sections 3.1, 3.2, 3.3, and 3.4 achieves the state-of-the-art results in multiple settings. In the next section, we describe our datasets, baselines, and training setup.

4 EXPERIMENTAL SETUP

We begin this section by describing the datasets that we used, then we provide a brief overview of the baseline models, and finally review the detail of the experiments.

4.1 Datasets

We evaluate Self-Pretraining on three Twitter text classification tasks[4]: 1) Adverse Drug Reaction monitoring (ADR). In this task, the goal is to detect the tweets that report an adverse drug effect. We used the dataset introduced in [53] prepared for the ACL 2019 SMM4H Shared Task. 2) Crisis Report Detection (CRD). In this task, the goal is to detect the tweets that mention an event related to natural disasters. We used the dataset introduced in [2] about the 2015 Nepal earthquake. 3) Product Consumption Pattern identification (PCP). In this task, the goal is to identify the tweets that report the usage of a product. We used the dataset introduced in [28], which is about receiving an influenza vaccine.

[3] No projection is performed in the first iteration.
[4] Please refer to the cited articles for the analysis and discussion on the difficulties of these tasks, we skip this subject.

	Training			Test		
Dataset	Tweets	Neg	Pos	Tweets	Neg	Pos
ADR	20624	91%	9%	4992	92%	8%
Earthquake	8166	53%	47%	3502	53%	47%
Product	4503	69%	31%	2114	78%	22%

Table 1: Summary of ADR , Earthquake , and Product datasets.

The ADR and Earthquake datasets are released with pre-specified training and test sets. In Product dataset we used the tweets published in 2013 and 2014 for the training set, and the tweets published in 2015 and 2016 for the test set. Table 1 summarizes the datasets. We see that Earthquake dataset is balanced and ADR dataset is highly imbalanced. The Earthquake dataset is released along a set of unlabeled tweets. For the other two datasets, we used the Twitter API and crawled 10,000 related tweets for each one to be used as the unlabeled sets (the set U in Algorithm 1). For ADR dataset we used the drug names to collect the unlabeled set and for Product dataset we used the query "flu AND (shot OR vaccine)" to collect the set.

4.2 Baselines

We compare our model with six baselines.

Baseline. The setting for evaluating semi-supervised learning models should be realistic. Pretrained contextual language models are the primary ingredient of the state-of-the-art text classifiers. Thus, we used BERT [18] as the naive baseline, and also as the underlying classifier for all the other baselines. Note that this makes any improvement over the base classifier very challenging, since the improvement should be additive. We train this model on the set of labeled documents, and evaluate on the test set. We used the published pretrained *base* variant, followed by one fully connected layer and one softmax layer. We used the Pytorch implementation [54] of BERT; the settings are identical to the suggestions in [18].

Self-training. We included the regular self-training algorithm [58], where in each iteration the top pseudo-labels, subject to a minimum threshold confidence, are selected and added to the labeled set. We used one instance of *Baseline* in this algorithm.

Tri-training+. We included a variant of tri-training algorithm called tri-training with disagreement [50]. In [44], the authors show that this model is a very strong baseline for semi-supervised learning. We used three instances of *Baseline* in this algorithm.

Mutual-learning. We included the model introduced in [61]. This model is an ensemble, and is based on the idea that increasing the entropy of the class predictions improves generalization [40]. We used two instances of *Baseline* in this model–in the parallel setting.

Spaced-rep. We included the model introduced in [4]. This model employs a queuing technique along a validation set to select the unlabeled documents that are easy and also informative for the task. We used our own implementation of this model.

Co-Decomp. We included the framework introduced in [30]. This model uses domain knowledge to decompose the task into a set of subtasks to be solved in a multi-view setting. We used the keyword level representations and sentence level representations as the two views. We used two instances of *Baseline* in this algorithm.

Self-Pretraining. The model that we introduced in Section 3. We used two instances of *Baseline* as M_1 and M_2.

4.3 Experimental Details

To evaluate the models in the semi-supervised setting, we sampled a small subset of the training sets[5] and did not use the rest of the tweets. Note that the remaining set was not used as the unlabeled data either–see Section 4.1 for the description of the unlabeled sets. To sample the data, we used a stratified random sampling to preserve the ratio of the positive to the negative documents. We also ensured that the initial labeled set is identical for all the models. We repeated all the experiments 3 times with different random seeds. We will report the average across the runs. All the baseline models use throttling [1] with confidence thresholding ($\theta = 0.9$). We also linearly increased the size of the sample set [45], however, did not add more than 10% of the current training set in each iteration.

In our experiments we observed that the performances of *self-training* and *Co-Decomp* degrade if we use the entire set of unlabeled data–due to the semantic drift problem. Thus, we assumed the number of the iterations in these algorithms is a hyper-parameter and used 20% of the labeled set as the validation set to find the best value. *Tri-training+* has an internal stopping criterion. *Mutual-learning* uses the unlabeled data as a regularizer. *Spaced-rep* requires a validation set for the stopping criterion and also for the candidate selection. Thus, in this model we used 20% of the labeled set as the validation set. We also set the number of queues to 6, the rest of the settings are identical to what is used in [4].

Since we are experimenting in the semi-supervised setting, we did not do full hyper-parameter tuning. We used the training set in Product dataset and searched for the optimal values of λ in Equation 2 and α in Equation 3. Their best values are 0.3 and 0.1 respectively. We set the step size k in Algorithm 1 to 2,000 and the temperature T in Equation 1 to 3. In our two-stage training procedure the goal of the first step is the model initialization, thus we trained the network for only 1 epoch. In the rest of the cases, including in our model and the baselines we trained the models for 3 epochs. The only exception is *Space-rep*, which requires a certain number of training epochs with early stopping. To train BERT in all of the cases we used a setting identical to that of the reference [18]–we set the batch-size to 32. Following the argument in [37], we used F1 in the positive class to tune the models. In the next section, we report average F1, Precision, and Recall of the models across the runs.

5 RESULTS AND DISCUSSION

We begin this section by reporting the main results. Then we present a series of experiments that we carried out to better understand the properties of Self-Pretraining.

5.1 Main Results

Table 2 reports the performance of Self-Pretraining in comparison to the baselines under two sampling quantities–i.e, 300 and 500 initial random tweets–in the three datasets. We see that in all of the cases Self-Pretraining is either the top model or on a par with the top model. The difference in ADR dataset is substantial, however, in Earthquake dataset the difference is very small. ADR is an imbalanced dataset. Our case by case inspections also showed that the positive tweets in this dataset are very diverse, which makes

[5]Using the entire set of labeled tweets turns the classification task into a supervised problem, which is not the subject of our study.

# Tweets	Model	ADR dataset			Earthquake dataset			Product dataset		
		F1	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall
300	*Baseline*	0.238	0.237	0.342	0.715	0.692	0.749	0.728	0.696	0.770
	Self-training	0.303	0.269	0.350	0.728	0.697	0.762	0.731	0.675	0.798
	Tri-training+	0.306	0.236	**0.448**	0.735	0.680	0.799	0.734	0.659	**0.828**
	Mutual-learning	0.024	**0.707**	0.012	**0.743**	0.685	**0.814**	0.753	**0.778**	0.730
	Spaced-rep	0.258	0.248	0.277	0.721	0.650	0.811	0.727	0.701	0.760
	Co-Decomp	0.310	0.288	0.356	0.728	**0.722**	0.735	0.754	0.756	0.758
	Self-Pretraining	**0.397**	0.370	0.440	0.737	0.704	0.772	**0.766**	0.757	0.777
500	*Baseline*	0.312	0.253	0.411	0.746	0.735	0.760	0.740	0.704	0.782
	Self-training	0.335	0.300	0.387	0.737	**0.765**	0.714	0.741	0.739	0.745
	Tri-training+	0.365	0.298	0.480	0.747	0.707	**0.793**	0.758	0.697	**0.833**
	Mutual-learning	0.108	**0.638**	0.059	0.751	0.730	0.773	0.767	**0.811**	0.728
	Spaced-rep	0.295	0.274	0.417	0.728	0.694	0.775	0.737	0.693	0.788
	Co-Decomp	0.345	0.313	0.388	0.749	0.746	0.752	0.766	0.771	0.764
	Self-Pretraining	**0.420**	0.376	**0.483**	**0.752**	0.718	0.789	**0.787**	0.784	0.792

Table 2: F1, precision, and recall of Self-Pretraining in ADR , Earthquake , and Product datasets compared to the baseline models. The models were trained on 300 and 500 labeled user postings.

the models very susceptible to the number of training examples. We also see that *Mutual-learning* completely fails in this dataset. Our experiments showed that this is due the the skewed class distributions in this dataset[6]. Surprisingly, we see that *Spaced-rep* is performing poorly in the experiments, even though this model was evaluated on social media tasks before [4]. We believe the reason is as follows: This model relies on the number of training epochs to construct its internal data structure for ranking the candidate tweets. When the underlying classifier is a pretrained language model, e.g., bert, increasing the number of epochs may result in overfitting and therefore, contradicts the purpose. On the other hand, early stopping also prevents the model from separating the informative from uninformative tweets.

5.2 Empirical Analysis

We begin this section by reporting the effect of the step size k on Self-Pretraining –see Algorithm 1. Table 3 reports F1, precision, and recall of Self-Pretraining at varying step sizes in the test set of ADR dataset. Since this datset is the largest one, we report all of the experiments in this dataset. We see that the performance improves up to the step size of 3000 unlabeled tweets per iteration. We still do not have a concrete explanation to justify this trend, since it is natural to expect the smaller step sizes yield better results. One reason may be that if the set of pseudo-labels is small, the network can perfectly learn the noise in the set during the pretraining. In Section 3.2 we argued that the two-stage training can cope with the semantic drift problem. To support this argument, we report the performance of the middle classifiers M_1 at the end of every iteration. Figure 2 reports the performances during the training for varying step sizes. We see that for none of the step sizes the performance drops as the number of unlabeled tweets grows–the typical symptom of semantic drift.

[6]We built two imbalanced datasets by subsampling from Earthquake and Product datasets, this model also failed in these cases.

k	F1	Precision	Recall
1000	0.395	0.306	0.565
2000	0.420	0.376	0.483
3000	0.428	0.386	0.485
4000	0.413	0.347	0.537

Table 3: Results of Self-Pretraining with different values of k–the number of randomly selected pseudo-labels–in the test set of ADR dataset. The models began with 500 labeled user postings.

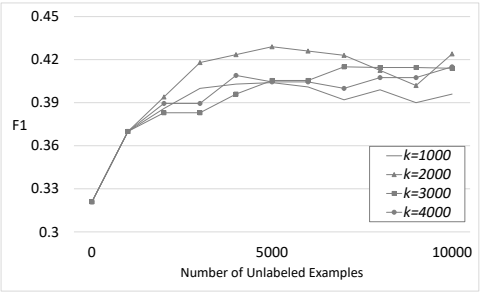

Figure 2: F1 of the resulting classifier in every iteration of Self-Pretraining with different values of k–the number of randomly selected pseudo-labels. The middle values are interpolated. The results are in the test set of ADR dataset.

Self-Pretraining relies on iterative distillation–Section 3.1–to transfer knowledge from one iteration to the next one. Model distillation leverages the temperature T in the softmax layer, see Equation 1. It is informative to find the degree at which this hyper-parameter can affect the learning performance. Table 4 reports the model performance at varying values of the hyper-parameter T. We see that the performance peaks at $T = 5$. In section 3.2 we proposed an objective function and argued that the second term of the function prevents the hard labels of the training set from erasing the information transferred from the previous iteration. To demonstrate the impact of the second term, in Table 5 we report the model performance at varying values of the hyper-parameter λ–the weight of the second term. We see that the performance almost

T	F1	Precision	Recall
2	0.422	0.361	0.514
3	0.420	0.376	0.483
4	0.421	0.356	0.517
5	0.433	0.382	0.506
6	0.422	0.370	0.491

Table 4: Results of Self-Pretraining in the test set of ADR dataset at varying values of the temperature (T) for iterative distillation.

λ	F1	Precision	Recall
0.1	0.425	0.357	0.529
0.2	0.428	0.355	0.541
0.3	0.420	0.376	0.483
0.4	0.438	0.377	0.531
0.5	0.421	0.350	0.534

Table 5: Results of Self-Pretraining in the test set of ADR dataset at varying values of the hyper-parameter (λ) for our two-stage learning–see Equation 2.

α	F1	Precision	Recall
0.1	0.420	0.376	0.483
0.2	0.422	0.353	0.530
0.3	0.413	0.345	0.518
0.4	0.424	0.355	0.532
0.5	0.429	0.363	0.527

Table 6: Results of Self-Pretraining in the test set of ADR dataset at varying values of the hyper-parameter (α) for the inertial transformation of the pseudo-labels–see Equation 3.

gradually improves as we increase λ and peaks at $\lambda = 0.4$. This is primarily due to the improvement in precision.

In Section 3.4 we proposed to transform the class probability distribution in the iteration $t + 1$ into a new distribution that resembles the distribution in the iteration t. We argued that this transformation can help to mitigate the semantic drift problem via constraining the degree at which the pseudo-labels can evolve in every iteration, therefore, can potentially limit the negative impact of noisy pseudo-labels. In Table 6 we report the model performance at varying values of the hyper-parameter α in Equation 3. This hyper-parameter governs the degree of the transformation. We see that the performance noticeably improves as we increase the value of α. Finally, we report an ablation study in Table 7. In the previous experiments we showed that a better performance in ADR dataset is achievable by a dataset specific hyper-parameter tuning. Nonetheless, we still expect that, with the current hyper-parameters in ADR, the ablation study can reveal the relative importance of the Self-Pretraining modules in general. In this experiment, we replaced the two-stage training model (Section 3.2) with the simple data augmentation of the labeled and pseudo-labels. Additionally, we replaced our right trapezoidal learning rate (Section 3.3) with the default slanted learning rate [18]. We replaced our iterative distillation process (Section 3.1) with simply using the hard labels in every iteration. Finally, we deactivated our pseudo-label transformation step (Section 3.4). We see that the two-stage training model and the inertial transform have the highest and the lowest contributions.

In summary, we showed that Self-Pretraining is the state-of-the-art in multiple settings. The authors in [4] show that semi-supervised models–although under domain shift–typically fail when they are evaluated on a different task from what they are initially proposed for. Thus, they conclude that these models should be evaluated in at least two datasets. In this study we evaluated Self-Pretraining in three Twitter datasets. We selected strong baselines, i.e., Tri-training with disagreement [50], Mutual Learning [61], Spaced Repetition [4], and Co-Decomp [30], and showed that some of them fail under certain cases. As opposed to these models, we demonstrated that Self-Pretraining is either the best model or on a par with the best model in every setting. We also reported an extensive set of experiments that we carried out to reveal the qualities of Self-Pretraining. These experiments empirically supported the claims that we made throughout.

Deactivated Step	F1	Precision	Recall
two-stage learning	0.339	0.373	0.333
trapezoidal lr	0.360	0.235	0.770
iterative distillation	0.389	0.320	0.495
inertial transform	0.420	0.365	0.497

Table 7: Results of Self-Pretraining in the test set of ADR dataset after deactivating the distillation (Section 3.1), the two-stage learning (Section 3.2), the trapezoidal learning rate (Section 3.3), and the inertial transformation (Section 3.4).

Our study is not flawless. To avoid imposing any constrain on the underlying classifier, we proposed to randomly draw the unlabeled documents–Algorithm 1, Line 7. However, if one can guarantee certain classifier properties, then perhaps a sophisticated selection policy will be more effective. The application of our framework in other modalities, e.g., image classification, is also an unexplored topic. Future work may investigate these directions.

6 CONCLUSIONS

In this study, we proposed a semi-supervised learning model called Self-Pretraining . Our model is inspired by the traditional self-training algorithm. Self-Pretraining employs the properties of neural networks to cope with the inherent problems of self-training. Particularly, it employs an iterative distillation procedure to transfer information across the iterations. It also utilizes a two-stage training model to mitigate the semantic drift problem. Additionally, Self-Pretraining uses an efficient learning rate schedule and a pseudo-label transformation heuristic. We evaluated our model in three publicly available Twitter datasets, and compared with six baselines, including pretrained BERT. The experiments show that our model consistently outperforms the existing baselines.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their insightful feedback.

REFERENCES

[1] Steven Abney. 2007. *Semisupervised Learning for Computational Linguistics* (1st ed.). Chapman & Hall/CRC.
[2] Firoj Alam, Shafiq Joty, and Muhammad Imran. 2018. Domain Adaptation with Adversarial Training and Graph Embeddings. In *Proceedings of the 56th ACL*. Association for Computational Linguistics, Melbourne, Australia, 1077–1087.
[3] Thayer Alshaabi, David R Dewhurst, and et al. 2020. The growing echo chamber of social media: Measuring temporal and social contagion dynamics for over 150 languages on Twitter for 2009–2020. *arXiv preprint arXiv:2003.03667* (2020).
[4] Hadi Amiri. 2019. Neural Self-Training through Spaced Repetition. In *Proceedings of the 2019 Conference of NAACL*. Minneapolis, Minnesota, 21–31.

[5] Eric Arazo, Diego Ortego, Paul Albert, and et al. 2020. Pseudo-Labeling and Confirmation Bias in Deep Semi-Supervised Learning. In *2020 International Joint Conference on Neural Networks, IJCNN, July 19-24, 2020*. IEEE, 1–8.

[6] David Bamman and Noah A. Smith. 2015. Contextualized Sarcasm Detection on Twitter. In *Proceedings of the Ninth ICWSM*. 574–577.

[7] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum Learning. In *Proceedings of the 26th ICML* (Montreal, Quebec, Canada) *(ICML '09)*. Association for Computing Machinery, New York, NY, USA, 41–48.

[8] David Berthelot, Nicholas Carlini, Ian J. Goodfellow, Nicolas Papernot, Avital Oliver, and Colin Raffel. 2019. MixMatch: A Holistic Approach to Semi-Supervised Learning. In *NeurIPS 2019, 8-14 Vancouver, BC, Canada*. 5050–5060.

[9] Avrim Blum and Tom M. Mitchell. 1998. Combining Labeled and Unlabeled Data with Co-Training. In *Proceedings of the Eleventh COLT, 1998, Madison, Wisconsin, USA, July 24-26, 1998*. 92–100.

[10] Tom B Brown, Benjamin Mann, and et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165* (2020).

[11] Cristian Buciluundefined, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model Compression. In *Proceedings of the 12th ACM SIGKDD* (Philadelphia, PA, USA) *(KDD '06)*. 535–541.

[12] 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*. 1306–1313.

[13] Paola Cascante-Bonilla, Fuwen Tan, Yanjun Qi, and Vicente Ordonez. 2020. Curriculum Labeling: Self-paced Pseudo-Labeling for Semi-Supervised Learning. *arXiv preprint arXiv:2001.06001* (2020).

[14] Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien (Eds.). 2006. *Semi-Supervised Learning*. The MIT Press.

[15] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709* (2020).

[16] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey Hinton. 2020. Big self-supervised models are strong semi-supervised learners. *arXiv preprint arXiv:2006.10029* (2020).

[17] 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*, Vol. 6. Bali, 172–180.

[18] 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*. 4171–4186.

[19] Tommaso Furlanello, Zachary Chase Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. 2018. Born-Again Neural Networks. In *Proceedings of the 35th ICML, Stockholm, Sweden, July 10-15, 2018*, Vol. 80. 1602–1611.

[20] Roberto Gonzalez-Ibaez, Smaranda Muresan, and Nina Wacholder. 2011. Identifying Sarcasm in Twitter: A Closer Look. In *Proceedings of the 49th ACL* (Portland, Oregon) *(HLT '11)*. 581–586.

[21] Suchin Gururangan, Tam Dang, Dallas Card, and Noah A. Smith. 2019. Variational Pretraining for Semi-supervised Text Classification. In *Proceedings of the 57th ACL*. Florence, Italy, 5880–5894.

[22] 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*.

[23] 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, April 26-30, 2020*. OpenReview.net.

[24] Dan Hendrycks, Kimin Lee, and Mantas Mazeika. 2019. Using Pre-Training Can Improve Model Robustness and Uncertainty. In *Proceedings of the 36th ICML, California, USA*, Vol. 97. 2712–2721.

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

[26] Nils Holzenberger, Andrew Blair-Stanek, and Benjamin Van Durme. 2020. A Dataset for Statutory Reasoning in Tax Law Entailment and Question Answering. *arXiv preprint arXiv:2005.05257* (2020).

[27] Jeremy Howard and Sebastian Ruder. 2018. Universal Language Model Fine-tuning for Text Classification. In *Proceedings of the 56th ACL*. 328–339.

[28] 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*.

[29] Payam Karisani and Eugene Agichtein. 2018. Did You Really Just Have a Heart Attack? Towards Robust Detection of Personal Health Mentions in Social Media. In *Proceedings of the 2018 World Wide Web Conference* (Lyon, France). 137–146.

[30] 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* (Taipei, Taiwan). 2411–2420.

[31] James Kirkpatrick, Razvan Pascanu, and et al. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences* 114, 13 (2017), 3521–3526.

[32] Samuli Laine and Timo Aila. 2017. Temporal Ensembling for Semi-Supervised Learning. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

[33] 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*, Vol. 3.

[34] Jinhyuk Lee, Wonjin Yoon, and et al. 2019. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics* 36, 4 (09 2019), 1234–1240.

[35] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. 2018. A Simple Unified Framework for Detecting Out-of-Distribution Samples and Adversarial Attacks. In *Advances in Neural Information Processing Systems 31*. 7167–7177.

[36] Michael McCloskey and Neal J. Cohen. 1989. Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem. Psychology of Learning and Motivation, Vol. 24. Academic Press, 109 – 165.

[37] Richard Mccreadie, Cody Buntain, and Ian Soboroff. 2019. TREC Incident Streams: Finding Actionable Information on Social Media. In *Proceedings of the 16th ISCRAM, 2019*.

[38] Tom M Mitchell et al. 1997. Machine learning. 1997. *Burr Ridge, IL: McGraw Hill* 45, 37 (1997), 870–877.

[39] Subhabrata Mukherjee and Ahmed Hassan Awadallah. 2020. Uncertainty-aware Self-training for Text Classification with Few Labels. arXiv:2006.15315 [cs.CL]

[40] Gabriel Pereyra, George Tucker, Jan Chorowski, Lukasz Kaiser, and Geoffrey E. Hinton. 2017. Regularizing Neural Networks by Penalizing Confident Output Distributions. In *5th ICLR 2017, Toulon, France, April 24-26, 2017*.

[41] 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 NAACL*. New Orleans, Louisiana, 2227–2237.

[42] Siyuan Qiao, Wei Shen, Zhishuai Zhang, Bo Wang, and Alan Yuille. 2018. Deep Co-Training for Semi-Supervised Image Recognition. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

[43] Colin Raffel and et al. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683* (2019).

[44] Sebastian Ruder and Barbara Plank. 2018. Strong Baselines for Neural Semi-Supervised Learning under Domain Shift. In *Proceedings of the 56th ACL* (Melbourne, Australia). Association for Computational Linguistics, 1044–1054.

[45] Kuniaki Saito, Yoshitaka Ushiku, and Tatsuya Harada. 2017. Asymmetric Tri-Training for Unsupervised Domain Adaptation. In *Proceedings of the 34th ICML* (Sydney, NSW, Australia) *(ICML'17)*. 2988–2997.

[46] M. Sajjadi, M. Javanmardi, and T. Tasdizen. 2016. Mutual exclusivity loss for semi-supervised deep learning. In *2016 IEEE (ICIP)*. 1908–1912.

[47] Dale Schuurmans and Finnegan Southey. 2002. Metric-Based Methods for Adaptive Model Selection and Regularization. *Machine Learning* 48, 1 (2002), 51–84.

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

[49] Weiwei Shi, Yihong Gong, Chris Ding, Zhiheng MaXiaoyu Tao, and Nanning Zheng. 2018. Transductive semi-supervised deep learning using min-max features. In *Proceedings of (ECCV)*. 299–315.

[50] Anders Søgaard. 2010. Simple Semi-Supervised Training of Part-of-Speech Taggers. In *Proceedings of the ACL 2010* (Uppsala, Sweden). USA, 205–208.

[51] Baochen Sun, Jiashi Feng, and Kate Saenko. 2016. Return of Frustratingly Easy Domain Adaptation. In *Proceedings of the Thirtieth AAAI, February 12-17, 2016, Phoenix, Arizona, USA*. 2058–2065.

[52] Antti Tarvainen and Harri Valpola. 2017. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in Neural Information Processing Systems 30*. 1195–1204.

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

[54] Thomas Wolf, Lysandre Debut, and et al. 2019. HuggingFace's Transformers: State-of-the-art Natural Language Processing. *ArXiv* abs/1910.03771 (2019).

[55] Jiawei Wu, Lei Li, and William Yang Wang. 2018. Reinforced Co-Training. In *Proceedings of the 2018 NAACL*. New Orleans, Louisiana, 1252–1262.

[56] Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. 2019. Unsupervised Data Augmentation for Consistency Training. *arXiv preprint arXiv:1904.12848* (2019).

[57] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V. Le. 2020. Self-Training With Noisy Student Improves ImageNet Classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[58] David Yarowsky. 1995. Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. In *33rd ACL*. Cambridge, Massachusetts, USA, 189–196.

[59] Kiran Zahra, Muhammad Imran, and Frank O. Ostermann. 2020. Automatic identification of eyewitness messages on twitter during disasters. *Information Processing & Management* 57, 1 (2020), 102107.

[60] Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. 2018. mixup: Beyond Empirical Risk Minimization. In *6th ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

[61] Ying Zhang, Tao Xiang, Timothy M. Hospedales, and Huchuan Lu. 2018. Deep Mutual Learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.