Neighborhood-Regularized Self-Training for Learning with Few Labels

Ran Xu,[1] Yue Yu,[2] Hejie Cui,[1] Xuan Kan,[1] Yanqiao Zhu,[3] Joyce Ho,[1] Chao Zhang,[2] Carl Yang[1]*

[1] Emory University, [2] Georgia Institute of Technology, [3] University of California, Los Angeles
{ran.xu,hejie.cui,xuan.kan,joyce.c.ho,j.carlyang}@emory.edu, {yueyu,chaozhang}@gatech.edu, yzhu@cs.ucla.edu

Abstract

Training deep neural networks (DNNs) with limited supervision has been a popular research topic as it can significantly alleviate the annotation burden. Self-training has been successfully applied in semi-supervised learning tasks, but one drawback of self-training is that it is vulnerable to the label noise from incorrect pseudo labels. Inspired by the fact that samples with similar labels tend to share similar representations, we develop a neighborhood-based sample selection approach to tackle the issue of noisy pseudo labels. We further stabilize self-training via aggregating the predictions from different rounds during sample selection. Experiments on eight tasks show that our proposed method outperforms the strongest self-training baseline with 1.83% and 2.51% performance gain for text and graph datasets on average. Our further analysis demonstrates that our proposed data selection strategy reduces the noise of pseudo labels by 36.8% and saves 57.3% of the time when compared with the best baseline. Our code and appendices will be uploaded to: https://github.com/ritaranx/NeST.

1 Introduction

In the era of deep learning, neural network models have achieved promising performance in most supervised learning settings, especially when combined with self-supervised learning techniques (Chen et al. 2020; Devlin et al. 2019; Hu et al. 2020; Zhu et al. 2022). However, they still require a sufficient amount of labels to achieve satisfactory performances on many downstream tasks. For example, in the text domain, curating NLP datasets often require domain experts to read thousands of documents and carefully label them with domain knowledge. Similarly, in the graph domain, molecules are examples naturally represented as graphs, and characterizing their properties relies on density functional theory (DFT) (Cohen, Mori-Sánchez, and Yang 2012) which often takes several hours. Such a dependency on labeled data is one of the barriers to deploy deep neural networks (DNNs) in real-world applications.

To better adapt the DNNs to target tasks with limited labels, one of the most popular approaches is *semi-supervised learning* (SSL), which jointly leverages unlabeled data and labeled data to improve the model's generalization power on the target task (Yang et al. 2021; Wang et al. 2022). Although generative models (Gururangan et al. 2019) and consistency-based regularization (Tarvainen and Valpola 2017; Miyato et al. 2018; Xie et al. 2020a) methods have been proposed for semi-supervised learning, they either suffer from the issue of limited representation power (Tsai, Lin, and Fu 2022) or require additional resources to generate high-quality augmented samples (e.g., for text classification, Xie et al. (2020a) generate augmented text via back-translation, which rely on a Machine Translation model trained with massive labeled sentence pairs). Consequently, they cannot be readily applied to low-resource scenarios.

Self-training is a proper tool to deal with the deficiency of labeled data via gradually enlarging the training set with pseudo-labeled data (Rosenberg, Hebert, and Schneiderman 2005). Specifically, it can be interpreted as a *teacher-student framework*: the teacher model generates pseudo labels for the unlabeled data, and the student model updates its parameters by minimizing the discrepancy between its predictions and the pseudo labels (Xie et al. 2020b; Mukherjee and Awadallah 2020). Though conceptually simple, self-training has achieved superior performance for various tasks with limited labels, such as image classification (Sohn et al. 2020; Rizve et al. 2021), natural language understanding (Du et al. 2020), sequence labeling (Liang et al. 2020), text generation (He et al. 2019), and graph learning (Hao et al. 2020). Self-training has also been successfully extended to other settings including weak supervision (Zhang et al. 2021b) and zero-shot learning (Li, Savarese, and Hoi 2022).

However, one major challenge of self-training is that it suffers from *confirmation bias* (Arazo et al. 2020) — when the teacher model memorizes some biases and generates incorrect pseudo labels, the student model will be reinforced to train with these wrong biases. As a result, the biases may amplify over iterations and deteriorates the final performance. To suppress the noisy pseudo labels in self-training, Xu et al. (2021); Zhang et al. (2021a); Sohn et al. (2020); Kim et al. (2022b) leverage model predictive confidence with a thresholding function, Mukherjee and Awadallah (2020); Tsai, Lin, and Fu (2022) propose to leverage model uncertainty to select samples with low uncertainty, and Wang et al. (2021) use meta-learning to conduct instance reweighting for sequence labeling. Although

*Corresponding author.

these approaches attempt to reduce the label noise, they select the data for self-training based on the model prediction only. However, the predictions of the deep neural network can be over-confident and biased (Guo et al. 2017; Kong et al. 2020; Kan, Cui, and Yang 2021), and directly using such predictions without any intervention to filter pseudo labels cannot effectively resolve the label noise issue. Another problem from self-training is *training instability*, as it selects pseudo-labeled data only based on the prediction of the *current* round. Due to the stochasticity involved in training neural networks (e.g., random initialization, training order), the prediction can be less stable (Yu et al. 2022), especially for the noisy pseudo-labeled data (Xia et al. 2022). Consequently, the noise in the previous rounds may propagate to later rounds, which deteriorate the final performance.

Motivated by the above, we propose NeST, a simple yet powerful approach guided by the data representations, to boost the performance of self-training for few-shot learning. Inspired by recent works indicating that the representations from deep neural networks can be discriminative and less affected by noisy labels (Li et al. 2021), we harness the features learned from the neural models to select the most reliable samples in self-training. In addition, several works have indicated that samples within the same category tend to share similar representations, such as category-guided text mining (Meng et al. 2020) and motif-driven graph learning (Zhang et al. 2020). Similarly, we hypothesize that a sample's pseudo label is more likely to be correct only if its prediction is similar to the neighbor labeled instances in the embedding space. To fulfill the denoising purpose, NeST creates the neighborhood for each unlabeled data by finding the top-k nearest labeled samples, then calculates the divergence between its current prediction and the label of its neighbors to rank the unlabeled data. As a result, only the instances with the lowest divergence will be selected for self-training, which mitigates the issue of label noise. Moreover, to robustly select the training samples for self-training, we aggregate the predictions on different iterations to promote samples that have lower uncertainty over multiple rounds for self-training.

We remark that NeST is an efficient substitution for existing self-training approaches and can be combined with various neural architectures. The contributions of this paper are:

- We propose NeST to improve the robustness of self-training for learning with few labels only.
- We design two additional techniques, namely neighborhood-regularized sample selection to reduce label noise, and prediction aggregation to alleviate the training instability issue.
- Experiments on 4 text datasets and 4 graph datasets with different volumes of labeled data verify that NeST improves the performance by 1.83% and 2.51% respectively and saves the running time by 57.3%.

2 Related Work

Self-training is one of the earliest approaches to semi-supervised learning (Rosenberg, Hebert, and Schneiderman 2005). The method uses a teacher model to generate new labels on which a student model is fitted. The major drawback of self-training is that it is vulnerable to label noise (Arazo et al. 2020). There are several popular approaches to stabilize the self-training process, such as using sample selection (Mukherjee and Awadallah 2020; Sohn et al. 2020) and reweighting strategies (Zhou, Kantarcioglu, and Thuraisingham 2012; Wang et al. 2021) to filter noisy labels or designing noise-aware loss functions (Liang et al. 2020; Tsai, Lin, and Fu 2022) to improve the model's robustness against incorrectly labeled data. In addition, data augmentation methods (Kim et al. 2022a; Chen, Yang, and Yang 2020; Zhang, Yu, and Zhang 2020) are also combined with self-training to improve the model's generalization ability.

Leveraging representation information has also been explored in semi-supervised learning. For example, Li, Xiong, and Hoi (2021); Zhao et al. (2022); Yu et al. (2021) improve the representation via contrastive learning to assist semi-supervised learning. Moreover, ACPL (Liu et al. 2022) and SimMatch (Zheng et al. 2022) aggregates the labels from their neighbors in the feature space. While these approaches also attempt to harness sample representations, they do not directly denoise the pseudo labeled data for boosting the performance of self-training, which is the focus of our work. One concurrent work (Lang, Vijayaraghavan, and Sontag 2022) combines data representations with the cut statistic to select high quality training data. In particular, it aims to select reliable subsets directly from the weakly-labeled data. Instead, our work focuses on using clean labeled data to better denoise instances in self-training.

3 Preliminaries

In this section, we first present the setup of semi-supervised learning and self-training, and then point out issues of the existing sample selection algorithms for self-training.

3.1 Task Definition

In this paper, we study the semi-supervised learning problem, which is defined as follows. Given a few labeled $\mathcal{X}_l = \{(x_i, y_i)\}_{i=1}^{L}$ and unlabeled data $\mathcal{X}_u = \{x_j\}_{j=1}^{U}$ ($L \ll U$), we seek to learn a predictor $f(x;\theta) : \mathcal{X} \to \mathcal{Y}$. Here $\mathcal{X} = \mathcal{X}_l \cup \mathcal{X}_u$ denotes all the input data and $\mathcal{Y}$ is the label set, which can either be discrete (for classification) or continuous (for regression). $f(x;\theta)$ is either a C-dimensional *probability simplex* for classification where C is the number of classes or a *continuous value* for regression.

3.2 Introduction to Self-training

Self-training can be interpreted as a teacher-student framework (Mukherjee and Awadallah 2020; Xie et al. 2020b), with θ_t and θ_s denoting the teacher and student model, respectively. The process of the self-training is in Alg. 1. We discuss the key components in self-training as belows.

Initialization of Models. The labeled data $\mathcal{X}_l$ are used to initialize the models as $\theta_s^{(0)} = \theta_t^{(0)} = \theta_{\text{init}}$, where

$$\theta_{\text{init}} = \min_{\theta} \mathcal{L}_{\text{sup}}(\theta) = \mathbb{E}_{(x_i,y_i)\in\mathcal{X}_l} \ell_{\text{sup}}\left(f(x_i;\theta), y_i\right). \quad (1)$$

Algorithm 1: Training Procedures of Self-training.

Input: Labeled and unlabeled samples $\mathcal{X}_l$, $\mathcal{X}_u$; Neural prediction model $f(\cdot;\theta)$; Unlabeled set $\widehat{\mathcal{X}_u}$; Number of self-training iterations T; Number of steps in each iteration T_1.

// Train the model on labeled data $\mathcal{X}_l$ as initialization.

Update θ_s, θ_t by Eq. 1 using Adam.

// Self-training.

for $t = 1, 2, \cdots, T$ **do**

  Select $\widehat{\mathcal{X}_u^t}$ ($|\widehat{\mathcal{X}_u^t}| = b$) with θ_t by Eq. 4.

  Adding $\widehat{\mathcal{X}_u^t}$ for self-training $\widehat{\mathcal{X}_u} = \widehat{\mathcal{X}_u} \cup \widehat{\mathcal{X}_u^t}$.

  Update pseudo labels $\widetilde{y}$ by Eq. 2 or 3 for $\widehat{\mathcal{X}_u^t}$.

  for $k = 1, 2, \cdots, T_1$ **do**

    Sample a minibatch $\mathcal{B}$ from $\widehat{\mathcal{X}_u}$.

    Update θ_s with loss $\mathcal{L}$ in Eq. 5 using Adam.

  Update teacher model $\theta_t \leftarrow \theta_s$.

Output: Final model $f(\cdot;\theta_s)$.

$\ell_{\text{sup}}(\cdot;\cdot)$ represents the supervised loss, which is the cross-entropy loss for classification and the mean squared error loss for regression.

Pseudo Label Generation with Teacher Model θ_t. We use the teacher model's prediction $f(x;\theta_t)$ to generate pseudo labels for $\mathcal{X}_u$. For classification problems, the pseudo labels can be written as

$$\widetilde{\boldsymbol{y}}_{\text{hard},j} = \begin{cases} 1, & \text{if } j = \underset{k \in \mathcal{Y}}{\text{argmax}} \left[f(x;\theta_t)\right]_k; \\ 0, & \text{else.} \end{cases} \tag{2}$$

For the regression task, since the output is a continuous value, the teacher model's output is directly used as the pseudo label

$$\widetilde{y} = f(x;\theta_t). \tag{3}$$

Sample selection. Directly using all the pseudo-labeled data for self-training often yields sub-optimal results, as the erroneous pseudo labels hurt the model performance. To mitigate this issue, recent works attempt to select only a subset of the unlabeled data for self-training. We denote the sample policies as $\psi(\cdot)$, which can be generally written as

$$\widehat{\mathcal{X}_u} = \psi\left(\mathcal{X}_u, f(x;\theta_t)\right). \tag{4}$$

We omit the superscript for simplicity. The common choice for $\psi(\cdot)$ including using predictive confidence (Sohn et al. 2020; Zhang et al. 2021a) or model uncertainty (Mukherjee and Awadallah 2020; Tsai, Lin, and Fu 2022).

Model Training and Update. With the generated pseudo labels, we then train a student model θ_s to minimize the loss for both labeled and unlabeled data by solving

$$\min_{\theta_s} \lambda \mathcal{L}_{\text{sup}}(\theta_s) + (1-\lambda)\mathbb{E}_{x_j \in \widehat{\mathcal{X}_u}} \ell_{\text{st}}\left(f(x_j;\theta_s), \widetilde{y}_j\right), \tag{5}$$

where $\mathcal{L}_{\text{sup}}$ is defined in Eq. 1, $\widehat{\mathcal{X}_u}$ is obtained via Eq. 4, and $\ell_{\text{st}} = \mathbb{1}\{[f(x_j;\theta_s)]_{\widetilde{y}_j} > \gamma\} \cdot \ell_{\text{sup}}$ is the loss function for unlabeled data with the thresholding function (Sohn et al. 2020; Xie et al. 2020a). We iterate the process by treating the trained student model as the teacher to generate new pseudo labels and train a new student model based on the new generated labels until the model converges.

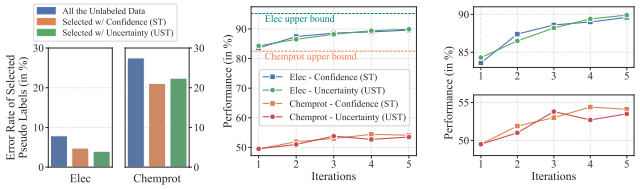


Figure 1: **Left:** The average error rate of all pseudo labels and the selected pseudo labels with different selection strategies. **Middle:** The performance on different self-training iterations. The upper bounds are the accuracy with full clean labels. **Right:** Zoom in of the performance in the middle.

3.3 Challenges of Self-training

To illustrate that the existing sample selection approaches are flawed and cannot resolve the label noise issue, we first demonstrate the performance of two widely-used selection criteria: *predictive confidence* (ST (Rosenberg, Hebert, and Schneiderman 2005; Du et al. 2020; Liang et al. 2020)) and *model uncertainty* (UST (Mukherjee and Awadallah 2020)) for self-training. The details of these two approaches are discussed in Appendix B. Note that, for these two approaches, we follow the original implementation to select the unlabeled set $\mathcal{X}_u$ in each iteration. We use a binary sentiment classification dataset *Elec* and a chemical relation extraction dataset *Chemprot* with ten classes as an example for *easier* and *harder* task, respectively. For both datasets, we first train the model with 30 clean labels per class.

Figure 1 shows the error rate of pseudo labels selected following these two criteria. We observe that these two methods are effective on easier tasks, where the selected data has a relatively low error rate. It achieves comparable performance with the fully-supervised method (95%) with less than 1% of the clean labeled data. However, for more challenging tasks with a larger number of classes, the performance of the initial model may not be satisfactory. The error rate of pseudo labels increases up to 16% on *Chemprot* compared with *Elec*. Consequently, the gap between semi-supervised learning and fully-supervised learning is even larger — more than 25% in terms of F1 score. This phenomenon suggests that the *label noise issue* is still the major challenge that hampers the self-training performance. Moreover, using *model uncertainty* (Gal and Ghahramani 2016) for sample selection does not fully address this challenge; the gain can be marginal on harder datasets.

Apart from the label noise issue, we also observe performance fluctuations over different self-training iterations. We name this as the *training instability* issue, which occurs when the teacher model only looks at the previous round and memorizes the label noise specifically in that round. Then in the next iteration, the student model can easily overfit to the

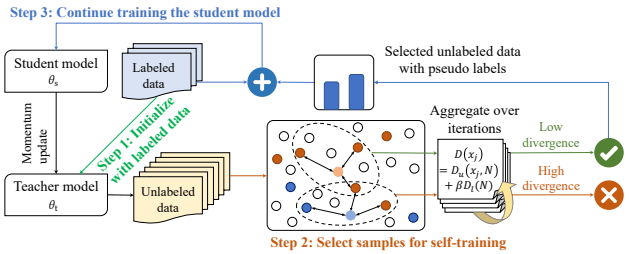


Figure 2: The framework of NeST. Red and blue points stand for labeled data with different labels. White points represent unlabeled data. Light red and blue points stand for predictions of unlabeled data.

noise and thus hurt the performance.

4 Method

We present NeST to improve the stability of self-training by tackling the challenges mentioned in the previous section. The overview of NeST is in Figure 2. Notably, we focus on the *sample selection* step (Eq. 4), and we propose two key components, namely neighborhood-regularized selection strategy and prediction aggregation to promote the performance. The details of the two designs will be discussed in Section 4.1 and Section 4.2 respectively.

4.1 Neighborhood-regularized Sample Selection

Prior works have demonstrated that leveraging embeddings from the deep neural networks can identify the noisy labeled data (Zhu, Dong, and Liu 2022). Motivated by this, we propose to harness the similarities in the embedding space to mitigate the issue of erroneous pseudo labels in self-training.

Concretely, for each unlabeled sample $\boldsymbol{x}_j$ with representation $\boldsymbol{v}_j$, we adopt the k-nearest neighbors (KNN) algorithm to find the most similar *labeled* samples in the feature space:

$$\mathcal{N}_j = \{x_i \mid x_i \in \mathcal{X}_l \cap \text{KNN}(\boldsymbol{v}_j, \mathcal{X}_l, k)\}, \tag{6}$$

where $\text{KNN}(\boldsymbol{v}_j, \mathcal{X}_l, k)$ denotes k labeled examples in $\mathcal{X}_l$ that are nearest to $\boldsymbol{v}_j$.

Divergence-based Sample Selection. We then calculate the scores for unlabeled samples $x_j \in \mathcal{X}_u$ based on the weighted divergence

$$\mathcal{D}(x_j) = \mathcal{D}_\text{u}(x_j, \mathcal{N}) + \beta \mathcal{D}_\text{l}(\mathcal{N}), \tag{7}$$

where unlabeled divergence $\mathcal{D}_\text{u}$ and labeled divergence $\mathcal{D}_\text{l}$ are defined below, and β is a hyperparameter. This score $\mathcal{D}(x_j)$ will be further used for sample selection.

Unlabeled Divergence $\mathcal{D}_\text{u}$. For each sample x_j in the unlabeled set with the neighbor set $\mathcal{N}$, we calculate the divergence between the prediction of x_j and labeled data in $\mathcal{N}$ as

$$\mathcal{D}_\text{u}(x_j, \mathcal{N}) = \sum_{(x_i, y_i) \in \mathcal{N}} d(f(x_j; \theta_t), y_i), \tag{8}$$

where d is the Kullback–Leibler (KL) divergence for classification and L2 distance for regression (same as follows). To interpret Eq. 8, we note that samples having the prediction *close* to the nearest labeled instances will have lower $\mathcal{D}_\text{u}$.

Labeled Divergence $\mathcal{D}_\text{l}$. It measures the divergence among the labels within the neighbor set $\mathcal{N}$. We first calculate the average label $\bar{y} = \sum_{(x_i, y_i) \in \mathcal{N}} \frac{y_i}{|\mathcal{N}|}$, and then measure the labeled divergence as

$$\mathcal{D}_\text{l}(\mathcal{N}) = \sum_{(x_i, y_i) \in \mathcal{N}} d(\bar{y}, y_i). \tag{9}$$

For each group $\mathcal{N}$, samples with similar labels will have smaller divergence $\mathcal{D}_\text{l}(\mathcal{N})$.

To summarize, a low divergence score $\mathcal{D}(x_j)$ indicates that the prediction of the unlabeled data point x_j is *close* to its neighbors, and the labels of its neighbors are *consistent*. Thus, we hypothesize that such samples are more likely to be correct and use them for self-training.

4.2 Robust Aggregation of Predictions from Different Iterations

The results in Figure 1c clearly demonstrate that only using the prediction on the current iteration for sample selection cause training instabilities. To effectively mitigate the bias in the current iteration and stabilize the self-training, we propose to exploit the training prediction at different training iterations more robustly (Xia et al. 2022). To achieve this, we aggregate the value $\mathcal{D}^{(t)}(x_j)$ in the t-th round as

$$\mu^{(t)}(x_j) = (1-m) \times \mu^{(t-1)}(x_j) + m \times \left(\mathcal{D}^{(t)}(x_j)\right), \tag{10}$$

where m is a hyperparameter bounded between 0 and 1 that controls the weight for previous rounds.

To interpret Eq. 10, we argue that $\mu(x_j)$ will be small only when the model outputs *consistently low* scores for a sample x_j in different iterations of self-training, as the model is more certain about these samples. On the contrary, if the model gives inconsistent predictions in different iterations, then the model is potentially uncertain about the prediction, thus adding its pseudo label in the next iteration may hurt the self-training process. Motivated by this idea, we remove the sample with inconsistent predictions over different iterations to further suppress noisy labels.

To put the above two strategies together, our policy for sample selection in the t-th round $\psi(\cdot)$ is mainly based on the value of $\mu^{(t)}(x_j)$ in Eq. 10. Specifically, in t-th iteration, we sample instances $x_j \in \mathcal{X}_u$ without replacement using the probability

$$p(x_j) \propto \frac{W - \mu^{(t)}(x_j)}{\sum_{x_u \in \mathcal{X}_u} (W - \mu^{(t)}(x_u))}, \tag{11}$$

where $W = \max_{x \in \mathcal{X}_u}(\mu^{(t)}(x))$ is the normalizing factor.

Remark. Our method introduces little computation overhead. For each unlabeled data, the neighborhood regularized sampling requires one extra kNN operation, which can be efficiently supported via FAISS (Johnson, Douze, and Jégou 2021). The $\mu^{(t)}(x_j)$ from previous iterations can be cached on disk and merged when selecting the training data for the new iteration. Compared with the previous state-of-the-art methods, which rely on model uncertainty and need to infer over unlabeled data multiple times during sample selection (Tsai, Lin, and Fu 2022; Mukherjee and Awadallah

Dataset	Domain	Task	# Train / Test	# Class	Metric
Elec	Reviews	Sentiment Analysis	25K / 25K	2	Acc.
AG News	News	Topic Classification	120K / 7.6K	4	Acc.
NYT	News	Topic Classification	30K / 3.0K	9	Acc.
Chemprot	Chemical	Relation Classification	12K / 1.6K	10	F1
BBBP	Physiology	Classification	1.6k / 204	2	ROC-AUC
BACE	Biophysics	Classification	1.2k / 151	2	ROC-AUC
Esol	Physical Chemistry	Regression	902 / 112	—	RMSE
Lipophilicity	Physical Chemistry	Regression	3.3k / 420	—	RMSE

Table 1: Statistics of text and molecule datasets.

2020), the overall running time of the above operations is much shorter. Other than the sample selection method $\psi(\cdot)$, NeST keeps other components intact and can be plugged-in with any noise-robust learning techniques (Menon et al. 2021) and neural architectures.

5 Experiments

5.1 Experiment Setup

We conduct experiments for semi-supervised learning on eight datasets to demonstrate the efficacy of NeST. Four of them are text-related tasks, including text classification and relation extraction. We employ the pre-trained `BERT` from the HuggingFace (Wolf et al. 2019) codebase for the implementation. The other four are graph-based tasks, where we choose molecular property prediction as the main task and use pre-trained `Grover-base`[1] (Rong et al. 2020) as the backbone. The same backbone is used for both NeST and baselines to ensure a fair comparison.

Semi-supervised Learning Settings. For each dataset, we train our method and baselines with different numbers of labeled data from {30, 50, 100} per class. The remaining in the training set is considered as *unlabeled data*. As suggested by Bragg et al. (2021), we keep the size of the validation set to be the same as the number of labeled data to simulate the realistic setting. For each dataset, we apply 3 runs on 3 splits and report the mean and standard deviations.

Parameter Settings. We use Adam (Kingma and Ba 2014) as the optimizer and tune the learning rate in {1e-5, 2e-5, 5e-5}. The batch size is selected from {8, 16, 32}. Other hyperparameters in NeST include T, T_1, γ for self-training, β, b, k for sample selection in Eq. 7, and λ in Eq. 5. We set $\beta = 0.1$, $\gamma = 0.9$, $\lambda = 0.5$, $m = 0.6$, $T = 5$, $T_1 = 1000$ for all datasets, and tune $b = c|\mathcal{X}_l|$ with $c \in \{3, 5, 10, 20\}$ for text datasets and $c \in \{1, 3, 5\}$ for graph datasets. We study the effect of k and c in Section 5.4. Details for each dataset are in Appendix C.2.

Baselines. We compare NeST with the following baselines. We use † to represent baselines designed for text-based tasks and ‡ to represent baselines for graph-based tasks.

- **BERT**† (Devlin et al. 2019; Lee et al. 2020) is the supervised baseline for text-based tasks.
- **Grover**‡ (Rong et al. 2020) is the supervised baseline for molecular property prediction.
- **Mean-Teacher (MT)** (Tarvainen and Valpola 2017) updates the teacher model as a moving average of the student model's weight and adds a consistency regularization between the student and teacher model.
- **Virtual Adversarial Training (VAT)** (Miyato et al. 2018) adds a regularization term between the sample with the adversarial noise and its prediction.
- **Self-training (ST)** (Rosenberg, Hebert, and Schneiderman 2005) is a conventional self-training method that adds *most confident* pseudo labeled data to labeled set.
- **UST** (Mukherjee and Awadallah 2020) selects data with *lowest uncertainty* using MC-dropout for self-training.
- **UDA**† (Xie et al. 2020a) adopts back translation and TF-IDF word replacement as the data augmentation and adds consistency loss on predictions on the augmented data.
- **MixText**† (Chen, Yang, and Yang 2020) interpolates training data in the hidden space via Mixup as the data augmentation to improve model performance.
- **CEST**† (Tsai, Lin, and Fu 2022) improves the UST method by designing the contrastive loss over sample pairs and noise-aware loss function.
- **InfoGraph**‡ (Sun et al. 2020) is a semi-supervised graph classification method via maximizing mutual information between graph and substructures.
- **ASGN**‡ (Hao et al. 2020) is a semi-supervised molecular property prediction that jointly exploits information from molecular structure and overall distribution.

We do not compare our methods with baselines without using pre-trained models (Gururangan et al. 2019) as their performance is worse than above baselines using pre-trained models as shown in Tsai, Lin, and Fu (2022).

5.2 Semi-supervised Learning on Text

Datasets. We conduct experiments on four widely used datasets in NLP. We adopt Elec (McAuley and Leskovec 2013) for sentiment classification, AGNews (Zhang, Zhao, and LeCun 2015) and NYT (Meng et al. 2020) for topic classification, and Chemprot (Taboureau et al. 2010) for chemical relation extraction in this set of experiments. The statistics and evaluation metrics for each dataset are shown in Table 1. We use `BioBERT` (Lee et al. 2020) as the backbone for Chemprot as it is a domain specific dataset (Cui et al. 2022) and use `BERT-base-uncased` for other datasets.

Results. Table 2 summarizes the experimental results on text datasets. We observe that NeST outperforms all baselines across all the four datasets under different volumes of labeled data, and the performance gain compared to the best baseline is around 1.83% on average. Note that UDA and MixText require *additional* data augmentation, which can be computationally expensive. Instead, NeST does not leverage any external resources but achieves better performance.

For other self-training baselines, we observe that they cannot outperform our proposed method. As we keep other components unchanged, the gain is mainly due to the pseudo-labeled data denosing benefit of NeST. We will illustrate this in Section 5.5. We also notice that the performance gain is more prominent on NYT and Chemprot

[1]Since we do not focus on developing better self-supervised learning methods, we do not compare with other graph neural network pretraining approaches (You et al. 2021; Hu et al. 2020; Zhu et al. 2021a,b)

Method	AG News (Accuracy, ↑)			Elec (Accuracy, ↑)			NYT (Accuracy, ↑)			Chemprot (F1, ↑)		
	30	50	100	30	50	100	30	50	100	30	50	100
BERT (2019)	80.2±1.4	83.1±1.6	86.0±1.1	83.7±1.9	87.0±1.0	90.2±1.2	79.4±1.6	83.0±1.1	85.7±0.5	49.1±2.3	51.2±1.7	54.9±1.4
Mean-Teacher (2017)	81.8±1.2	83.9±1.4	86.9±1.1	87.6±0.9	88.5±1.0	91.7±0.7	80.2±1.1	83.5±1.3	86.1±1.1	50.0±0.7	54.1±0.8	56.8±0.4
VAT (2018)	82.1±1.2	85.0±0.8	87.5±0.9	87.9±0.8	89.8±0.5	91.5±0.4	80.7±0.7	84.4±0.9	86.5±0.6	50.7±0.7	53.8±0.4	57.0±0.5
UDA (2020a)	86.5±0.9	87.1±1.2	87.8±1.2	89.6±1.1	91.2±0.6	92.3±1.0	—	—	—	—	—	—
MixText† (2020)	87.0±1.2	87.7±0.9	88.2±1.0	91.0±0.9	91.8±0.4	92.4±0.5	—	—	—	—	—	—
ST (2005; 2020)	86.0±1.4	86.9±1.0	87.8±0.6	89.6±1.2	91.4±0.4	92.1±0.5	85.4±0.9	86.9±0.5	87.5±0.5	54.1±1.1	55.3±0.7	59.3±0.5
UST (2020)	86.9*	87.4*	87.9*	90.0*	91.6*	91.9*	85.0±0.6	86.7±0.4	87.1±0.3	53.5±1.3	55.7±0.4	59.5±0.7
CEST‡ (2022)	86.5*	87.0*	88.4*	91.9*	92.1*	92.6*	—	—	—	—	—	—
NeST	**87.8±0.8**	**88.4±0.7**	**89.5±0.3**	**92.2±0.3**	**92.7±0.2**	**93.0±0.2**	**86.8±0.8**	**88.4±0.7**	**88.9±0.6**	**56.5±0.7**	**57.2±0.4**	**62.0±0.5**
Supervised		93.0*			95.3*			93.6±0.5			82.5±0.4	

Table 2: Performance on four datasets with various amounts of labeled data. The higher value always indicates better performance. **Bold** and underline indicate the best and second best results for each dataset, respectively (Same as below). *: The number is reported from the original paper. The implementation of CEST is *not publicly available*. †: The result is lower than the reported result in the original paper since they use a much larger development set. ‡: We remove the noise-aware loss as well as graph-based regularization for a fair comparison. The effect of these two terms is presented in table 3.

Method	AG News (Accuracy, ↑)			Elec (Accuracy, ↑)		
	30	50	100	30	50	100
CEST w/o NRL	86.5	87.5	88.4	91.9	92.1	92.6
CEST w/ NRL	87.1	88.0	88.9	92.2	92.4	92.8
NeST	87.8±0.8	88.4±0.7	89.5±0.3	92.2±0.3	92.7±0.2	93.0±0.2
NeST w/ NRL	**88.3±0.5**	**88.9±0.6**	**89.8±0.2**	**92.5±0.2**	**93.1±0.2**	**93.4±0.3**

Table 3: Performance comparison of CEST (Tsai, Lin, and Fu 2022) and NeST with noise-robust loss functions (NRL).

datasets which have more classes, indicating NeST can be better adapted to tasks with fine-grained classes.

Incorporating Noise-robust Loss Functions. To demonstrate NeST can be combined with other noise-robust loss functions, Table 3 further compares NeST with CEST (Tsai, Lin, and Fu 2022) which incorporates additional noise-aware loss (Menon et al. 2021) and graph-based regularization. The results show that these components can further improve the performance of NeST. Under both settings, NeST outperforms CEST with non-negligible margins, which justifies the efficacy of our proposed strategies.

5.3 Semi-supervised Learning on Graphs

Datasets. We choose molecular property prediction as the target task for graph classification. We conduct experiments on four widely used datasets from the MoleculeNet (Wu et al. 2018), including BBBP (Martins et al. 2012), BACE (Subramanian et al. 2016), Esol (Delaney 2004) and Lipophilicity (Gaulton et al. 2012). The statistics and evaluation metrics for each dataset are shown in Table 1. We adopt the scaffold split to determine train/validation/test.

Experiment results. From the results in Table 4, we can see that NeST outperforms all the baselines on all the datasets. In particular, the performance gain compared to the best baseline is around 2.5% on average. Compared to the Grover model using labeled data only, the gain is around 8.5% on average. Notice that the traditional self-training method (ST) sometimes performs even worse than Grover fine-tuned on labeled data only, because confidence-based selection introduces large label noise, which leads to many wrong predictions. With proper control of noisy pseudo labels, UST generally outperforms other baselines. However, since they do not consider neighbor information, their performance is not as good as NeST.

Adapting NeST to different backbones. We use two datasets as an example to demonstrate that NeST can be adapted to different backbones. Table 5 shows the results of training an AttentiveFP (Xiong et al. 2019), another popular GNN backbone based on graph attention networks for molecular property prediction. Unlike Grover, AttentiveFP is not pre-trained on massive molecules, but trained from scratch in our experiments. We can see that the performance of AttentiveFP is worse than Grover in most cases. A key reason is that pre-trained Grover has considerably more parameters than AttentiveFP, and incorporates rich domain knowledge with self-supervised learning on unlabeled molecules. Nevertheless, NeST still outperforms all the baselines by 1.5% on two datasets. This indicates that NeST does not rely on any specific architecture, and it serves as an effective plug-in module for different GNN models.

5.4 Parameter and Ablation Studies

We study the effect of different parameters of NeST on NYT and Chemprot with 30 labels per class, shown in Figure 3. Results on other datasets are in Appendix D. The performance first improves as k increases, because larger k allows more labeled data in each neighborhood, introducing less randomness and regularizing divergence calculation. When $k > 7$, the performance drops as the neighborhood includes labeled data far away, no longer serving as an appropriate regularizer. Similarly, as c gets larger, the performance first increases and then decreases. This indicates that when c is too small, the data are insufficient to train an effective student model, and when c is too large, unlabeled data tend to be noisier and can hurt the performance.

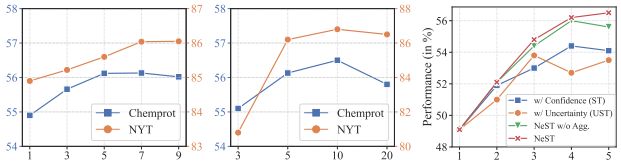
We also inspect components of NeST, shown in Fig-

Method	BBBP (ROC-AUC, ↑)			BACE (ROC-AUC, ↑)			Esol (RMSE, ↓)			Lipo (RMSE, ↓)		
	30	50	100	30	50	100	30	50	100	30	50	100
Grover (2020)	69.0±1.9	79.3±2.2	86.4±1.1	68.6±1.1	75.2±3.1	78.1±1.5	1.573±0.061	1.340±0.028	1.147±0.022	1.209±0.025	1.133±0.036	1.088±0.020
Mean-Teacher (2017)	71.8±1.2	80.9±1.4	87.2±0.6	69.5±0.8	76.6±0.7	80.2±0.2	1.500±0.018	1.298±0.020	1.104±0.021	1.154±0.027	1.091±0.018	1.067±0.035
VAT (2018)	72.2±1.0	80.0±1.6	88.5±0.5	69.8±0.5	76.5±0.4	78.6±0.8	1.492±0.039	1.264±0.031	1.056±0.014	1.199±0.016	1.089±0.021	1.050±0.027
ASGN (2020)	72.1±1.3	80.5±1.3	87.3±0.9	69.6±1.0	77.2±1.0	79.3±0.4	1.449±0.030	1.249±0.018	1.096±0.004	1.123±0.033	1.084±0.039	1.023±0.032
InfoGraph (2020)	72.5±0.9	81.2±0.3	88.5±0.5	70.2±0.6	77.8±0.8	80.4±0.5	1.414±0.041	1.222±0.017	1.082±0.013	1.125±0.024	1.082±0.027	1.039±0.020
ST (2005)	71.0±2.0	80.4±1.4	87.8±1.4	67.9±1.3	75.8±2.0	78.9±1.0	1.463±0.043	1.225±0.030	1.105±0.025	1.135±0.047	1.090±0.043	1.030±0.013
UST (2020)	71.7±1.1	81.8±0.7	88.8±0.4	69.9±0.3	78.0±0.4	80.4±0.5	1.408±0.026	1.174±0.034	1.023±0.010	1.115±0.020	1.069±0.014	1.010±0.009
NeST	**75.4±1.0**	**83.5±0.8**	**90.0±0.4**	**70.5±0.2**	**79.3±0.3**	**81.6±0.3**	**1.325±0.024**	**1.130±0.026**	**1.001±0.002**	**1.088±0.011**	**1.039±0.021**	**0.992±0.013**
Supervised◇ (2020)		93.6			87.8			0.888			0.563	

Table 4: Performance on four datasets with various amounts of labeled data. For classification datasets (BBBP, BACE), the higher value indicates better performance, while for regression datasets (Esol, Lipo), the lower value stands for better performance. **Bold** and underline indicate the best and second best results, respectively (Same as below).

Method	BBBP (ROC-AUC, ↑)			BACE (ROC-AUC, ↑)		
	30	50	100	30	50	100
AttentiveFP (2019)	66.4±1.3	75.6±1.1	82.9±0.5	68.0±1.4	72.2±1.3	74.0±0.9
Mean-Teacher (2017)	67.9±0.7	76.3±0.3	83.4±0.2	70.1±0.3	73.8±0.3	75.5±0.6
VAT (2018)	68.3±0.4	76.7±0.5	84.0±0.5	70.5±0.3	73.4±0.5	76.0±0.4
ASGN (2020)	70.0±0.4	77.1±0.2	84.1±0.3	70.9±0.5	74.9±0.7	77.9±0.6
InfoGraph (2020)	68.5±0.4	79.3±0.8	83.8±0.4	70.7±0.3	75.3±0.2	78.8±0.3
ST (2005)	69.8±0.2	76.0±1.4	84.0±1.1	67.5±0.5	71.4±3.4	76.0±0.8
UST (2020)	70.7±0.3	79.2±0.5	84.5±0.7	70.3±0.4	75.5±0.3	78.7±0.6
NeST	**71.2±0.4**	**80.7±0.5**	**85.2±0.5**	**72.2±0.4**	**76.6±0.5**	**80.7±0.5**

Table 5: Performance on two datasets with various amounts of labeled data using AttentiveFP as the backbone.



(a) k (b) c (c) Acc. over iters.

Figure 3: Effect of different components of NeST.

ure 3c. It is observed that both two strategies help to improve performance. The aggregation module stabilizes the self-training as the fluctuation issue has been mitigated.

5.5 Analysis

We take a closer look at the performance of NeST and other self-training algorithms using four text datasets. For each dataset, we study the setting with 30 labels per class.

Error of Pseudo Labels. To demonstrate how NeST reduces the noise of pseudo labels, we compare the error rate of pseudo labels selected by ST, UST and NeST. From Figure 4 we can notice that ST and UST tend to have high error rates due to their sole reliance on model prediction, and UST cannot stably improve the denoising ability of ST. In contrast, NeST significantly reduces the pseudo labels error rate by 36.8% on average compared to the best baseline. As a result, cleaner pseudo labels lead to performance gain.

Running Time. We compare the running time for one self-training iteration of NeST with UST and CEST, which

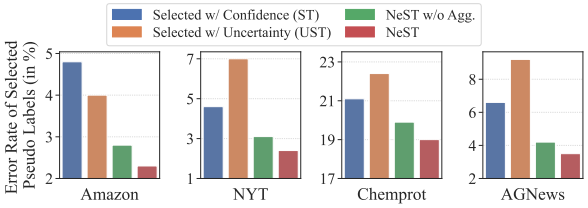


Figure 4: Error rates of pseudo labels selected by different methods. Agg. is the aggregation technique in Section 4.2.

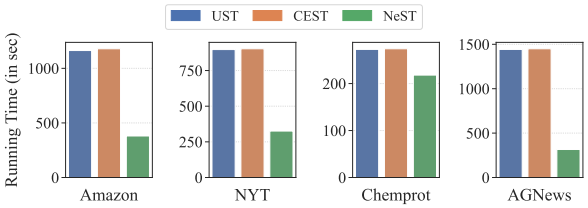


Figure 5: Running time of different methods.

are the two strongest baselines using model uncertainty. As shown in Figure 5, the running time is reduced by 57.3% on average. The gain is even more significant on larger datasets (e.g., AGNews) where inference multiple times becomes the efficiency bottleneck. Instead, the KNN operation takes less than 2 seconds with FAISS. To sum up, NeST is more efficient and can be readily combined with self-training.

6 Conclusion

In this paper, we propose NeST to improve sample selection in self-training for robust label efficient learning. We design a neighborhood-regularized approach to select more reliable samples based on representations for self-training. Moreover, we propose to aggregate the predictions on different iterations to stabilize self-training. Experiments on four text datasets and four graph datasets show that NeST outperforms the baselines by 1.83% and 2.51% on average. NeST also significantly reduce the noise in pseudo labels by 36.8% and reduce the running time by 57.3% when compared with the strongest baseline. For future works, we plan to extend NeST to other application domains and modalities.

Acknowledgement

We thank the anonymous reviewers for the valuable feedbacks. This research was partially supported by the internal funds and GPU servers provided by the Computer Science Department of Emory University. In addition, YY and CZ were partly supported by NSF IIS-2008334, IIS-2106961, and CAREER IIS-2144338. JH was supported by NSF grants IIS-1838200 and IIS-2145411.

References

Arazo, E.; Ortego, D.; Albert, P.; O'Connor, N. E.; and McGuinness, K. 2020. Pseudo-labeling and confirmation bias in deep semi-supervised learning. In *IJCNN*.

Bragg, J.; Cohan, A.; Lo, K.; and Beltagy, I. 2021. Flex: Unifying evaluation for few-shot nlp. *NeurIPS*.

Chen, J.; Yang, Z.; and Yang, D. 2020. Mixtext: Linguistically-informed interpolation of hidden space for semi-supervised text classification. In *ACL*.

Chen, T.; Kornblith, S.; Norouzi, M.; and Hinton, G. 2020. A simple framework for contrastive learning of visual representations. In *ICML*.

Cohen, A. J.; Mori-Sánchez, P.; and Yang, W. 2012. Challenges for density functional theory. *Chemical reviews*.

Cui, H.; Lu, J.; Ge, Y.; and Yang, C. 2022. How Can Graph Neural Networks Help Document Retrieval: A Case Study on CORD19 with Concept Map Generation. In *ECIR*.

Delaney, J. S. 2004. ESOL: estimating aqueous solubility directly from molecular structure. *J. Chem. Inf. Comp. Sci.*

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Du, J.; Grave, E.; Gunel, B.; Chaudhary, V.; Celebi, O.; Auli, M.; Stoyanov, V.; and Conneau, A. 2020. Self-training improves pre-training for natural language understanding. *arXiv preprint arXiv:2010.02194*.

Gal, Y.; and Ghahramani, Z. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *ICML*.

Gaulton, A.; Bellis, L. J.; Bento, A. P.; Chambers, J.; Davies, M.; Hersey, A.; Light, Y.; McGlinchey, S.; Michalovich, D.; Al-Lazikani, B.; et al. 2012. ChEMBL: a large-scale bioactivity database for drug discovery. *Nucleic acids research*.

Guo, C.; Pleiss, G.; Sun, Y.; and Weinberger, K. Q. 2017. On calibration of modern neural networks. In *ICML*.

Gururangan, S.; Dang, T.; Card, D.; and Smith, N. A. 2019. Variational Pretraining for Semi-supervised Text Classification. In *ACL*.

Hao, Z.; Lu, C.; Huang, Z.; Wang, H.; Hu, Z.; Liu, Q.; Chen, E.; and Lee, C. 2020. ASGN: An active semi-supervised graph neural network for molecular property prediction. In *KDD*.

He, J.; Gu, J.; Shen, J.; and Ranzato, M. 2019. Revisiting Self-Training for Neural Sequence Generation. In *ICLR*.

Hu, W.; Liu, B.; Gomes, J.; Zitnik, M.; Liang, P.; Pande, V.; and Leskovec, J. 2020. Strategies for Pre-training Graph Neural Networks. In *ICLR*.

Johnson, J.; Douze, M.; and Jégou, H. 2021. Billion-Scale Similarity Search with GPUs. *IEEE TBD*.

Kan, X.; Cui, H.; and Yang, C. 2021. Zero-shot scene graph relation prediction through commonsense knowledge integration. In *ECML-PKDD*.

Kim, H. H.; Woo, D.; Oh, S. J.; Cha, J.-W.; and Han, Y.-S. 2022a. ALP: Data Augmentation using Lexicalized PCFGs for Few-Shot Text Classification. In *AAAI*.

Kim, J.; Min, Y.; Kim, D.; Lee, G.; Seo, J.; Ryoo, K.; and Kim, S. 2022b. ConMatch: Semi-supervised Learning with Confidence-Guided Consistency Regularization. In *ECCV*.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kong, L.; Jiang, H.; Zhuang, Y.; Lyu, J.; Zhao, T.; and Zhang, C. 2020. Calibrated Language Model Fine-Tuning for In-and Out-of-Distribution Data. In *EMNLP*.

Lang, H.; Vijayaraghavan, A.; and Sontag, D. 2022. Training Subset Selection for Weak Supervision. In *NeurIPS*.

Lee, J.; Yoon, W.; Kim, S.; Kim, D.; Kim, S.; So, C. H.; and Kang, J. 2020. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*.

Li, J.; Savarese, S.; and Hoi, S. C. 2022. Masked Unsupervised Self-training for Zero-shot Image Classification. *arXiv preprint arXiv:2206.02967*.

Li, J.; Xiong, C.; and Hoi, S. C. 2021. Comatch: Semi-supervised learning with contrastive graph regularization. In *ICCV*.

Li, J.; Zhang, M.; Xu, K.; Dickerson, J. P.; and Ba, J. 2021. How does a Neural Network's Architecture Impact its Robustness to Noisy Labels? In *NeurIPS*.

Liang, C.; Yu, Y.; Jiang, H.; Er, S.; Wang, R.; Zhao, T.; and Zhang, C. 2020. Bond: Bert-assisted open-domain named entity recognition with distant supervision. In *KDD*.

Liu, F.; Tian, Y.; Chen, Y.; Liu, Y.; Belagiannis, V.; and Carneiro, G. 2022. ACPL: Anti-Curriculum Pseudo-Labelling for Semi-Supervised Medical Image Classification. In *CVPR*, 20697–20706.

Martins, I. F.; Teixeira, A. L.; Pinheiro, L.; and Falcao, A. O. 2012. A Bayesian approach to in silico blood-brain barrier penetration modeling. *J. Chem. Inf. Model.*

McAuley, J.; and Leskovec, J. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In *RecSys*.

Meng, Y.; Huang, J.; Wang, G.; Wang, Z.; Zhang, C.; Zhang, Y.; and Han, J. 2020. Discriminative topic mining via category-name guided text embedding. In *WWW*.

Menon, A. K.; Jayasumana, S.; Rawat, A. S.; Jain, H.; Veit, A.; and Kumar, S. 2021. Long-tail learning via logit adjustment. In *ICLR*.

Miyato, T.; Maeda, S.-i.; Koyama, M.; and Ishii, S. 2018. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *TPAMI*.

Mukherjee, S.; and Awadallah, A. 2020. Uncertainty-aware self-training for few-shot text classification. *NeurIPS*.

Rizve, M. N.; Duarte, K.; Rawat, Y. S.; and Shah, M. 2021. In Defense of Pseudo-Labeling: An Uncertainty-Aware Pseudo-label Selection Framework for Semi-Supervised Learning. In *ICLR*.

Rong, Y.; Bian, Y.; Xu, T.; Xie, W.; Wei, Y.; Huang, W.; and Huang, J. 2020. Grover: Self-supervised message passing transformer on large-scale molecular data. *NeurIPS*.

Rosenberg, C.; Hebert, M.; and Schneiderman, H. 2005. Semi-Supervised Self-Training of Object Detection Models. In *WACV/MOTION*.

Sohn, K.; Berthelot, D.; Carlini, N.; Zhang, Z.; Zhang, H.; Raffel, C. A.; Cubuk, E. D.; Kurakin, A.; and Li, C.-L. 2020. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *NeurIPS*.

Subramanian, G.; Ramsundar, B.; Pande, V.; and Denny, R. A. 2016. Computational modeling of β-secretase 1 (BACE-1) inhibitors using ligand based approaches. *J. Chem. Inf. Model.*

Sun, F.-Y.; Hoffman, J.; Verma, V.; and Tang, J. 2020. InfoGraph: Unsupervised and Semi-supervised Graph-Level Representation Learning via Mutual Information Maximization. In *ICLR*.

Taboureau, O.; Nielsen, S. K.; Audouze, K.; Weinhold, N.; Edsgärd, D.; Roque, F. S.; Kouskoumvekaki, I.; Bora, A.; Curpan, R.; Jensen, T. S.; et al. 2010. ChemProt: a disease chemical biology database. *Nucleic acids research*.

Tarvainen, A.; and Valpola, H. 2017. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. *NeurIPS*.

Tsai, A. C.-Y.; Lin, S.-Y.; and Fu, L.-C. 2022. Contrast-enhanced Semi-supervised Text Classification with Few Labels. In *AAAI*.

Wang, Y.; Chen, H.; Fan, Y.; Wang, S.; Tao, R.; Hou, W.; Wang, R.; Yang, L.; Zhou, Z.; Guo, L.-Z.; et al. 2022. USB: A Unified Semi-supervised Learning Benchmark for Classification. In *NeurIPS Datasets and Benchmarks Track*.

Wang, Y.; Mukherjee, S.; Chu, H.; Tu, Y.; Wu, M.; Gao, J.; and Awadallah, A. H. 2021. Meta Self-training for Few-shot Neural Sequence Labeling. In *KDD*.

Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C.; Moi, A.; Cistac, P.; Rault, T.; Louf, R.; Funtowicz, M.; et al. 2019. HuggingFace's Transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Wu, Z.; Ramsundar, B.; Feinberg, E. N.; Gomes, J.; Geniesse, C.; Pappu, A. S.; Leswing, K.; and Pande, V. 2018. MoleculeNet: a benchmark for molecular machine learning. *Chemical science*, 9(2).

Xia, X.; Liu, T.; Han, B.; Gong, M.; Yu, J.; Niu, G.; and Sugiyama, M. 2022. Sample Selection with Uncertainty of Losses for Learning with Noisy Labels. In *ICLR*.

Xie, Q.; Dai, Z.; Hovy, E.; Luong, M.-T.; and Le, Q. V. 2020a. Unsupervised Data Augmentation for Consistency Training. In *NeurIPS*.

Xie, Q.; Luong, M.-T.; Hovy, E.; and Le, Q. V. 2020b. Self-training with noisy student improves imagenet classification. In *CVPR*.

Xiong, Z.; Wang, D.; Liu, X.; Zhong, F.; Wan, X.; Li, X.; Li, Z.; Luo, X.; Chen, K.; et al. 2019. Pushing the boundaries of molecular representation for drug discovery with the graph attention mechanism. *Journal of medicinal chemistry*.

Xu, Y.; Ding, J.; Zhang, L.; and Zhou, S. 2021. DP-SSL: Towards Robust Semi-supervised Learning with A Few Labeled Samples. *NeurIPS*.

Yang, X.; Song, Z.; King, I.; and Xu, Z. 2021. A survey on deep semi-supervised learning. *arXiv preprint arXiv:2103.00550*.

You, Y.; Chen, T.; Shen, Y.; and Wang, Z. 2021. Graph Contrastive Learning Automated. In *ICML*.

Yu, Y.; Zhang, R.; Xu, R.; Zhang, J.; Shen, J.; and Zhang, C. 2022. Cold-Start Data Selection for Few-shot Language Model Fine-tuning: A Prompt-Based Uncertainty Propagation Approach. *arXiv preprint arXiv:2209.06995*.

Yu, Y.; Zuo, S.; Jiang, H.; Ren, W.; Zhao, T.; and Zhang, C. 2021. Fine-Tuning Pre-trained Language Model with Weak Supervision: A Contrastive-Regularized Self-Training Approach. In *NAACL*.

Zhang, B.; Wang, Y.; Hou, W.; Wu, H.; Wang, J.; Okumura, M.; and Shinozaki, T. 2021a. Flexmatch: Boosting semi-supervised learning with curriculum pseudo labeling. *NeurIPS*.

Zhang, J.; Yu, Y.; Li, Y.; Wang, Y.; Yang, Y.; Yang, M.; and Ratner, A. 2021b. WRENCH: A Comprehensive Benchmark for Weak Supervision. In *NeurIPS Datasets and Benchmarks Track*.

Zhang, R.; Yu, Y.; and Zhang, C. 2020. Seqmix: Augmenting active sequence labeling via sequence mixup. *EMNLP*.

Zhang, S.; Hu, Z.; Subramonian, A.; and Sun, Y. 2020. Motif-driven contrastive learning of graph representations. *arXiv preprint arXiv:2012.12533*.

Zhang, X.; Zhao, J.; and LeCun, Y. 2015. Character-level Convolutional Networks for Text Classification. In *NeurIPS*.

Zhao, Z.; Zhou, L.; Wang, L.; Shi, Y.; and Gao, Y. 2022. LaSSL: Label-guided Self-training for Semi-supervised Learning. In *AAAI*.

Zheng, M.; You, S.; Huang, L.; Wang, F.; Qian, C.; and Xu, C. 2022. SimMatch: Semi-supervised Learning with Similarity Matching. In *CVPR*.

Zhou, Y.; Kantarcioglu, M.; and Thuraisingham, B. 2012. Self-training with selection-by-rejection. In *ICDM*.

Zhu, Q.; Yang, C.; Xu, Y.; Wang, H.; Zhang, C.; and Han, J. 2021a. Transfer Learning of Graph Neural Networks with Ego-graph Information Maximization. In *NeurIPS*.

Zhu, Y.; Xu, Y.; Cui, H.; et al. 2022. Structure-enhanced heterogeneous graph contrastive learning. In *SDM*.

Zhu, Y.; Xu, Y.; Yu, F.; Liu, Q.; Wu, S.; and Wang, L. 2021b. Graph contrastive learning with adaptive augmentation. In *WWW*.

Zhu, Z.; Dong, Z.; and Liu, Y. 2022. Detecting corrupted labels without training a model to predict. In *ICML*.

A Dataset Information

- **AGNews** (Zhang, Zhao, and LeCun 2015): This dataset is a collection of more than one million news articles. It is constructed by Zhang, Zhao, and LeCun (2015) choosing the 4 largest topic classes from the original corpus. The total number of training samples is 120K and both validation and testing are 7.6K.
- **Elec** (McAuley and Leskovec 2013): This dataset is a subset of Amazon's product reviews for binary sentiment classification. It is constructed by McAuley and Leskovec (2013), including 25K training samples, and 25K testing samples.
- **NYT** (Meng et al. 2020): This dataset is collected using the New York Times API. It is constructed by Meng et al. (2020), including 30K training samples, and 3K testing samples.
- **ChemProt** (Taboureau et al. 2010). This is a 10-class relation extraction dataset constructed by Taboureau et al. (2010), containing 12K training samples and 1.6K testing samples.
- **BBBP** (Martins et al. 2012): This is a binary classification task for predicting whether a compound carries the permeability property of penetrating the blood-brain barrier. It contains 2039 molecules in total.
- **BACE** (Subramanian et al. 2016): This is a binary classification task for predicting compounds which could act as the inhibitors of human β-secretase 1 in the past few years. It contains 1513 molecules in total.
- **Esol** (Delaney 2004) is a regression task for predicting the solubility of compounds. It contains 1128 molecules in total.
- **Lipophilicity** (Gaulton et al. 2012) is a regression task for predicting the property that affects the molecular membrane permeability and solubility. It contains 4200 molecules in total.

B Introduction of Confidence-based and Uncertainty-based Selection Criterion

B.1 Confidence-based Method

For confidence-based strategy, we grant higher selection probability $\boldsymbol{y}_i = p(x_i)$ for sample x_i with higher predictive confidence

$$p(x_i) \propto \frac{\max\left(\boldsymbol{y}_i\right)}{\sum_{x_u \in \mathcal{X}_u} \max\left(\boldsymbol{y}_u\right)} \tag{12}$$

B.2 Uncertainty-based Method

For uncertainty-based method, the overall goal is to select samples that model is less uncertain about. For each data sample $x_i \in \mathcal{X}_u$, the information gain $\mathbb{B}$ with respect to its expected label y_i as

$$\mathbb{B}\left(y_i, \theta \mid x_i, D_U\right) = \mathbb{H}\left(y \mid x, \mathcal{X}_u\right) - \mathbb{E}_{p(\theta \mid D_U)}[\mathbb{H}(y \mid x, \theta)] \tag{13}$$

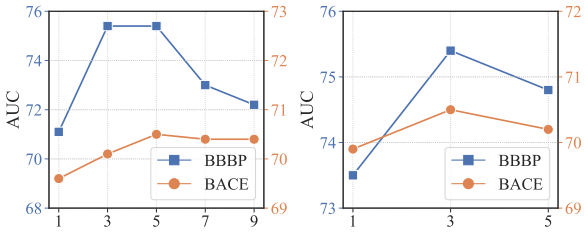
As the above equation is computationally intractable, BALD (Gal and Ghahramani 2016) is used to approximate the model uncertainty as

$$\widehat{\mathbb{B}}\left(y_i, \theta \mid x_i, \mathcal{X}_u\right) = -\sum_{c=1}^{C}\left(\frac{1}{M}\sum_{m=1}^{M}\boldsymbol{y}_c^m\right)\log\left(\frac{1}{M}\sum_{m=1}^{M}\boldsymbol{y}_c^m\right) + \frac{1}{M}\sum_{c=1}^{C}\sum_{m=1}^{M}\boldsymbol{y}_c^m\log(\boldsymbol{y}_c^m) \tag{14}$$

Then, the sample selection probability is calculated as

$$p(x_i) \propto \frac{1-\widehat{\mathbb{B}}\left(\boldsymbol{y}_i \mid x_i, \mathcal{X}_u\right)}{\sum_{x_u \in \mathcal{X}_u} 1-\widehat{\mathbb{B}}\left(\boldsymbol{y}_u \mid x_u, \mathcal{X}_u\right)}. \tag{15}$$

Note that, in the above equations, M denotes the number of inferences. Usually M is set to a large value ($M = 10$ is used in this work). This causes the inefficiency issue for deep neural networks, especially for large datasets.



(a) k (b) c

Figure 6: Effect of different hyperparameters of NeST for two graph learning datasets.

C Implementations

C.1 Computing Infrastructure

System: Ubuntu 18.04.3 LTS; Python 3.7; Pytorch 1.8.
CPU: Intel(R) Core(TM) i7-5930K CPU @ 3.50GHz.
GPU: NVIDIA A5000.

C.2 Hyperparameters

Details of hyperparameters in text datasets is shown in Table 6. Details of hyperparameters in graph datasets is shown in Table 7.

D Hyperparameter Study on Graph Datasets

We use BBBP and BACE as an example to study the effect of different hyperparameters on graph datasets, shown in Figure 6. We find that for these two datasets, $k = 5$ and $c = 3$ lead to the best performance.

E Error of Pseudo Labels and Running Time on Graph Datasets

Figure 7 shows the error of pseudo labels for four graph learning datasets. Note that for BBBP and BACE, we show the error rate of pseudo labels, same as the text datasets. For Esol and Lipo, since they are regression datasets, we show the average RMSE of the pseudo labels. Note that the lower value indicates higher quality of pseudo labels. The results

Hyper-parameter	AGNews	Elec	NYT	Chemprot
Dropout Ratio	0.1			
Maximum Tokens	128	256	160	64
Batch Size for Labeled Data	16	8	16	32
Batch Size for Unlabeled Data	32			
Learning Rate	2e-5	2e-5	1e-5	5e-5
Initialization Epochs	12	15	15	15
k	5	7	9	7
c	10	20	10	10

Table 6: Hyper-parameter configurations for semi-supervised text classification.

Hyper-parameter	BBBP	BACE	Esol	Lipo
Dropout Ratio	0.1			
Batch Size for Labeled Data	16	16	8	8
Batch Size for Unlabeled Data	16			
Weight Decay	1e-4			
Learning Rate	5e-5			
Initialization Epochs	10	10	12	12
k	5	5	3	5
c	3	3	1	3

Table 7: Hyper-parameter configurations for semi-supervised graph learning.

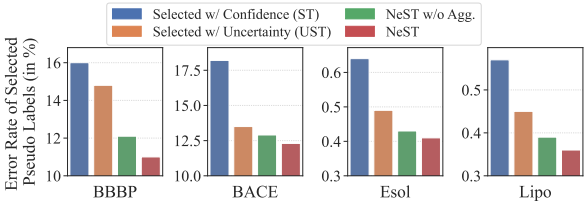


Figure 7: Error rates of pseudo labels selected by different methods. Agg. is the aggregation technique in Section 4.2.

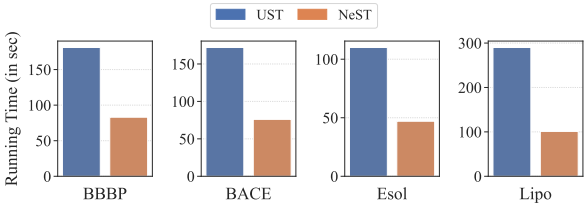


Figure 8: Running time of NeST and the best baseline (UST) on four graph datasets.

indicate that our proposed two strategies also improve the quality of pseudo labels.

Figure 8 shows the running time of the best baseline (UST) and NeST. Note that CEST is designed for text data, and cannot be directly used for molecular data. From the results, we find that NeST saves 54.1%–65.1% of the running time. We point out that the improvement of running time is rather smaller when compared with the text datasets. This is mainly because the size of datasets are much smaller, thus the additional time for multiple inferences in UST is not excessive. That being said, these results can well illustrate that NeST is more efficient and will serve as a complement to existing self-training approach.