Improving Self-training with Prototypical Learning for Source-Free Domain Adaptation on Clinical Text

Seiji Shimizu[1] **Shuntaro Yada**[1] **Lisa Raithel**[2,3,4] **Eiji Aramaki**[1]
[1]Nara Institute of Science and Technology (NAIST)
[2]BIFOLD – Berlin Institute for the Foundations of Learning and Data
[3]Quality & Usability Lab, Technische Universität Berlin
[4]German Research Center for Artificial Intelligence (DFKI)
shimizu.seiji.so8@is.naist.jp

Abstract

Domain adaptation is crucial in the clinical domain since the performance of a model trained on one domain (source) degrades seriously when applied to another domain (target). However, conventional domain adaptation methods often cannot be applied due to data sharing restrictions on source data. Source-Free Domain Adaptation (SFDA) addresses this issue by only utilizing a source model and unlabeled target data to adapt to the target domain. In SFDA, self-training is the most widely applied method involving retraining models with target data using predictions from the source model as pseudo-labels. Nevertheless, this approach is prone to contain substantial numbers of errors in pseudo-labeling and might limit model performance in the target domain. In this paper, we propose a Source-Free Prototype-based Self-training (SFPS) aiming to improve the performance of self-training. SFPS generates prototypes without accessing source data and utilizes them for prototypical learning, namely prototype-based pseudo-labeling and contrastive learning. Also, we compare entropy-based, centroid-based, and class-weights-based prototype generation methods to identify the most effective formulation of the proposed method. Experimental results across various datasets demonstrate the effectiveness of the proposed method, consistently outperforming vanilla self-training. The comparison of various prototype-generation methods identifies the most reliable generation method that improves the source model persistently. Additionally, our analysis illustrates SFPS can successfully alleviate errors in pseudo-labeling.

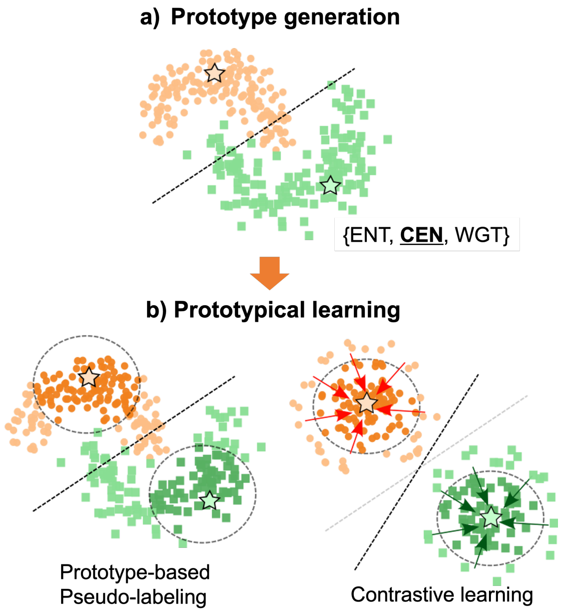

Figure 1: Illustration of SFPS. ⋆ denotes prototypes, and dotted lines denote the model's decision boundaries. First, we generate prototypes with either the entropy-based (**ENT**), centroid-based (**CEN**), or class-weights-based (**WGT**) method (a). **CEN** is chosen in this example. Then, we utilize these prototypes for prototypical learning (b), consisting of prototype-based pseudo-labeling and contrastive learning to update the source model and obtain distinct representations of target data.

1 Introduction

Domain adaptation is crucial in Clinical Natural Language Processing (Clinical NLP) since it is known that the performance of the model trained on one domain (source) degrades seriously on another domain (target) in the face of domain shifts such as different specialty or institution's formatting (Wu et al., 2014; Bethard et al., 2017; Miller et al., 2017). Despite the significant advancements in research on domain adaptation, most existing methods assume access to the labeled source data (Kouw and Loog, 2019; Ramponi and Plank, 2020). This assumption is frequently violated in the clinical domain, where data sharing is restricted due to patients' privacy concerns (Laparra et al., 2020). Source-Free Domain Adaptation (SFDA) addresses this issue by only utilizing a source model and unlabeled target

Proceedings of the 23rd Workshop on Biomedical Language Processing, pages 1–13
August 16, 2024. ©2024 Association for Computational Linguistics

data to adapt to the target domain (Liang et al., 2020; Chidlovskii et al., 2016).

Self-training (Kumar et al., 2010; Li and Zhang, 2019) has been shown to be a versatile and effective method for SFDA in computer vision (Yu et al., 2023). In Clinical NLP, a shared task was newly introduced in SemEval 2021 Task 10 (Laparra et al., 2021), prompting the development of SFDA methods on clinical text. Active learning and self-training combined with data augmentation emerged as widely applied methods (Su et al., 2021). The systematic comparison of the proposed methods indicates that active learning can reliably improve the source model's performance, while self-training is unreliable, failing to consistently outperform the source model (Su et al., 2022). However, active learning requires additional annotation on the target data, which can be difficult due to the expertise required for the annotation (Su et al., 2021). This necessitates improvement of the existing self-training method that does not rely on either additional annotation or source data for adapting the source model in Clinical NLP.

Prototypical learning (Snell et al., 2017; Wang et al., 2022) can potentially improve self-training, yet existing methods are not applicable in the SFDA setting. In general, self-training involves retraining the source model with target data by assigning the predictions from the source model as pseudo-labels. Nevertheless, pseudo-labels assigned in this manner contain a substantial number of errors and might limit the model's performance. Prototypical learning, such as prototype-based pseudo-labeling (Gu, 2020), and contrastive learning (Li et al., 2021) are proven to be effective for improving self-training (Yang et al., 2023; Mou et al., 2023; Zhou et al., 2023). However, existing methods assume access to labeled source data to generate reliable prototypes, making them inapplicable in source-free settings. How to generate reliable prototypes without accessing source data remains unanswered.

In this paper, we aim to provide answers to the following questions:

Q1: *Can prototypical learning improve self-training in **SFDA**?*

Q2: *Which method can generate reliable prototypes in the absence of labeled source data?*

Q3: *Is prototypical learning effective for alleviating errors in pseudo-labeling?*

To answer **Q1**, we introduce source-free prototype-based self-training (SFPS). Unlike existing methods, we generate prototypes without accessing source data (Fig. 1a) and leverage the generated prototypes for prototypical learning (Fig. 1b) consisting of prototype-based pseudo-labeling (Gu, 2020) and contrastive learning (Li et al., 2021) to alleviate errors in pseudo-labeling. To answer **Q2**, we explore three source-free prototype generation methods, namely entropy-based, centroid-based, and class-weights-based methods, inspired by the works in computer vision (Kim et al., 2021; Liang et al., 2020; Ding et al., 2024). To answer **Q3**, we compare the pseudo-label quality of SFPS and vanilla self-training.

We conduct experiments on negation detection and time expression recognition tasks from SemEval2021 Task10 with the source models trained on clinical texts. Our experimental results show the effectiveness of SFPS, outperforming vanilla self-training methods in most datasets. A comparison of various prototype-generation methods reveals that the centroid-based generation method can reliably improve the source model performance among other generation methods. We evaluate the pseudo-label quality of the proposed method and demonstrate the proposed method could successfully alleviate the errors in pseudo-labeling.

To summarize, we provide answers to the above questions as follows:

A1: *Prototypical learning can improve self-training in SFDA and consistently outperform vanilla self-training.*

A2: *Centroid-based prototype generation can reliably improve model performance without accessing the source data.*

A3: *Prototypical learning effectively alleviates the errors in pseudo-labels.*

2 Related Work

2.1 Source-Free Domain Adaptation

Source-free domain adaptation (SFDA) only uses a source model and unlabeled target data to adapt the model to the target domain. In recent years, SFDA has gained significant traction in computer vision. Various methods have been proposed, such as virtual domain generation (Tian et al., 2022), image style translation (Luan et al., 2017), and neighborhood clustering (Yang et al., 2021). Among them, self-training (Kumar et al., 2010; Li and Zhang,

2019) has been proven to be versatile and effective (Yu et al., 2023).

In contrast, SFDA methods in NLP are comparatively limited. Yin et al. (2022) introduced the SFDA method in question answering. They utilized an additional masking module during source model training and froze some weights of the masking module during self-training to maintain domain invariant knowledge. Zhang et al. (2021) aligned joint distributions between a trained source model and target domain samples using joint maximum mean discrepancy during knowledge distillation. In Clinical NLP, SemEval-2021 Task10 (Laparra et al., 2021) introduced a shared task for SFDA consisting of negation detection and time expression recognition. Only source model and unlabeled target data were provided to the participants. Self-training, active learning, and data augmentation methods were proposed. Although active learning can reliably improve the source model (Su et al., 2022), this method requires additional annotation on target data, which can be difficult due to data sharing restriction and expertise required for the annotation (Su et al., 2021). Hence, we extend the self-training method by developing the SFDA method, which is feasible in a wider range of situations.

2.2 Prototypical Learning

Prototypical learning, which aims to summarize a class by representative prototypes, has been widely used in semi-supervised and unsupervised learning (Wang et al., 2022; Snell et al., 2017). In self-training, pseudo-labeling based on the source model predictions suffers from errors. To improve self-training, prototype-based pseudo labeling (Gu, 2020) combined with contrastive learning (Li et al., 2021) are employed in semi-supervised learning (SSL) and unsupervised domain adaptation. Prototype-based pseudo-labeling assigns labels based on the similarities/distances between prototypes and target data representations instead of relying on model prediction. Contrastive learning enhances representations of target data by facilitating the formation of clusters of prototypes and text representations.

Yang et al. (2023) applied prototype-based pseudo-labeling and contrastive learning for text classification in SSL setting. They defined a centroid of the labeled data as a class-specific prototype and assigned pseudo-labels to unlabeled samples based on their distances from prototypes. These prototypes were then utilized as anchors to create high-density clusters of text representations via contrastive learning. In zero-shot cross-lingual named entity recognition, Zhou et al. (2023) defined the moving average of labeled data as a class prototype and used them for pseudo-labeling and contrastive learning. Mou et al. (2023) introduced prototype-based pseudo-labeling and contrastive learning in out-of-distribution intent classification. They defined randomly initialized embeddings as prototypes and updated them using embedded text representations of samples belonging to the same class. While prototype-based pseudo-labeling and contrastive learning have shown effectiveness in sentence and token classification, existing methods assume access to source data, making them inapplicable in the SFDA setting.

2.3 Source-free Prototype Generation

In the field of computer vision, various source-free prototype-generation methods have been proposed. Kim et al. (2021) defined samples with low entropy as prototypes and leveraged them for unsupervised learning by assigning pseudo-labels based on the distance between target image representations and prototypes. Liang et al. (2020) obtained the centroid of each class based on source model outputs. Pseudo-labels are assigned to unlabeled target data based on the distance between the class centroid and target samples. They further employ information maximization between target image representation and classifier output to update the model encoder. Ding et al. (2024) used the weights of the source classifier as class prototypes, constructing a class-balanced proxy source domain. The proxy source domain is then used for an inter-domain mixup that aligns the proxy domain and the target domain. While these works have independently combined source-free prototype generation with various SFDA techniques, we systematically compare the effectiveness of different prototype generation methods. Specifically, we combine various prototype generation methods with prototype-based pseudo-labeling and contrastive learning.

3 Problem Definition

Unlike conventional domain adaptation, we only have access to the source model and unlabeled target data in SFDA. Let $c \in C$ be a class from the set of all classes of interest, M the source model, and $X = \{x_0, ..., x_n\}$ the target data where n is the number of target samples. In general, source-free self-training aims to improve the performance of M using pseudo-labels $\hat{y}_i \in C$ assigned to x_i. How $\hat{y}_i$ is assigned and leveraged for the improvement depends on an individual method. However, only M and X are available for the adaptation.

For the generalizability of our study, the only assumption we make on the source model M is that it can be decomposed into an encoder (denoted by F) and classifier (denoted by G), i.e., $M := G(F(\cdot))$.

Strictly speaking, the definition of x_i differs between sentence classification and token classification. For sentence classification, one sample is equivalent to one input, i.e., $x_i = seq_i$ where seq is a sentence. For token classification, one sample contains a series of inputs, i.e., $x_i = [w^i_0, ..., w^i_m]$ where w is a token and m is a sentence length. For convenience, we use x_i to denote both a sentence and a token input.

4 Methodology

This section presents the proposed source-free prototype-based self-training (SFPS). The conceptual workflow is shown in Figure 2. First, we generate class prototypes from unlabeled target data with a source model (composed of F and G) using prototype generation (Section 4.1). Then, we utilize generated prototypes for prototypical learning, consisting of prototype-based pseudo-labeling and contrastive learning. Prototype-based pseudo-labeling assigns pseudo-labels based on the similarity between prototypes and text representations (Section 4.2). Contrastive learning improves the representation of target data by increasing/decreasing similarity between prototypes and target representations belonging to the same/a different class(Section 4.3). We describe the overall algorithm (Section 4.4) with the variants of SFPS.

4.1 Prototype generation

We generate a set of prototypes for a class $c \in C$ using only M and X. We experiment with three different source-free prototype generation methods, namely entropy-based (**ENT**), centroid-based (**CEN**), and class-weights-based (**WGT**) methods. In each method, we construct a set of prototypes for c, which is denoted by $\Phi_c = \{\phi^c_0, ..., \phi^c_K\}$ where K is the number of prototypes.

ENT: The entropy-based method chooses the representations of samples with high entropy as prototypes. Following Kim et al. (2021), we first calculate the lowest entropy for each class and set the largest value among them as a threshold (denoted by η), which is calculated by:

$$\begin{aligned} &\eta = \max\{\min(\mathcal{H}_c)|c \in C\}, \\ &\mathcal{H}_c = \{H(x_i)|x_i \in X_c\} \end{aligned} \tag{1}$$

where $H(x_i)$ denotes the entropy of x_i given by M, and X_c denotes the set of samples predicted as c by M.

Then, a set of prototypes is generated by:

$$\Phi_c = \{F(x_i)|x_i \in X, H(x_i) \leq \eta\} \tag{2}$$

CEN: The centroid-based method chooses the centroid for each class as a prototype generated by:

$$\Phi_c = \frac{1}{|X_c|} \sum_{x_i \in X_c} F(x_i) \tag{3}$$

WGT: The class-weights-based method chooses the weights of G corresponding to c as a prototype. Inspired by Ding et al. (2024), we also include the top $K-1$ most similar text representations $F(x_i)$ (denoted by $\mathcal{X}_c$) with the class-specific weights as prototypes. A set of prototypes is generated by:

$$\begin{aligned} &\Phi_c = \mathcal{X}_c \cup \{w_c\} \\ &\mathcal{X}_c = \{F(x_i)|x_i \in \max_{x_i;K-1}(\text{sim}(F(x_i), w_c))\} \end{aligned} \tag{4}$$

where w_c denotes the corresponding weights and $\max\limits_{x_i;K-1}(\text{sim}(\cdot))$ denotes choosing top $K-1$ samples with maximum similarity for each class. In this work, we use cosine similarity as a similarity measure (denoted by sim).

4.2 Prototype-based Pseudo-labeling

For prototype-based pseudo labeling, we find the most similar Φ_c to x_i and assign c as a label. Since relying on a single prototype can be unstable due to the unsupervised nature of the prototype generation method, we assign the label based on the prototype set Φ_c. To do so, we first calculate the similarity score s_c between Φ_c and x_i:

$$s_c(x_i) = \frac{1}{|\Phi_c|} \sum_k \text{sim}(\phi^c_k, F(x_i)) \tag{5}$$

and assign a pseudo-label by:

$$\hat{y}_i = \underset{c}{\text{argmax}}\ s_c(x_i), \forall c \in C \tag{6}$$

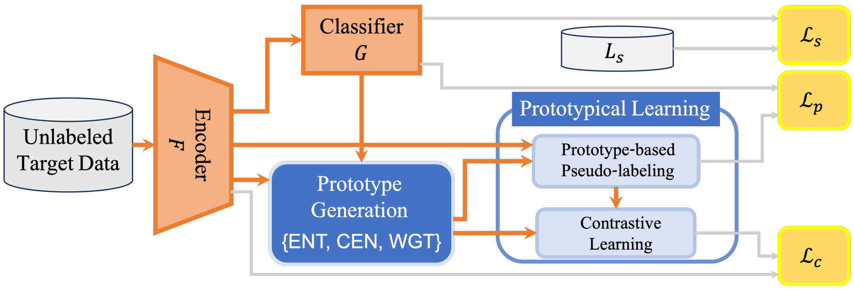

Figure 2: The workflow of SFPS. Prototypes are generated based on unlabeled target data and the source model (Section 4.1) and utilized for prototypical learning, consisting of prototype-based pseudo-labeling (Section 4.2) and contrastive learning (Section 4.3). This flow is illustrated by orange arrows. Three learning objectives are used for fine-tuning. $\mathcal{L}_p$ is an unsupervised loss between the pseudo-labels and the model predictions (Eq. 7). $\mathcal{L}_c$ is a contrastive loss based on the distance between prototypes and text representations (Eq. 9). $\mathcal{L}_s$ is a regularization loss based on an un-updated source model predictions (denoted by L_s) and the model predictions (Eq. 11).

Based on $\hat{y}_i$, the learning objective for fine-tuning M is given by:

$$\mathcal{L}_p = -\frac{1}{n}\sum_{i}^{n} \mathbb{1}(\hat{y}_i = c) \log \frac{\exp(p_i^c)}{\sum_{c \in C} \exp(p_i^c)} \tag{7}$$

where $\mathbb{1}(\cdot)$ is a indicator function and p_i^c is the predicted probability for the class c given by M with respect to x_i.

4.3 Contrastive Learning

Contrastive learning aims to obtain a distinct representation of x_i by increasing the similarity between $F(x_i)$ and prototypes of the same class while decreasing the similarity for the prototypes of the different classes. Inspired by Zhou et al. (2023), we employ the moving average of Φ_c per batch to update the representation of x_i, which is calculated by:

$$\mu_c = \alpha \frac{1}{|\Phi_c|} \sum_{k} \phi_k^c + (1-\alpha)\frac{1}{|B|}\sum_{i} F(x_i),$$
$$\forall i \in \{i | \hat{y}_i = c\}, \tag{8}$$

where α denotes the hyperparameter controlling the degree of updates and $|B|$ denotes the number of inputs per batch. In this way, we can ensure further stability of updates since μ_c is dynamically changing in accordance with $F(x_i)$ throughout the fine-tuning. Based on μ_c, we update $F(x_i)$ by the contrastive learning objective given by:

$$\mathcal{L}_c{=}-\sum_{i,c} \log \mathbb{1}(\hat{y}_i{=}c) \frac{\exp(\mathrm{sim}(F(x_i),\mu_c)/\beta)}{\sum_C \exp(\mathrm{sim}(F(x_i),\mu_c)/\beta)} \tag{9}$$

where β is a temperature coefficient.

4.4 Overall Algorithm

Algorithm 1 describes the whole process of SFPS. In line 14, we construct a set of pseudo-labels based on confidence scores. For sentence classification, we use the similarity score (Eq. 5) as a confidence score, i.e., confidence $= s_c(x_i)$. For token classification, x_i is a single token in a sentence. We take the average similarity scores of tokens for each sentence and use it as a confidence score. The confidence score for token classification is given by:

$$\text{confidence} = \frac{1}{m}\sum_{i}^{m} s_c(x_i) \tag{10}$$

Following Kim et al. (2021), we use pseudo-labels $\hat{y}_i^0$ given by the un-updated source model $M_0(x_i)$ as a regularizer so that the model does not diverge too much from the original source model (in line 15). The regularizer learning objective is given by:

$$\mathcal{L}_s = -\frac{1}{n}\sum_{i}^{n} \mathbb{1}(\hat{y}_i^0 = c) \log \frac{\exp(p_i^c)}{\sum_{c \in C} \exp(p_i^c)} \tag{11}$$

The overall objective $\mathcal{L}$ is the sum of Eq. 7, 9 and 11, namely:

$$\mathcal{L} = \mathcal{L}_p + \mathcal{L}_s + \mathcal{L}_c \tag{12}$$

Su et al. (2022) compared various formulations of self-training by changing the maximum number of iterations, the data construction strategy, and the model training strategy as parameters. Following this, we change the parameters of Algorithm 1 below to investigate which combination of prototype generation, data construction strategy, and model training strategy is most effective.

T the maximum number of iterations.

S_D the data construction strategy: KD to keep the training data from the previous iteration, or RD to reset.

S_M the model training strategy: KM to keep the model from the previous iteration, or RM to reset.

S_G the prototype generation methods: **ENT**, **CEN** or **WGT** to use entropy-based, centroid-based or class-weights-based method.

Algorithm 1: SFPS

```
Input:
    $M$: the source-domain model
    $X$: the target domain data
    $T$: the maximum number of iterations
    $L_p$: the pseudo-labels assigned via Eq. 6
    $L_s$: the pseudo-labels assigned by the un-updated source model
    $S_D$: the data construction strategy
    $S_M$: the model training strategy
    $S_P$: the prototype generation strategy
$M_0 \leftarrow \text{Copy}(M)$
$X_0 \leftarrow \text{Copy}(X)$
$L_p \leftarrow \emptyset$
for $t \leftarrow 0$ to $T$ do
    if $X = \emptyset$ then
        Stop training
    end
    if $S_D = RD$ then
        $L_p = \emptyset$
        $X = X_0$
    end
    Get $\Phi_c$ by Eq. 2, 3 or 4 based on $S_P$
    Get $s_c$ and $\hat{y}$ by Eq. 5 and 6
    $L_p \leftarrow \{(x_i, \hat{y}_i)$ for $x_i \in X$ if confidence $> \tau\}$
    $L_s \leftarrow \{(x_i, \hat{y}_i^0)$ for $(x_i, \hat{y}_i) \in L_p\}$
    if $L_{p_t} = \emptyset$ or $L_{p_t} = L_{p_{t-1}}$ then
        Stop training
    end
    if $S_D = KD$ then
        $X \leftarrow X - \{x_i$ for $(x_i, \hat{y}_i) \in L_{p_t}\}$
    end
    if $S_M = RM$ then
        $M \leftarrow M_0$
    end
    Fine-tune $M$ given $\Phi_c$, $L_p$, and $L_s$, using Eq. 12
end
```

5 Experiments

We conduct experiments with negation detection and time expression recognition datasets and compare a fully fine-tuned model (Oracle), an unadapted source model (Source), all vanilla self-training variants in Su et al. (2022) (Vanilla), and variants of SFPS. Vanilla and SFPS **do not utilize labeled target data** because our target problem setting is SFDA. However, datasets used in the experiments are fully annotated and used to train Oracle models.

We note that we do not expect SFPS to outperform Oracle. We consider Oracle as a upper bound for the performance in each dataset.

5.1 Datasets

We use the target data and source models from SemEval2021 Task10: negation detection and time expression recognition (Laparra et al., 2021). The provided source models were fine-tuned using English RoBERTa-base (Liu et al., 2019) as base models.

As described in Su et al. (2022), these two tasks are suitable for SFDA because (1) source data is difficult to share, (2) target data can not be easily annotated due to the complexity of the annotation task, and (3) models suffer a large performance loss in the face of domain shift in these tasks.

The negation detection task involves the classification of an event within a context span (indicated by special tokens "*<e>*" and "*</e>*") as in below.

> *She did not complain of <e> any fever </e>*

This task aims to correctly predict whether "*any fever*" is negated or not. The source model for this task was trained using Mayo Clinic clinical notes. Two target data for this task are clinical notes from Partners HealthCare's participation in the i2b2 2010 Challenge (**i2b2**) and ICU progress notes from Beth Israel in the MIMIC-III corpus (**MIMIC**).

The time expression recognition task involves sequence tagging, aiming to identify time entities in a document and assign them SCATE types (Bethard and Parker, 2016). An example sentence is given below.

> *the patient underwent surgery for gallstones on July 14, 2019*

The goal of this task is to predict "*July*" as *Month-Of-Year*, "*14*" as *Day-Of-Month* and "*2019*" as *Year*. The source model for this task was trained using clinical notes from Mayo Clinic as a part of SemEval 2018 Task 6 (Laparra et al., 2018). Two target datasets for this task are news articles from SemEval 2018 Task 6 (**News**) and reports from food security warning systems, including the UN World Food Programme and the Famine Early Warning Systems Network (**Food**).

We used the same development-test split as in Su et al. (2022) for all datasets as shown in Table 1. Note that the unit of numbers is a sentence for negation detection and a document for time expression recognition. Each document is preprocessed into sentences in time expression recognition.

	MIMIC	**i2b2**	**News**	**Food**
Dev	1916	1109	20	4
Test	7664	4436	79	13

Table 1: The number of development and test data. The unit is a sentence for negation detection (**MIMIC** and **i2b2**) and a document for time expression recognition (**News** and **Food**). Development sets are used for the adaptation.

5.2 Implementation Details

We used PyTorch[1] for the implementation of SFPS. For the preprocessing and implementation of vanilla self-training methods, we used the provided scripts from Su et al. (2022)[2]. We set the hyperparameters for SFPS, K, α, β, and τ to be 10, 0.9 and 0.5 respectively. We set the maximum number of iterations T to be 1 or 30. For a fair comparison, all the hyperparameters for fine-tuning except for learning rate are the same as in source model training and used for both SFPS and vanilla self-training. Since the proposed method has more learning objectives, we set the learning rate to 1.0×10^{-5} for SFPS and kept the original learning rate of 5.0×10^{-5} for vanilla self-training models. Other hyperparameters used for both SFPS and vanilla self-training are summarized in the Appendix A.1.

5.3 Results

We evaluated all models using the same evaluation metrics (F1, precision, and recall) as in Su et al. (2022). The results are the average of five different seeds. Due to limited space, we only present F1 scores (in percentage points) in Table 2. We provide the full results in Appendix A.2.

Several formulations of SFPS are shown to be effective. The best-performing SFPS formulations outperformed the best-performing vanilla self-training methods in most datasets (3 out of 4). In **MIMIC**, the best-performing formulation for SFPS was **WGT** with Single with 3.1 points higher F1 score than the best vanilla self-training method. In **i2b2**, no prototype-based method outperformed the best-performing vanilla self-training method. **CEN** with RD+RM has the highest F1 score among other SFPS formulations and scored 0.3 points below the best-performing vanilla self-training method. Given that the F1 score achieved by the best-performing vanilla self-training method is already high and relatively close to the Oracle, achieving further improvement may be challenging without the availability of labeled target data.

Strategy	**MIMIC**	**i2b2**	**News**	**Food**
Oracle	88.9	92.3	85.1	87.6
Source	63.5	84.6	79.1	78.5
Vanilla				
Single	67.4	87.1	79.1	77.4
KD+KM	66.5	87.6	79.3	77.7
KD+RM	68.7	87.6	79.2	78.2
RD+KM	55.4	87.8*	79.0	77.9
RD+RM	67.9	87.3	79.2	77.8
SFPS$_{\mathbf{ENT}}$				
Single	**71.3**	85.5	79.3	**78.9**
KD+KM	68.4	86.3	77.1	78.2
KD+RM	66.1	86.0	77.2	78.2
RD+KM	66.6	86.6	**80.1***	**78.7**
RD+RM	53.6	86.7	**79.9**	**78.9**
SFPS$_{\mathbf{CEN}}$				
Single	**70.3**	84.8	**79.4**	**79.2***
KD+KM	66.8	85.1	76.8	**79.2***
KD+RM	67.8	85.8	79.0	**78.8**
RD+KM	63.6	86.8	**79.9**	77.2
RD+RM	67.6	87.5	**79.8**	**78.5**
SFPS$_{\mathbf{WGT}}$				
Single	**71.8***	85.2	78.5	**78.3**
KD+KM	66.6	85.9	78.5	78.1
KD+RM	66.6	86.0	78.5	74.9
RD+KM	65.7	86.8	78.5	78.2
RD+RM	64.7	86.3	78.5	75.7

Table 2: The results of the experiment in F1 scores. Oracle, Source, and Vanilla denote the fully fine-tuned model, the un-updated source model, and vanilla self-training variants, respectively. KD, RD, KM, and RM are the variations due to the choice of the training strategies (see Section 4.4). $T = 1$ for Single and $T = 30$ for the others. The strategies that outperformed the source model are underlined. The scores above the best-performing vanilla method are in bold. Scores with a star are the best among all the self-training methods.

[1] https://pytorch.org/

[2] https://github.com/xinsu626/SourceFreeDomainAdaptation

In **News**, the best-performing SFPS formulation was **ENT** with RD+KM improving 0.8 points in F1 score from the best vanilla method. In **Food**, the best combination was **CEN** with Single and KD+KM, with an F1 score 1.0 point higher than the best vanilla method.

6 Discussion

Experimental results indicate that **CEN** with Single can reliably improve the source model compared with vanilla self-training. While no vanilla self-training method could outperform the un-updated source models in all datasets, **CEN** with Single outperformed the un-updated source model in all datasets and the best-performing vanilla self-training model in 3 out of 4 datasets. Since labeled target data is not available (or difficult to obtain) in SFDA, hyperparameter tuning is not realistic. Hence, it is important for an SFDA method to consistently outperform the source model regardless of task and dataset.

In the following section, we show that SFPS can properly alleviate the errors in pseudo-labeling (Section 6.1). We also conducted an ablation study to show both contrastive learning and regularization are effective for improving model performance (Section 6.2).

6.1 Pseudo-label Quality

Although we did not use any labeled data for adaptation, labels of the target data are available for all the datasets. In order to compare the pseudo-label qualities of the un-updated source model, vanilla self-training, and SFPS, we calculate the accuracy and macro F1 score of the pseudo-labeling by best-performing models of each method. The results are shown in Table 3. SFPS have the highest accuracy in all datasets and F1 score in 3 out of 4 datasets, indicating that prototype-based pseudo-labeling combined with contrastive learning could successfully alleviate the errors in pseudo-labeling.

	MIMIC		**i2b2**	
	ACC	**F1**	**ACC**	**F1**
Source	93.5	77.2	93.2	88.9
Vanilla	93.7	77.9	**94.0**	90.2
SFPS	**94.5**	**81.9**	**94.0**	**90.4**
	News		**Food**	
	ACC	**F1**	**ACC**	**F1**
Source	98.4	50.7	**95.9**	56.2
Vanilla	98.3	50.0	95.8	**56.4**
SFPS	**98.5**	**52.1**	**95.9**	56.3

Table 3: Pseudo-labeling accuracy and F1 score on development data. **Source** and **Vanilla** denotes the un-updated source model and the best-performing vanilla self-training model. SFPS successfully alleviates the errors in pseudo-labeling compared with vanilla self-training.

6.2 Ablation study

In order to investigate the effectiveness of the contrastive learning objective ($\mathcal{L}_c$) and the regularization term ($\mathcal{L}_s$) in Eq.12, we conduct an ablation study. We compared the performance of the model with (1) all objectives (Full), (2) without contrastive learning ($-\mathcal{L}_c$), (3) without regularization ($-\mathcal{L}_s$), and (4) only unsupervised learning objective ($-\mathcal{L}_c - \mathcal{L}_s$). Table 4 shows the results on four datasets in F1 score. In most datasets, the contrastive learning objective or regularization term alone improves the performance compared with only using unsupervised learning with pseudo labels. With the exception of **MIMIC**, Full models have the highest F1 scores, indicating that the contrastive learning objective combined with the regularization term is effective for improving model performance.

Objectives	**MIMIC**	**i2b2**	**News**	**Food**
Full	71.8	**87.5**	**80.1**	**79.2**
$-\mathcal{L}_c$	**72.3**	86.1	79.4	78.2
$-\mathcal{L}_s$	70.8	77.8	78.4	77.7
$-\mathcal{L}_c - \mathcal{L}_s$	71.5	46.8	78.4	77.7

Table 4: Results of ablation study in F1 score. Using both the contrastive-learning objective and the regularization term is effective in most of the datasets.

7 Conclusion

In this paper, we proposed source-free prototype-based self-training (SFPS) composed of prototype generation, prototype-based pseudo labeling, and contrastive learning. We compared entropy-based, centroid-based, and class-weights-based methods to identify the most reliable prototype generation method. We conducted experiments with two negation detection datasets and two time expression recognition datasets. Experimental results show the effectiveness of SFPS, consistently outperforming vanilla self-training. The comparison of various prototype generation methods reveals that the centroid-based generation method combined with a single iteration strategy is the most reliable formulation, outperforming the source model in all datasets and the best vanilla self-training model in 3 out of 4 datasets. Also, our analysis demonstrates that the proposed method can successfully alleviate errors in pseudo-labeling.

8 Limitations

We show that the proposed SFPS has an advantage over vanilla self-training methods in negation detection and time expression recognition tasks. However, this work has several limitations:(1) Experiments are only conducted on clinical/English corpora, limiting the generalizability of the results to other domains.; (2) The conclusions stated in this paper are based only on empirical evidence. Hence, they lack a theoretical analysis; (3) The gap between fully fine-tuned models and the proposed method is still large. This is expected, considering that the proposed method does not utilize labeled data at all for the adaptation. Yet, the model performance can be improved via task-specific modules such as class balancing for time expression recognition; (4) Although we tackled both sentence and token classification, the tasks employed in this experiment are limited in number. It is desirable to test the effectiveness of the proposed method in other tasks in the clinical domain and other domains as well.

Acknowledgements

This work was supported by JST CREST Grant Number JPMJCR22N1, and Cross-ministerial Strategic Innovation Promotion Program (SIP) on "Integrated Health Care System" Grant Number JPJ012425, Japan. We are grateful for the support of the German Research Foundation (DFG) under the trilateral ANR-DFG-JST AI Research project KEEPHA (DFG-442445488), and the German Federal Ministry of Education and Research under the grant BIFOLD24B.

References

Steven Bethard and Jonathan Parker. 2016. A semantically compositional annotation scheme for time normalization. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3779–3786, Portorož, Slovenia. European Language Resources Association (ELRA).

Steven Bethard, Guergana Savova, Martha Palmer, and James Pustejovsky. 2017. SemEval-2017 task 12: Clinical TempEval. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 565–572, Vancouver, Canada. Association for Computational Linguistics.

Boris Chidlovskii, Stephane Clinchant, and Gabriela Csurka. 2016. Domain adaptation in the absence of source domain data. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 451–460, New York, NY, USA. Association for Computing Machinery.

Yuhe Ding, Lijun Sheng, Jian Liang, Aihua Zheng, and Ran He. 2024. Proxymix: Proxy-based mixup training with label refinery for source-free domain adaptation. *Neural Netw.*, 167(C):92–103.

Xiaowei Gu. 2020. A self-training hierarchical prototype-based approach for semi-supervised classification. *Information Sciences*, 535:204–224.

Youngeun Kim, Donghyeon Cho, Kyeongtak Han, Priyadarshini Panda, and Sungeun Hong. 2021. Domain adaptation without source data. *IEEE Transactions on Artificial Intelligence*, 2(6):508–518.

Wouter M. Kouw and Marco Loog. 2019. An introduction to domain adaptation and transfer learning.

Abhishek Kumar, Avishek Saha, and Hal Daume. 2010. Co-regularization based semi-supervised domain adaptation. In *Advances in Neural Information Processing Systems*, volume 23. Curran Associates, Inc.

Egoitz Laparra, Steven Bethard, and Timothy A Miller. 2020. Rethinking domain adaptation for machine learning over clinical language. *JAMIA Open*, 3(2):146–150.

Egoitz Laparra, Xin Su, Yiyun Zhao, Özlem Uzuner, Timothy Miller, and Steven Bethard. 2021. SemEval-2021 task 10: Source-free domain adaptation for semantic processing. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 348–356, Online. Association for Computational Linguistics.

Egoitz Laparra, Dongfang Xu, Ahmed Elsayed, Steven Bethard, and Martha Palmer. 2018. SemEval 2018 task 6: Parsing time normalizations. In *Proceedings of the 12th International Workshop on Semantic Evaluation*, pages 88–96, New Orleans, Louisiana. Association for Computational Linguistics.

Junnan Li, Pan Zhou, Caiming Xiong, and Steven Hoi. 2021. Prototypical contrastive learning of unsupervised representations. In *International Conference on Learning Representations*.

Limin Li and Zhenyue Zhang. 2019. Semi-supervised domain adaptation by covariance matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(11):2724–2739.

Jian Liang, Dapeng Hu, and Jiashi Feng. 2020. Do we really need to access the source data? Source hypothesis transfer for unsupervised domain adaptation. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 6028–6039. PMLR.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Fujun Luan, Sylvain Paris, Eli Shechtman, and Kavita Bala. 2017. Deep photo style transfer.

Timothy Miller, Dmitriy Dligach, Steven Bethard, Chen Lin, and Guergana Savova. 2017. Towards generalizable entity-centric clinical coreference resolution. *Journal of Biomedical Informatics*, 69:251–258.

Yutao Mou, Xiaoshuai Song, Keqing He, Chen Zeng, Pei Wang, Jingang Wang, Yunsen Xian, and Weiran Xu. 2023. Decoupling pseudo label disambiguation and representation learning for generalized intent discovery. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9661–9675, Toronto, Canada. Association for Computational Linguistics.

Alan Ramponi and Barbara Plank. 2020. Neural unsupervised domain adaptation in NLP—A survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6838–6855, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 4080–4090, Red Hook, NY, USA. Curran Associates Inc.

Xin Su, Yiyun Zhao, and Steven Bethard. 2021. The University of Arizona at SemEval-2021 task 10: Applying self-training, active learning and data augmentation to source-free domain adaptation. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 458–466, Online. Association for Computational Linguistics.

Xin Su, Yiyun Zhao, and Steven Bethard. 2022. A comparison of strategies for source-free domain adaptation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8352–8367, Dublin, Ireland. Association for Computational Linguistics.

Jiayi Tian, Jing Zhang, Wen Li, and Dong Xu. 2022. Vdm-da: Virtual domain modeling for source data-free domain adaptation. *IEEE Transactions on Circuits and Systems for Video Technology*, 32(6):3749–3760.

Haobo Wang, Ruixuan Xiao, Yixuan Li, Lei Feng, Gang Niu, Gang Chen, and Junbo Zhao. 2022. PiCO: Contrastive label disambiguation for partial label learning. In *International Conference on Learning Representations*.

Stephen Wu, Timothy Miller, James Masanz, Matt Coarr, Scott Halgrim, David Carrell, and Cheryl Clark. 2014. Negation's not solved: Generalizability versus optimizability in clinical natural language processing. *PLOS ONE*, 9(11):1–11.

Shiqi Yang, Yaxing Wang, Joost van de Weijer, Luis Herranz, and Shangling Jui. 2021. Generalized source-free domain adaptation.

Weiyi Yang, Richong Zhang, Junfan Chen, Lihong Wang, and Jaein Kim. 2023. Prototype-guided pseudo labeling for semi-supervised text classification. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16369–16382, Toronto, Canada. Association for Computational Linguistics.

M. Yin, B. Wang, Y. Dong, and C. Ling. 2022. Source-free domain adaptation for question answering with masked self-training.

Zhiqi Yu, Jingjing Li, Zhekai Du, Lei Zhu, and Heng Tao Shen. 2023. A comprehensive survey on source-free domain adaptation.

Bo Zhang, Xiaoming Zhang, Yun Liu, Lei Cheng, and Zhoujun Li. 2021. Matching distributions between model and data: Cross-domain knowledge distillation for unsupervised domain adaptation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5423–5433, Online. Association for Computational Linguistics.

Ran Zhou, Xin Li, Lidong Bing, Erik Cambria, and Chunyan Miao. 2023. Improving self-training for cross-lingual named entity recognition with contrastive and prototype learning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4018–4031, Toronto, Canada. Association for Computational Linguistics.

A Appendix

A.1 Hyperparameters

Except for the learning rate, we used the same hyperparameters for both vanilla self-training and prototype-based self-training. For both negation detection and time expression recognition tasks, we set the hyperparameters to the same values as Su et al. (2022), which are summarized in Table 5 and 6.

Hyperparameter	Value
maximum sequence length	128
batch size	8
epochs	10
gradient accumulation steps	4
learning rate warm up steps	0
weight decay	0.0
adam epsilon	1.0×10^{-8}
maximum gradient norm	1.0

Table 5: Hyperparameters for negation detection

Hyperparameter	Value
maximum sequence length	271
batch size	2
epochs	3
gradient accumulation steps	1
learning rate warm up steps	500
weight decay	0.01
adam epsilon	1.0×10^{-8}
maximum gradient norm	1.0

Table 6: Hyperparameters for time expression recognition

All models were trained on AdamW (Loshchilov and Hutter, 2019) and a single NVIDIA Quadro RTX 8000 GPU. A training process took about 30 minutes per fine-tuning.

A.2 Full Results

The full results (in percentage points) for negation detection are presented in Table 7, and the results for time expression recognition are presented in Table 8.

Strategy	**MIMIC**			**i2b2**		
	F1	P	R	F1	P	R
Oracle	88.9	88.4	89.5	92.3	93.3	91.3
Source	63.5	93.8	48.0	84.6	92.6	77.9
Vanilla						
Single	67.4	94.7	52.4	87.1	95.1	80.4
KD+KM	66.5	95.4	51.1	87.6	94.4	81.8
KD+RM	68.7	95.4	53.6	87.6	95.3	81.0
RD+KM	55.4	75.7	43.7	87.8*	94.2	82.3
RD+RM	67.9	95.5	52.6	87.3	94.6	81.2
SFPS$_{\mathbf{ENT}}$						
Single	**71.3**	90.4	58.9	85.5	88.9	82.4
KD+KM	68.4	92.7	54.3	86.3	94.1	79.7
KD+RM	66.1	95.0	50.8	86.0	93.0	79.9
RD+KM	66.6	94.5	51.5	86.6	92.9	81.1
RD+RM	53.6	74.8	41.8	86.7	92.1	81.9
SFPS$_{\mathbf{CEN}}$						
Single	**70.3**	89.0	58.3	84.8	89.5	80.7
KD+KM	66.8	95.1	51.6	85.1	91.6	79.6
KD+RM	67.8	92.9	53.5	85.8	93.2	79.5
RD+KM	63.6	96.2	47.7	86.8	93.4	81.1
RD+RM	67.6	92.7	53.5	87.5	94.1	81.8
SFPS$_{\mathbf{WGT}}$						
Single	**71.8***	88.7	60.3	85.2	88.6	82.0
KD+KM	66.6	94.5	51.7	85.9	93.0	79.9
KD+RM	66.6	93.8	51.8	86.0	93.9	79.4
RD+KM	65.7	95.0	50.3	86.8	93.8	80.8
RD+RM	64.7	93.4	50.4	86.3	90.7	82.3

Table 7: The results in the negation detection task. Oracle, Source, and Vanilla denote the fully fine-tuned model, an unadapted source model, and vanilla self-training variants, respectively. $T = 1$ for Single and $T = 30$ for the others. The strategies outperformed the source model are underlined. The scores above the best-performing vanilla method are in bold. Scores with a star are the best among all the self-training methods.

Strategy	**News**			**Food**		
	F1	P	R	F1	P	R
Oracle	85.1	85.4	84.8	87.6	85.7	89.7
Source	79.1	79.5	78.7	78.5	82.9	74.6
Vanilla						
Single	79.1	79.8	78.5	77.4	80.7	74.5
KD+KM	79.3	78.4	80.2	77.7	79.9	75.7
KD+RM	79.2	78.4	80.0	78.2	80.8	75.7
RD+KM	79.0	78.1	79.9	77.9	80.0	75.8
RD+RM	79.2	78.3	80.1	77.8	80.0	75.8
SFPS$_{\textbf{ENT}}$						
Single	79.3	80.9	77.7	**78.9**	84.8	73.9
KD+KM	77.1	82.4	72.5	78.2	87.5	70.8
KD+RM	77.2	81.7	73.2	78.2	87.3	70.9
RD+KM	**80.1***	80.8	79.3	**78.7**	83.2	74.7
RD+RM	**79.9**	81.2	78.7	**78.9**	83.7	74.6
SFPS$_{\textbf{CEN}}$						
Single	**79.4**	81.2	77.6	**79.2***	85.0	74.1
KD+KM	76.8	81.4	72.6	**79.2***	86.9	72.9
KD+RM	79.0	81.3	76.9	**78.8**	85.7	73.0
RD+KM	**79.9**	79.8	80.0	77.2	78.9	75.7
RD+RM	**79.8**	80.5	79.1	**78.5**	82.4	75.0
SFPS$_{\textbf{WGT}}$						
Single	78.5	80.8	76.3	**78.3**	84.2	73.2
KD+KM	78.5	80.8	76.3	78.1	84.4	72.7
KD+RM	78.5	80.8	76.3	74.9	88.3	65.2
RD+KM	78.5	80.8	76.3	78.2	84.1	73.0
RD+RM	78.5	80.8	76.3	75.7	88.3	66.6

Table 8: The results in time expression recognition task. Oracle, Source, and Vanilla denote the fully fine-tuned model, an un-adapted source model, and vanilla self-training variants, respectively $T = 1$ for Single and $T = 30$ for the others. The strategies that outperformed the source model are underlined. The scores above the best-performing vanilla method are in bold. Scores with a star are the best among all the self-training methods.