Friend-training: Learning from Models of Different but Related Tasks

Mian Zhang†, Lifeng Jin⋄, Linfeng Song⋄, Haitao Mi⋄, Xiabing Zhou† and Dong Yu⋄
†Soochow University, Suzhou, China
mzhang2@stu.suda.edu.cn, zhouxiabing@suda.edu.cn
⋄Tencent AI Lab, Bellevue, WA, USA
{lifengjin,lfsong,haitaomi,dyu}@tencent.com

Abstract

Current self-training methods such as standard self-training, co-training, tri-training, and others often focus on improving model performance on a single task, utilizing differences in input features, model architectures, and training processes. However, many tasks in natural language processing are about different but related aspects of language, and models trained for one task can be great teachers for other related tasks. In this work, we propose friend-training, a cross-task self-training framework, where models trained to do different tasks are used in an iterative training, pseudo-labeling, and retraining process to help each other for better selection of pseudo-labels. With two dialogue understanding tasks, conversational semantic role labeling and dialogue rewriting, chosen for a case study, we show that the models trained with the friend-training framework achieve the best performance compared to strong baselines.

1 Introduction

Many different machine learning algorithms, such as self-supervised learning (Mikolov et al., 2013; Devlin et al., 2019; Liu et al., 2021), semi-supervised learning (Yang et al., 2021) and weakly supervised learning (Zhou, 2018), aim at using unlabeled data to boost performance. They have been of even greater interest recently given the amount of unlabeled data available. Self-training (Scudder, 1965) is one semi-supervised learning mechanism aiming to improve model performance through pseudo-labeling and has been successfully applied to computer vision (Lee et al., 2013; Chen et al., 2021), natural language processing (Dong and Schäfer, 2011; Bhat et al., 2021) and other fields (Wang et al., 2019; Kahn et al., 2020).

The main challenge of self-training is how to select high-quality pseudo-labels. Current self-training algorithms mainly focus on a single task when assessing the quality of pseudo-labels and suffer from gradual drifts of noisy instances (Zhang et al., 2021). This work is motivated by the observation that learning targets of tasks represent different properties of the inputs, and some properties are shared across the tasks which can be used as supervision from one task to another. Such properties include certain span boundaries in dependency and constituency parsing, and some emotion polarities in sentiment analysis and emotion detection. Two dialogue understanding tasks, conversational semantic role labeling (CSRL) and dialogue rewriting (DR), are also such a pair, with shared properties such as coreference and zero-pronoun resolution. As shown in Figure 1, the rewritten utterance can be used to generate cross-task supervision to the arguments of predicate \( \text{like} \) from the CSRL parser, while \( \text{like} \) to the predicted arg-0.

![Figure 1](image.png)

Figure 1: An example of cross-task supervision between a CSRL parser and a DR system in a dialogue. 久石让 \( \text{(Joe Hisaishi)} \) from the rewritten utterance provides cross-task supervision to 宮崎骏 \( \text{(Hayao Miyazaki)} \), the predicted arg-1 of \( \text{like} \), from the CSRL parser, while \( \text{like} \) to the predicted arg-0.
are proposed: (1) a translation matcher, which maps the pseudo-labels of different tasks for one instance into the same space and computes a matching score representing the cross-task matching degree of pseudo-labels from different tasks; (2) an augmented (instance) selector, which leverages both the confidence of pseudo-labels from task-specific models and the matching score to select instances with pseudo-labels of high quality as new training data. We choose CSRL and DR as friend tasks to conduct a case study for friend-training, and specify the translation matcher and augmented selector for friend-training between these tasks. Experimental results of domain generalization and few-shot learning show friend-training surpasses both classical and state-of-the-art semi-supervised learning algorithms by a large margin. To summarize, contributions from this work include:

- We propose friend-training, the first cross-task self-training framework which exploits supervision from friend tasks for better selection of pseudo-labels in the iterative training process.
- We provide specific modeling of friend-training between CSRL and DR, with a novel translation matcher and a novel augmented selector.
- Extensive experiments with CSRL and DR demonstrate the effectiveness of friend-training, outperforming several strong baselines.

2 Related Work

Self-training Self-training (Scudder, 1965; Angluin and Laird, 1988; Abney, 2002; Lee et al., 2013) is a classical semi-supervised learning framework (Chapelle et al., 2009) which has been widely explored in recent years. The general idea of self-training is to adopt a trained model to pseudo-label easily acquired unlabeled data and use them to augment the training data to retrain the model iteratively. This paradigm shows promising effectiveness in a variety of tasks: including text classification (Mukherjee and Awadallah, 2020; Wang et al., 2020a), image classification (Xie et al., 2020; Zoph et al., 2020), machine translation (He et al., 2020) and model distillation (Mukherjee and Hassan Awadallah, 2020). Co-training (Blum and Mitchell, 1998) and tri-training (Zhou and Li, 2005) are similar iterative training frameworks to self-training but with a different number of models or considering different views of the training data, both of which see wide adoption in NLP (Mihalcea, 2004; McClosky et al., 2006; Wan, 2009; Li et al., 2014; Caragea et al., 2015; Lee and Chieu, 2021; Wagner and Foster, 2021). These frameworks aim at improving performance with multiple models trained on one task, without directly leveraging the benefit of supervision from related tasks.

Multi-task Learning Multi-task learning (Caruana, 1997; Yang et al., 2021) seeks to improve the learning performance of one task with the help of other related tasks, among which two lines of work are related to ours: (1) semi-supervised multi-task learning (Liu et al., 2007; Li et al., 2009) combines semi-supervised learning and multi-task learning. Liu et al. (2007) exploited unlabeled data by random walk and used a task clustering method for multi-task learning. Li et al. (2009) integrated active learning (MacKay, 1992) with the model in Liu et al. (2007) to retrieve data that are most informative for labeling. Although these works tried to utilize unlabeled data to enhance multi-task learning, our work differs from them in incorporating supervised signals among tasks to select high-quality pseudo-labels for updating models, which is an iterative training process without additional human annotation. (2) Task grouping (Kumar and III, 2012; Standley et al., 2020; Fifty et al., 2021) aims to find groups of related tasks and employs multi-task learning to each group of tasks, with one model for each group. Our work focuses on training single-task models, but task grouping techniques can be used to look for possible friend tasks.

Conversational Semantic Role Labeling CSRL is a task for predicting the semantic roles of predicates in a conversational context. Wu et al. (2021) leveraged relational graph neural networks (Schlichtkrull et al., 2018) to model both the speaker and predicate dependency, achieving some promising results. However, the current dataset (Xu et al., 2021) for CSRL is limited to mono-domain. High-quality labeled data for new domains are needed to empower more applicable CSRL models.

Dialogue Rewriting DR is commonly framed as a sequence-to-sequence problem which suffers large search space issue (Elgohary et al., 2019; Huang et al., 2021). To address it, Hao et al. (2021) cast DR to sequence labeling, transforming rewriting an utterance as deleting tokens from an utterance or inserting spans from the dialogue history into an utterance. Jin et al. (2022) improved the continuous span issue in (Hao et al., 2021) by first generating multiple spans for each token and slotted rules and
then replacing a fixed number rules with spans.

3 Friend-training

Friend-training is an iterative training framework to jointly refine models of several friend tasks. Different from self-training, friend-training injects cross-task supervision into the selection of pseudo-labels. We first briefly describe self-training before presenting friend-training.

3.1 Self-training

Classic self-training aims at iteratively refining a model of a single task by using both labeled data and a large amount of unlabeled corpus. At each iteration, the model first assigns the unlabeled data with pseudo-labels. Subsequently, a set of the unlabeled instances with pseudo-labels is selected for training, presumably with information for model generalization. Then cross-entropy of model predictions and labels on both gold and pseudo-labeled data is minimized to update the model:

\[ L = \sum_{i=1}^{N} y_i \log \frac{y_i}{p_i} + \lambda \sum_{i=1}^{N'} y'_i \log \frac{y'_i}{p'_i}, \]

where the left term is the loss for the labeled data and the right for the unlabeled data while \( \lambda \) is a coefficient to balancing them; \( N(N') \) is the number of instances, \( y (y') \) is the label and \( p (p') \) is the output probability of the model.

Self-training is usually limited to only one task, but there are thousands of NLP tasks already proposed and many of them are related. Models trained for one task can be great teachers for other related tasks. We explore this cross-task supervision in self-training by incorporating two novel modules introduced in subsection 3.2.

3.2 Friend-training

For friend-training with two tasks,\(^1\) we have two classifiers \( f_a \) and \( f_b \) trained on two different tasks with labeled training sets \( \mathcal{L}_a \) and \( \mathcal{L}_b \), with expected accuracies \( \eta_a \) and \( \eta_b \), respectively. The two datasets are created independently and the prediction targets of the two tasks are partially related through a pair of translation functions \( \mathcal{F}_a : \hat{Y}_a \rightarrow \Sigma \) and \( \mathcal{F}_b : \hat{Y}_b \rightarrow \Sigma \), where \( \Sigma \) is the set of possible sub-predictions that all possible predictions of the two tasks \( \hat{Y}_a \) and \( \hat{Y}_b \) can be reduced to. \( |\hat{Y}_a| \geq |\Sigma|, |\hat{Y}_b| \geq |\Sigma| \). We assume that the translation functions are general functions with the expected probability of generating a translation \( \epsilon_{\mathcal{F}} = \frac{1}{|\Sigma|} \). The translation functions are deterministic and always map the gold labels of the friend tasks for the same input to the same translation.

Both classifiers make predictions on the unlabeled set \( \mathcal{U} \) at iteration \( k \). Some instances \( \mathcal{U}^k \) with pseudo-labels are chosen as new training data based on the results of the translation functions, \( \phi_a(x) = \mathcal{F}_a(f_a(x)) \) and \( \phi_b(x) = \mathcal{F}_b(f_b(x)) \), and some selection criteria, such as total agreement. If total agreement is used as the selection criterion, the probability of erroneous predictions for \( f_a \) in these instances is

\[
\Pr_x[f_a(x) \neq f^*_a(x)|\phi_a(x) = \phi(b(x))]
\]

\[
= 1 - \frac{\eta_a \Pr_x[\phi_a(x) = \phi_b(x)|f_a(x) = f^*_a(x)]}{\Pr_x[\phi_a(x) = \phi_b(x)]},
\]

with \( f^* \) being the optimal classifier.

Because both classifiers are very different due to training data, annotation guidelines, models, prediction targets, etc., being all different, the two classifiers are very likely to be independent of each other. Under this condition Equation 2 becomes

\[
1 - \frac{\eta_a(\eta_b + \epsilon_{\mathcal{F}}(1 - \eta_b))}{\Pr_x[\phi_a(x) = \phi_b(x)]}
\]

\[
= 1 - \frac{Z}{Z + \eta_b \epsilon_{\mathcal{F}}(1 - \eta_b) + E},
\]

where \( Z = \eta_a(\eta_b + \epsilon_{\mathcal{F}}(1 - \eta_b)) \) and \( E = \epsilon_{\mathcal{F}}^2(1 - \eta_a)(1 - \eta_b) \). We give the detailed derivation of Equation 2 and 3 in Appendix A.1. This indicates that the quality of the picked instances is negatively correlated with the number of false positive instances brought by the noisy translation \( \eta_b \epsilon_{\mathcal{F}}(1 - \eta_a) \), and the number of matching negative instances \( E \). When \( \epsilon_{\mathcal{F}} \) is minimized by choosing translation functions with a sufficiently large co-domain \( \Sigma \), the probability of error instances chosen when two classifiers agree approaches 0. This also indicates that even when \( 1 - \eta_a \) is large, i.e. \( f_a \) performs badly, if the co-domain is large, the error rate of the chosen instances can still be kept very low.\(^2\) As the dependence between the

\(^1\)We focus on the two-friend version of friend-training in this work, however, friend-training can easily be extended to more than two friends.

\(^2\)Intuitively, this means independent classifiers trained to do different tasks are unlikely to predict the same but wrong sub-prediction for a given input, if the sub-prediction includes a large number of decisions.
two classifiers grows in training, the probability of error instances also increases. When they are completely dependent on each other, Equation 2 becomes $1 - \eta_a$, i.e. classic self-training.

Based on this formulation, two additional modules are needed: (1) a translation matcher that maps predictions of two models trained on different tasks into the same space and computes a matching score; (2) an augmented (instance) selector which selects instances with pseudo-labels for the classifiers considering both the matching score of the translated predictions and the model confidences.

**Translation Matcher** Given the prediction of models of two friend tasks $f_a(x)$ and $f_b(x)$, the translation matcher $\mathcal{M}$ leverages translation functions $\mathcal{F}_a$ and $\mathcal{F}_b$ to get the translated pseudo-labels and computes a matching score $m$ for the pair of pseudo-labels, which represents the similarity of the pair in the translation space:

$$m_{a,b} = \mathcal{M}(\mathcal{F}_a(f_a(x)), \mathcal{F}_b(f_b(x))),$$

with total agreement being 1. This matching score serves as a criterion for the selection of high quality pseudo-labels with cross-task supervision.

**Augmented Selector** Apart from pseudo-label similarity, other information about pseudo-label quality can be found from model confidence, which self-training algorithms specifically utilize, to augment matching scores. The augmented selector considers both the confidence of the pseudo-labels from task models, denoted as $\{c_a, c_b\}$, and the matching scores:

$$q_\tau = S_\tau(m_{a,b}, c_\tau),$$

where $q_\tau \in \{0, 1\}$ represents the selection result of the pseudo-label for task $\tau \in a, b$. Therefore, instances with low matching scores but high confidence may also be selected as the training data. The complete algorithm is shown in Algorithm 1.

### 4 Friend Training between CSRL and DR

To verify the effectiveness of friend-training, we select two dialogue understanding tasks as friend tasks to conduct friend-training experiments for a case study: conversational semantic role labeling (CSRL) and dialogue rewriting (DR). While both require skills such as coreference and zero-pronoun resolution, the two tasks focus on different properties of the dialogue utterance: (1) CSRL focuses on extracting arguments of the predicates in the utterance from the whole dialogue history; (2) DR aims to rewrite the last turn of a dialogue to make it context-free and fluent by recovering all the ellipsis and coreference in the utterance. Figure 2 provides an overview of friend-training between the above two tasks. Next, we first introduce the task models and then specify the translation matcher and augmented selector for applying friend-training.

#### 4.1 Task Models

**Task Definition** A dialogue consists of $N$ temporally ordered utterances $\{u_1, ..., u_N\}$. (1) Given utterance $u_t$ and $K$ predicates $\{\text{pred}_1, ..., \text{pred}_K\}$ of $u_t$, a CSRL parser predicts spans from the dialogue as arguments for all predicates. (2) A dialogue rewriter rewrites $u_t$ to make it context-free according to its context $\{u_1, ..., u_{t-1}\}$.

**Dialogue Encoder** We concatenate dialogue context $\{u_1, ..., u_{t-1}\}$ and the current utterance $u_t$ as a sequence of tokens $\{x_1, ..., x_M\}$ and encode it with BERT (Devlin et al., 2019) to get the contextualized embeddings:

$$E = e_1, ..., e_M = \text{BERT}(x_1, ..., x_M) \in \mathbb{R}^{H \times M}.$$  

Encoders for CSRL and DR share no parameters, but for simplicity, we use the same notation $E$ for their outputs.

**Conversational Semantic Role Labeling** With the contextualized embeddings, we further generate predicate-aware utterance representations $G = \{g_1, ..., g_M\} \in \mathbb{R}^{H \times M}$ as Wu et al. (2021) by applying self-attention (Vaswani et al., 2017) to $E$.
with predicate-aware masking, where a token is only allowed to attend to tokens in the same utterance and tokens from the utterance containing the predicate:

\[
\text{Mask}_{i,j} = \begin{cases} 
1 & \text{if } u_{[i]} = u_{[j]} \text{ or } u_{[j]} = u_{[\text{pred}]}, \\
0 & \text{otherwise},
\end{cases}
\]

where \(u_{[m]}\) denotes the utterance containing token \(x_m\) and \(u_{[\text{pred}]}\) denotes the one with the predicate.

The predicate-aware representations are then projected by a feed-forward network to get the distribution of labels for each token:

\[
P^c = \text{softmax}_{\text{column-wise}}(W_c G + b_c) \in \mathbb{R}^{C \times M},
\]

where \(W_c\) and \(b_c\) are learnable parameters and \(C\) is the number of labels. The labels follow the BIO sequence labeling scheme: B-X and I-X respectively denote the token is the first token and the inner token of argument X, where O means the token does not belong to any argument. The output of the CSRL parser for \(K\) predicates are denoted as \(\{A_1, \ldots, A_K\}\), where set \(A_k\) containing the arguments for \(\text{pred}_k\).

**Dialogue Rewriting** Following Hao et al. (2021), we cast DR as sequence labeling. Specifically, a binary classifier on the top of \(E\) first determines whether to keep each token for utterance \(u_t\) in the rewritten utterance:

\[
P^d = \text{softmax}_{\text{column-wise}}(W_d E + b_d) \in \mathbb{R}^{2 \times M},
\]

where \(W_d\) and \(b_d\) are learnable parameters. Next, a span of the context tokens is predicted to be inserted in front of each token. In practice, two self-attention layer (Vaswani et al., 2017) are adopted to calculate the probability of context tokens being the start index or end index of the span:

\[
P^{st} = \text{softmax}_{\text{column-wise}}(\text{Attn}_{st}(E)) \in \mathbb{R}^{M \times M},
\]

\[
P^{ed} = \text{softmax}_{\text{column-wise}}(\text{Attn}_{ed}(E)) \in \mathbb{R}^{M \times M},
\]

where \(P_{i,j}^{st}\) (\(P_{i,j}^{ed}\)) denotes the probability of \(x_i\) being the start (end) index of the span for \(x_j\). Then by applying \(\text{argmax}\) to \(P\), we could obtain the start and end indexes of the span for each token:

\[
s^{st} = \text{argmax}_{\text{column-wise}}(P^{st}) \in \mathbb{R}^M, \\
s^{ed} = \text{argmax}_{\text{column-wise}}(P^{ed}) \in \mathbb{R}^M,
\]

The probability of the span to be inserted in front of \(x_m\) is \(P^{st}_{s^{st}_m, m} \times P^{ed}_{s^{ed}_m, m}\) when \(s^{st}_m \leq s^{ed}_m\). When \(s^{st}_m > s^{ed}_m\), it means no insertion. The output of the dialogue rewriter for \(u_t\) is denoted as \(u'_t\).

**4.2 Translation Matcher**

To translate the outputs (pseudo-labels) from the CSRL parser \(\{A_1, \ldots, A_K\}\) and the dialogue rewriter \(u'_t\) into a same space, we leverage a normal sentence-level semantic role parser with fixed parameters to greedily extract arguments from the rewritten utterance \(u'_t\) for the \(K\) predicates, denoted as \(\{B_1, \ldots, B_K\}\) (Appendix A.5 shows an example).
So the common target space $\Sigma$ is the label space of CSRL, which is large enough to make the error rate of chosen instances keep very low (see the analysis in subsection 3.2). The matching score $m_k \in [0, 1]$ for $\text{pred}_k$ is calculated based on the edit distance between $A_k$ and $B_k$:

$$m_k = 1 - \frac{\text{dist}(\oplus A_k, \oplus B_k)}{\max(\text{len}(\oplus A_k), \text{len}(\oplus B_k))},$$

where $\text{dist}()$ calculates the edit distance between two strings, $\text{len}()$ returns the length of a string and $\oplus A_k$ denotes the concatenation of arguments in set $A_k$ in a predefined order of arguments (empty strings means arguments do not exist). Furthermore, we obtain the overall matching score $m^' \in [0, 1]$ for the rewritten utterance $u^'_t$ as follows:

$$m^' = \text{GM}(m_1, \ldots, m_K),$$

where $\text{GM}()$ represents the geometric mean.

### 4.3 Augmented Selector

The augmented selector selects high-quality pseudo-labels according to both their matching scores and confidence. For CSRL, we calculate the confidence score for each predicate based on the output of the softmax layer. Specifically, we obtain the confidence of an argument for $\text{pred}_k$ by multiplying the probability of its tokens, denoted as $\{a_{k1}, \ldots, a_{k|A_k|}\}$. We then use the geometric mean of all the confidence of arguments belonging to $\text{pred}_k$ as the confidence for $\text{pred}_k$. The overall score $s_k \in [0, 1]$ for $\text{pred}_k$ is calculated as follows:

$$s_k = \alpha \text{GM}(a_{k1}, \ldots, a_{k|A_k|}) + (1 - \alpha)m_k,$$

where hyper-parameter $\alpha$ gives a balance between the matching score and the confidence. For DR, we multiply the probabilities of spans to be inserted and of decisions on whether to keep tokens or not as the model confidence of $u^'_t$, denoted by $b_t$. The overall score $r_t \in [0, 1]$ of $u^'_t$ is as follows:

$$r_t = \beta b_t + (1 - \beta)m^',$$

where a larger value of hyper-parameter $\beta$ places more importance on the model confidence. $\alpha$ and $\beta$ are set to be 0.2 for both tasks in the experiments.

Pick thresholds are set for $s_k$ and $r_t$ to control the number and quality of selected pseudo-labels. We analyze the effects of different values of thresholds in subsection 5.4.

### 5 Experiments

#### 5.1 Setup

**Datasets** We use five dialogue datasets in our experiments with domains spanning movies, celebrities, book reviews, products, and social networks. For CSRL, we use DuConv (Xu et al., 2021) and WeiboCSRL and for DR, REWRITE (Su et al., 2019) and RESTORATION (Pan et al., 2019). The datasets of the same task differ in domains and sizes. WeiboCSRL is a newly annotated CSRL dataset for out-of-domain testing purposes. Moreover, we use LCCC-base (Wang et al., 2020b) as the unlabeled corpus, which is a large-scale Chinese conversation dataset with 79M rigorously cleaned dialogues from various social media. More details on the annotation of WeiboCSRL and the properties of the datasets could be found in Appendix A.2.

**Experiment Scenarios** Our main experiments involve two scenarios. (1) Domain generalization: we use DuConv as the training data in the source domain and WeiboCSRL for out-of-domain evaluation, while for DR, REWRITE is used for training and RESTORATION for evaluation. (2) Few-shot learning: we randomly select 100 cases from DuConv and REWRITE as the training data for CSRL and DR, respectively, and conduct in-domain evaluation, which means models of both the tasks are co-trained with only a few samples of each task. The unlabeled data for both scenarios are 20k dialogues extracted from LCCC-base. Implementation details are provided in Appendix A.3.

**Evaluation** We follow Wu et al. (2021) to report precision (Pre.), recall (Rec.), and F1 of the arguments for CSRL and Hao et al. (2021) to report word error rate (WER) (Morris et al., 2004), Rouge L (R-L) (Lin, 2004) and the percent of sentence-level exact match (EM) for DR.
### Table 1: Test results for domain generalization and few-shot learning.

<table>
<thead>
<tr>
<th>Method</th>
<th>Pre.</th>
<th>Rec.</th>
<th>F1</th>
<th>R-L</th>
<th>EM</th>
<th>WER(↓)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>WeiboCSRL RESTORATION</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Base</td>
<td>57.97</td>
<td>54.47</td>
<td>56.16</td>
<td>82.78</td>
<td>25.25</td>
<td>28.69</td>
</tr>
<tr>
<td>Multitask-Base</td>
<td>53.66</td>
<td>54.32</td>
<td>53.99</td>
<td>81.68</td>
<td>22.49</td>
<td>32.44</td>
</tr>
<tr>
<td>SST (Scudder, 1965)</td>
<td>60.85</td>
<td>56.54</td>
<td>58.62</td>
<td>85.22</td>
<td>32.97</td>
<td>22.22</td>
</tr>
<tr>
<td>MT (Tarvainen and Valpola, 2017)</td>
<td>58.42</td>
<td>55.71</td>
<td>57.03</td>
<td>83.76</td>
<td>28.82</td>
<td>26.49</td>
</tr>
<tr>
<td>CPS (Chen et al., 2021)</td>
<td>60.34</td>
<td>52.87</td>
<td>56.36</td>
<td>85.60</td>
<td>32.68</td>
<td>22.78</td>
</tr>
<tr>
<td>SCoT (Blum and Mitchell, 1998)</td>
<td>57.33</td>
<td>54.13</td>
<td>55.69</td>
<td>84.51</td>
<td>29.25</td>
<td>24.87</td>
</tr>
<tr>
<td>STBR (Bhat et al., 2021)</td>
<td>60.77</td>
<td>58.04</td>
<td>59.38</td>
<td>85.79</td>
<td>33.78</td>
<td>23.30</td>
</tr>
<tr>
<td>STea (Yu et al., 2021)</td>
<td>60.10</td>
<td>55.13</td>
<td>57.50</td>
<td>85.75</td>
<td>34.23</td>
<td>22.17</td>
</tr>
<tr>
<td><strong>FDT (Ours)</strong></td>
<td>65.29(↑4.44)</td>
<td>58.63(↑2.09)</td>
<td>61.78(↑3.16)</td>
<td>86.82(↑1.60)</td>
<td>38.22(↑5.25)</td>
<td>20.31(↑1.91)</td>
</tr>
</tbody>
</table>

(a) Domain generalization for models trained with DuConv and REWRITE.

<table>
<thead>
<tr>
<th>Method</th>
<th>Pre.</th>
<th>Rec.</th>
<th>F1</th>
<th>R-L</th>
<th>EM</th>
<th>WER(↓)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DuConv</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Base</td>
<td>29.50</td>
<td>21.90</td>
<td>25.14</td>
<td>73.44</td>
<td>3.60</td>
<td>39.98</td>
</tr>
<tr>
<td>Multitask-Base</td>
<td>22.43</td>
<td>20.63</td>
<td>21.49</td>
<td>78.97</td>
<td>11.70</td>
<td>40.46</td>
</tr>
<tr>
<td>SST (Scudder, 1965)</td>
<td>34.16</td>
<td>27.49</td>
<td>30.62</td>
<td>80.93</td>
<td>27.80</td>
<td>31.02</td>
</tr>
<tr>
<td>MT (Tarvainen and Valpola, 2017)</td>
<td>36.32</td>
<td>30.69</td>
<td>33.27</td>
<td>81.66</td>
<td>33.00</td>
<td>31.66</td>
</tr>
<tr>
<td>CPS (Chen et al., 2021)</td>
<td>37.14</td>
<td>29.47</td>
<td>32.86</td>
<td>79.56</td>
<td>23.30</td>
<td>32.60</td>
</tr>
<tr>
<td>SCoT (Blum and Mitchell, 1998)</td>
<td>38.37</td>
<td>26.15</td>
<td>31.10</td>
<td>78.58</td>
<td>22.31</td>
<td>33.79</td>
</tr>
<tr>
<td>STBR (Bhat et al., 2021)</td>
<td>32.37</td>
<td>25.21</td>
<td>28.34</td>
<td>82.37</td>
<td>29.80</td>
<td>30.31</td>
</tr>
<tr>
<td>STea (Yu et al., 2021)</td>
<td>39.34</td>
<td>28.78</td>
<td>33.25</td>
<td>83.04</td>
<td>31.57</td>
<td>30.36</td>
</tr>
<tr>
<td><strong>FDT (Ours)</strong></td>
<td>40.41(↑6.25)</td>
<td>30.82(↑3.33)</td>
<td>34.97(↑4.51)</td>
<td>82.83(↑1.90)</td>
<td>34.20(↑6.40)</td>
<td>27.87(↑3.15)</td>
</tr>
<tr>
<td><strong>FDT-S (Ours)</strong></td>
<td>40.12</td>
<td>33.41</td>
<td>36.46</td>
<td>83.11</td>
<td>37.10</td>
<td>26.88</td>
</tr>
</tbody>
</table>

(b) Few-shot learning for models trained with DuConv and REWRITE.

Table 1: Test results for domain generalization and few-shot learning. Base denotes the task models trained with data from a single task. Multitask-Base denotes the base model of CSRL and DR sharing the same dialogue encoder. Results are averaged across three runs. ↓ means lower is better. For few-shot learning, performance of the base models trained with the full training set from the single task is provided for reference.

### 5.3 Main Results

Table 1 shows the comparison between friend-training (FDT) and the baselines mentioned in subsection 5.2. FDT achieves the best overall performance over the baselines by significant margins in both domain generalization and few-shot learning scenarios, which demonstrates the effectiveness of FDT in different experimental situations to utilize large unlabeled corpora. Moreover, we show the absolute improvements of FDT over SST in parentheses (↑). As we could see, in few-shot learning, FDT obtain 4.51 and 3.15 higher absolute points over SST on F1 of DuConv and WER of REWRITE, respectively, than those of domain generalization, which are 3.16 and 1.91 points, revealing that FDT could realize its potential easier in few-shot learning. Besides, for few-shot learning, we further consider the situation where a fully-trained base model from the friend task is available, denoted as FDT-S. As we could see, when the target task is CSRL, FDT-S makes a gain of 1.49 points on F1 over FDT and when the target task is DR, FDT-S outperforms FDT on WER by 0.99 points and EM by 2.90 points, indicating that more reliable supervision from friend task could further enhance the few-shot learning of the target task.

### 5.4 Analysis

In this section, we conduct experiments to analyze how selected parameters and settings interact with model performance in FDT.

**Pick Thresholds** We vary the pick thresholds of CSRL and DR in domain generalization scenario and track the model performance: we fix the pick threshold of the friend task to the best (see Appendix A.3) when varying that of the evaluating task. As illustrated in Figure 3a, when the thresholds increase gradually, the models become better with higher F1 for CSRL and lower WER for DR. We attribute this to wrong pseudo-labels being filtered out by the augmented selector of FDT. Then the model performances hit the peaks and drop as the thresholds keep increasing in the interval of high values, which is owed to high thresholds producing insufficient pseudo-labels for iterative train-
(a) The effect of pick thresholds.
(b) F1 on CSRL test set.
(c) WER on DR test set.

Figure 3: Sub-figures (b) and (c) show the model performance of the comparing methods with different strengths of base models; the dashed horizontal line represents the performance of FDT with a fully trained base model.

The Role of Co-updating We also explore the case where one of the models of the friend tasks is fully trained and does not have to be updated. We consider FDT-SF, FDT with a fixed fully trained base model from the friend task in domain generalization. As illustrated in Figure 4, FDT-SF surpasses FDT when given a weak base model for the evaluating task because of the strong supervision from the friend task. However, FDT outperforms FDT-SF when the evaluating task is given a fairly-trained model, which demonstrates the benefits of co-updating the models in friend-training.

6 Conclusion

We propose friend-training, the first cross-task self-training framework, which leverages supervision from friend tasks for better selection of pseudo-labels. Moreover, we provide specific modeling of friend-training between conversational semantic role labeling and dialogue rewriting. Experiments on domain generalization and few-shot learning scenarios demonstrate the promise of friend-training, which outperforms prior classical or state-of-the-art semi-supervised methods by substantial margins.

4Specifically, when the evaluating task is CSRL, the amount of labeled data for the two tasks are set as {10%/100%, 30%/100%, 50%/100%, 70%/100%, 90%/100%}, and when the evaluating task is DR, {10%/100%, 100%/30%, 100%/50%, 100%/70%, 100%/90%}.
Limitations

We showed how the friend-training strategy can be applied to two dialogue understanding tasks in the case study here, but many other task pairs or task sets can be examined to fully explore the generality of the approach. Identifying friend tasks depends on expert knowledge in this work, but approaches for task grouping and task similarity may be used to automatically discover friend tasks. Besides, with the proliferation of cross-modal techniques, tasks of different modalities are expected to act as friend tasks as well. Also, designing translation functions and matchers for friend tasks in the friend-training framework requires an understanding of the relationship between the friend tasks, but prompting and model interpretability methods could potentially be applied for easing this process.

Acknowledgement

We thank the anonymous reviewers for their helpful comments and the support of National Nature Science Foundation of China (No.62176174).

References


Xiao Liu, Fanjin Zhang, Zhenyu Hou, Li Mian, Zhaoyu Wang, Jing Zhang, and Jie Tang. 2021. Self-supervised learning: Generative or contrastive. IEEE Transactions on Knowledge and Data Engineering.


Zhuofeng Pan, Kun Bai, Yan Wang, Lianqiang Zhou, and Xiaojiang Liu. 2019. Improving open-domain dialogue systems via multi-turn incomplete utterance


A Appendix

A.1 Error rates

We have two classifiers $f_a$ and $f_b$ trained on two different tasks with labeled training sets $L_a$ and $L_b$, with expected accuracies $\eta_a$ and $\eta_b$, respectively. The prediction targets of the two tasks are partially related through a pair of translation functions $F_a : Y_a \rightarrow \Sigma$ and $F_b : Y_b \rightarrow \Sigma$, where $\Sigma$ is the set of possible sub-predictions that all possible predictions of the two tasks $Y_a$ and $Y_b$ can be reduced to. $|Y_a| \geq |\Sigma|, |Y_b| \geq |\Sigma|$. The sub-predictions can be a part of the whole prediction targets for both tasks, or some lossy transformation of the prediction targets. For example, a sub-prediction for a POS-tagging task can be the POS tag of the first word (a part of the prediction) or the number of the NN tag in the whole prediction sequence (a transformation of the prediction). We assume that the translation functions are general functions with the expected probability of generating a translation $\epsilon_F = \frac{1}{M_f}$; they are deterministic and always map the gold labels of the friend tasks for the same input to the same translation. Both classifiers make predictions on the unlabeled set $U$ at iteration $k$. Some instances $U^\phi_k$ with pseudo-labels are chosen as new training data based on the results of the translation functions, $\phi_a(x) = F_a(f_a(x))$ and $\phi_b(x) = F_b(f_b(x))$, and some selection criteria, such as total agreement. If total agreement is used as the selection criterion, the probability of erroneous predictions for $f_a$ in these instances is

\[
\Pr_x[f_a(x) \neq f^*_a(x)|\phi_a(x) = \phi_b(x)] = 1 - \Pr_x[f_a(x) = f^*_a(x)|\phi_a(x) = \phi_b(x)]
\]

\[
= 1 - \Pr_x[f_a(x) = f^*_a(x)] \cdot \frac{\Pr_x[\phi_a(x) = \phi_b(x)]}{\Pr_x[\phi_a(x) = \phi_b(x)]} = 1 - \eta_a \cdot \frac{\Pr_x[\phi_a(x) = \phi_b(x)]}{\Pr_x[\phi_a(x) = \phi_b(x)]},
\]

with $f^*$ being the optimal classifier. If we consider the two classifiers very likely to be independent from each other, then the probability of the translation of the predictions from the two classifiers being the same given the prediction from classifier $f_a$ is correct, which is $\Pr_x[\phi_a(x) = \phi_b(x)|f_a(x) = f^*_a(x)]$, is the sum of the probability of the classifier $f_b$ making the correct prediction $\eta_b$ and the probability of an erroneous translation of the wrong prediction $\epsilon_F(1 - \eta_b)$. The probability of the translations matching $\Pr_x[\phi_a(x) = \phi_b(x)]$ has four situations: both predictions of the two classifiers are correct $\eta_a\eta_b$; $f_a(x)$ is correct but $f_b(x)$ is wrong and being translated erroneously $\eta_b\epsilon_F(1 - \eta_a)$; $f_b(x)$ is correct but $f_a(x)$ is wrong and being translated erroneously $\eta_a\epsilon_F(1 - \eta_b)$; both $f_a(x)$ and $f_b(x)$ are wrong but matching in the translation space $\epsilon_F^2(1 - \eta_a)(1 - \eta_b)$. Under these conditions Equation 6 becomes

\[
1 - \frac{\eta_a(\eta_b + \epsilon_F(1 - \eta_b))}{Z} = 1 - \frac{Z}{Z + \eta_b\epsilon_F(1 - \eta_a) + E},
\]

where $Z = \eta_a(\eta_b + \epsilon_F(1 - \eta_b))$ and $E = \epsilon_F^2(1 - \eta_a)(1 - \eta_b)$ which shows that the term $\eta_b\epsilon_F(1 - \eta_a) + E$ needs to be small to make the probability of matching translations with predictions being wrong small. This indicates that the quality of the picked instances based on the total agreement criterion is negatively correlated with the number of false positive instances brought by the noisy translation $\eta_b\epsilon_F(1 - \eta_a)$, and the number of matching negative instances $E$. $\epsilon_F$ can be minimized by choosing translation functions with a sufficiently large co-domain $\Sigma$, which means that when the translation space is large enough, it is unlikely that the two classifiers totally agree in the translation space but do not agree in their own prediction target spaces. So the probability of them agreeing and making correct predictions is much larger than agreeing but making incorrect predictions while the probability of error instances chosen when two classifiers agree approaches 0, indicating that even when $1 - \eta_a$ is large, i.e., $f_a$ performs badly, if the co-domain is large, the error rate of the chosen instances can still be kept very low.

A.2 Datasets

Annotation Procedure of WeiboCSRL The dialogues we use for CSRL annotation are extracted from LCCC-base \cite{wang2020cc}, which consists of at least 4 turns and 80 total characters to assure enough context for CSRL and DR. These dialogues and those used as unlabeled data for experiments in section 5 are from different parts of LCCC-base. For each dialogue, we annotate the predicates in the last utterance with the guidance of frame files of Chinese Propositional Logic.
tion Bank\(^5\). For each predicate, the arguments we annotate are numbered arguments ARG0-ARG4 and adjuncts ARGM-LOC, ARGM-MNR, ARGM-TMP and ARGM-NEG, whose definitions are shown in (Xue, 2006). ARGM-MNR is not included for evaluation in section 5 because annotation of ARGM-MNR is lacking in DuConv, the training data for CSRL. In the end, we obtain 3891 annotated predicates.

**Dataset Details**

Table 2 shows the statistic of the datasets used in the experiments. DuConv (Xu et al., 2021) focuses on movies and celebrities and we adopt the same train/dev/test splitting as Xu et al. (2021). REWRITE (Su et al., 2019) contains 20K dialogues with a wide range of topics crawled from Chinese social media platforms; the last utterance of each dialogue is rewritten to recover all co-referred and omitted information. RESTORATION (Pan et al., 2019) contains dialogues from Douban\(^6\), most of which are book, movie or product reviews. Compared with REWRITE, it contains more annotated dialogues, but around 40% of the last utterances require no rewriting.

<table>
<thead>
<tr>
<th>Domain</th>
<th>#Instance (train/dev/test)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DuConv</td>
<td>movies and celebrities: 23361 / 2852 / 2977</td>
</tr>
<tr>
<td>WeiboCSRL</td>
<td>social media: - / 1945 / 1946</td>
</tr>
<tr>
<td>REWRITE</td>
<td>social media: 16925 / 1000 / 1000</td>
</tr>
<tr>
<td>RESTORATION</td>
<td>book, movie and product reviews: - / 5000 / 5000</td>
</tr>
</tbody>
</table>

Table 2: Dataset statistics.

### A.3 Implementation Details

Dataset configuration of the tasks for the experimental scenarios are shown in Table 3.

**Preprocessing Details**

The maximum length of the input dialogue is set to 125. We transform the word-based labeling of DuConv to character-based labeling and we use the scripts\(^7\) provided by Hao et al. (2021) to generate token-level annotations for sequence-labeling-based DR. For unlabeled data, we discard dialogues with less than 4 turns to guarantee sufficient context for CSRL and DR.

**Model Details**

We use pretrained BERT\(^8\) (Devlin et al., 2019) as the dialogue encoder for CSRL and DR. Both the values of hyper-parameter \(\alpha\) and \(\beta\) are set to 0.2 and the pick thresholds are set to 0.6. We choose a state-of-the-art sentence-level semantic role labeling (SSRL) parser\(^9\) for the translation matcher which follows the same structure as (He and Choi, 2021).

**Training Details**

We adopt AdamW (Loshchilov and Hutter, 2019) to optimize models with a learning rate of 4e-5 and batch size of 16. We use \(\lambda = 1\) to balance the loss of labeled and unlabeled data.

<table>
<thead>
<tr>
<th>Task</th>
<th>Train</th>
<th>Dev/Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>DG</td>
<td>CSRL (train)</td>
<td>WeiboCSRL (dev/test)</td>
</tr>
<tr>
<td>DR</td>
<td>REWRITE (train)</td>
<td>RESTORATION (dev/test)</td>
</tr>
<tr>
<td>FSL</td>
<td>CSRL (100 cases)</td>
<td>DuConv (dev/test)</td>
</tr>
<tr>
<td>DR</td>
<td>REWRITE (100 cases)</td>
<td>REWRITE (dev/test)</td>
</tr>
</tbody>
</table>

Table 3: Dataset configuration of domain generalization (DG) and few-shot learning (FSL).

### A.4 Baselines

Standard self-training (Scudder, 1965) generates pseudo-labels to unlabeled data with a base model and uses them to train a new base model, which is repeated until convergence. Standard co-training (Blum and Mitchell, 1998) is similar to Standard self-training, but with two different base models dealing with the same task, generating pseudo-labels and adding the trusted ones for iterative training. Mean teacher (Tarvainen and Valpola, 2017) maintains a teacher model on the fly, whose weights are the exponential moving average of the weights of a student model across iterations. Cross pseudo supervision (Chen et al., 2021), a state-of-the-art variant of self-training, maintains two networks with different initialization; the pseudo-label of one network is used to supervise the other network. Self-training with batch reweighting (Bhat et al., 2021) is a state-of-the-art self-training method that reweights the pseudo-labels in a batch when training according to the confidence from the teacher model. Self-teaching (Yu et al., 2021), a state-of-the-art semi-supervised method that sequentially trains a junior teacher, a senior teacher and an expert student to leverage the unlabeled data.

For the hyper-parameters of the baselines, we keep the common hyper-parameters, such as learning rate, batch size, optimizer, and so on, the same as our proposed method. And we adopt the values of method-specific hyper-parameters used in the original papers, such as the merging weight of soft and hard labels of self-teaching and the smoothing parameter for updating of mean teacher.

---

5https://verbs.colorado.edu/chinese/cpb/
6https://www.douban.com
7https://github.com/freesunshine0316/RaST-plus
8https://huggingface.co/bert-base-chinese
9https://github.com/hankcs/HanLP
A.5 Case Study

We show a representative case of selecting pseudo-labels in Table 4. There are two predicates in current utterance: 是 (is) and 受 (attract). For 是 (is), the CSRL parser yields only ARG1 while SSRL parser gives the same ARG1 but more of ARG0 based on the rewritten utterance. With the difference in arguments, the overall score is not high and this predicate could be regarded as low-quality if a high pick threshold is set. For 受 (attract), the CSRL and SSRL parsers give the same arguments, which are the right answer. However, if we only consider the model confidence of the predicate, which is 0.54, this high-quality predicate are more likely to be discarded than consider the overall score, which is 0.90. And the rewritten utterance gets a high overall score, which is what we expected.

A.6 Discussion on Generalization of the Framework

It is not uncommon at all for different language tasks sharing some information. With one case study presented in detail in the main body of the paper, we also provide a short example of a different friend task pair – constituency parsing and dependency parsing – and explain how they can help each other and show the general nature of the friend-training framework.

Early work (Magerman, 1995; Collins, 2003) has shown relationship between dependency and constituency parsing through head-finding rules, and Jin and Schuler (2019) show directly how common structures between dependency and constituency trees can be derived for parsing evaluation. In a dependency graph, a set of nodes with a single incoming edge is usually indicative of a phrase structure, such as a noun phrase, a verb phrase or a prepositional phrase. Such phrasal structures are well-marked in constituency treebanks, and could be used as the shared friend information for friend-training. Here is a sketch of how friend-training can be applied to this pair:

1. Train a constituency parser and a dependency parser, presumably trained with a small number of training instances, as the models for the friend-training framework.
2. Run both parsers on a common set of unlabeled data for parsing results.
3. Find phrases such as noun, verb or prepositional phrases in the predicted constituency trees.
4. Compare with the dependency trees, and check if spans of such phrases have only a single incoming edge. If so, the constituency and dependency parsing results can be considered agreeing, and added to the silver training set. If not, the silver annotation is discarded.
5. Train the parsers again with the gold and silver training instances.

As long as some shared information can be identified between two seemingly different tasks, the
noisy agreement between that partial target can provide valuable supervision between two tasks. The translation and matching between constituency-dependency targets are simpler compared to the CSRL-rewriting pair presented in the paper, partly because no model is required for the translation process. However the CSRL-rewriting pair is more significant because heuristics may be difficult or not obvious to design where ‘bridging’ tasks such as single-sentence SRL may be readily available.