Translation-Based Matching Adversarial Network for Cross-Lingual Natural Language Inference

Kunxun Qi,[†,‡] **Jianfeng Du**[†*]
[†]Guangzhou Key Laboratory of Multilingual Intelligent Processing,
Guangdong University of Foreign Studies, Guangzhou 510420, China
[‡]Platform and Content Group, Tencent, Shenzhen 518000, China
kunxunqi@foxmail.com, jfdu@gdufs.edu.cn

Abstract

Cross-lingual natural language inference is a fundamental task in cross-lingual natural language understanding, widely addressed by neural models recently. Existing neural model based methods either align sentence embeddings between source and target languages, heavily relying on annotated parallel corpora, or exploit pre-trained cross-lingual language models that are fine-tuned on a single language and hard to transfer knowledge to another language. To resolve these limitations in existing methods, this paper proposes an adversarial training framework to enhance both pre-trained models and classical neural models for cross-lingual natural language inference. It trains on the union of data in the source language and data in the target language, learning language-invariant features to improve the inference performance. Experimental results on the XNLI benchmark demonstrate that three popular neural models enhanced by the proposed framework significantly outperform the original models.

Introduction

Cross-lingual language understanding (XLU) plays a crucial role in multilingual systems. In general XLU aims to train a model primarily on the data in one language and then apply it to natural language understanding in other languages. Natural language inference (NLI), also known as recognizing textual entailment (RTE), is one of the typical tasks in natural language understanding. It aims to determine the inferential relationship between two natural language sentences, one called the premise and the other called the hypothesis.

The importance of NLI in XLU has been identified in recent studies such as (Conneau et al. 2018). Cross-lingual natural language inference (XNLI) is considered as a better task for evaluating XLU than other tasks such as cross-lingual document classification, since XNLI is more challenging and has also large-scale benchmark data. By now NLI has been widely used in information retrieval (Clinchant, Goutte, and Gaussier 2006), question answering (Harabagiu and Hickl 2006), machine translation (Poliak et al. 2018) and other applications. Upgrading NLI to XNLI can adapt these applications to multilingual scenarios and improve the performance for low-resource languages. The challenges for achieving XNLI are two-fold. On one hand, knowledge transfer between two languages is difficult, usually relying on large parallel corpora (Hermann and Blunsom 2014; Conneau et al. 2018). On the other hand, XNLI needs to transfer not only information about individual sentences but also matching information between two sentences from the source language to the target language.

Neural models have become dominant in XNLI. Most of the neural model based methods such as (Conneau et al. 2018; Artetxe and Schwenk 2019) encode sentences in different languages into the same embedding space through parallel corpora. They hardly work for low-resource languages in which parallel corpora are rare. Recently more advanced methods such as multilingual BERT (Devlin et al. 2019) and XLM (Lample and Conneau 2019) have been proposed, which rely on pre-trained language models. These pre-training based methods learn a cross-lingual language model from a large multilingual corpus and then fine-tune the model on training data in one language. Although these methods do not rely on parallel corpora, they still have limitations since fine-tuning is performed in a single language. When the model is fine-tuned on the original training data in one language, matching information between two sentences can hardly be transferred to other languages. When the model is fined-tuned on the training data that have been translated to another language, sentence information on the original training data is ignored and translation errors can easily be propagated into the model.

Inspired by the promising results for adversarial training to domain adaption (Ganin et al. 2016; Chen et al. 2018; Wang and Pan 2018), we also adapt adversarial training to XNLI. We develop an adversarial training framework named TMAN (short for Translation-based Matching Adversarial Network) for XNLI. As shown in Figure 1, TMAN translates training data in the source language to the target language and merges these two sets of data. The merged set is then fed into an NLI model to generate matching representations between premises and hypotheses. To better capture language-invariant features, TMAN introduces a discriminator for classifying languages (source or target) to compete against the discriminator for classifying inferential relationships (*entailment*, *neutral* or *contradiction*). We employ TMAN to enhance two pre-trained language models, multilingual BERT (Devlin et al. 2019) and XLM (Lample and

*Jianfeng Du is the corresponding author.
Copyright © 2020, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

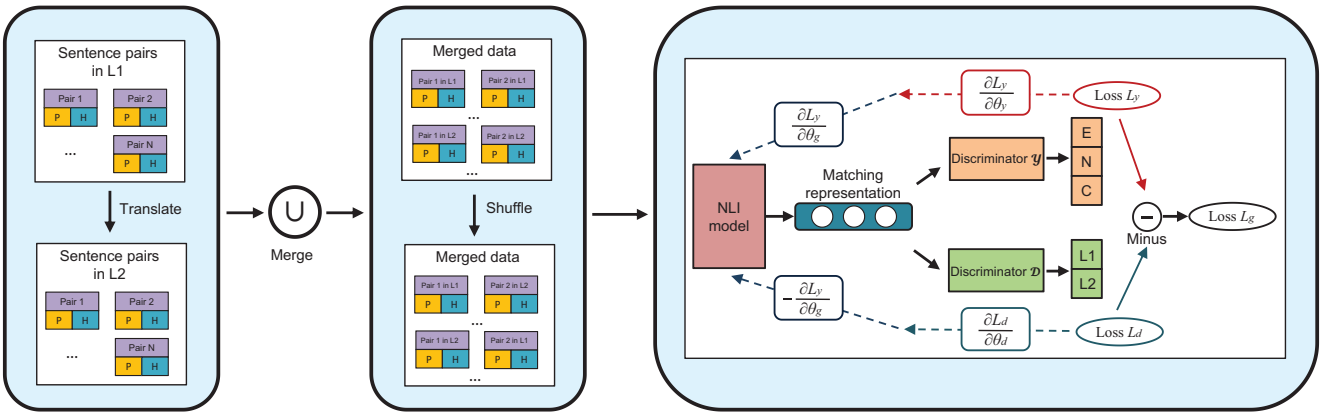


Figure 1: The proposed translation-based matching adversarial network (TMAN).

Conneau 2019), as well as a popular classical neural model ESIM (Chen et al. 2017). All these models are evaluated on the XNLI benchmark (Conneau et al. 2018) involving 15 languages. Experimental results show that, averaged by the 15 languages, TMAN pushes XLM by an absolute gain of 0.5% accuracy, BERT by an absolute gain of 1.7% accuracy, and ESIM by an absolute gain of 1.7% accuracy.

The main contributions of this work include:

(1) We propose an adversarial training framework for XNLI. As far as we know, this is the first adaptation of adversarial training to XNLI.

(2) We propose detailed methods for enhancing pre-trained language models and classical NLI models with the proposed framework.

(3) We conduct extensive experiments to verify the significant improvements achieved by the proposed framework.

Related Work

In recent years cross-lingual language understanding (XLU) is commonly achieved by aligning sentence embeddings for different languages in neural models. The study (Hermann and Blunsom 2014) proposes a compositional vector model for cross-lingual sentence classification. The studies (Schwenk and Douze 2017; España-Bonet et al. 2017; Johnson et al. 2017) aim to build cross-lingual sequence-to-sequence encoders for machine translation. The study (Conneau et al. 2018) proposes to align sentence embeddings based on a cross-lingual similarity defined by parallel data. The study (Artetxe and Schwenk 2019) proposes a language-invariant Bi-LSTM (Hochreiter and Schmidhuber 1997) encoder for 93 languages, which is trained on publicly available parallel corpora. Recently, in order to reduce the requirement of annotated parallel data, a new strategy for XLU is proposed. It often learns a cross-lingual language model from large-scale multilingual corpora and fine-tunes the model for downstream tasks. Typical methods using this strategy include multilingual BERT (Devlin et al. 2019) and XLM (Lample and Conneau 2019), where multilingual BERT extends the basic BERT by training with multilingual corpora. By introducing the translation language modeling (TLM) objective for improving cross-lingual pre-training, XLM has achieved the best accuracy on the XNLI benchmark (Conneau et al. 2018) prior to this work.

Adversarial networks (Goodfellow et al. 2014) have been widely used in domain adaption. The study (Ganin et al. 2016) proposes an adversarial network to learn domain-invariant features. The study (Chen et al. 2018) proposes a domain adversarial network for cross-lingual sentiment classification. The study (Wang and Pan 2018) proposes a transition-based adversarial network for cross-lingual aspect extraction. The study (Zou et al. 2018) proposes an adversarial network for feature adaption in cross-lingual relation classification. The study (Kang et al. 2018) proposes an adversarial network for NLI which introduces an adversarial example generator to compete against the NLI label discriminator. Compared with the above studies, this work adapts adversarial training to the XNLI problem in different ways. Firstly, we enlarge the training data set by translating the training data in the source language to the training data in the target language. Secondly, we randomly mix the data in the source language and the data in the target language in a training mini-batch, while existing studies use data in the same language to compose a training mini-batch. Finally, in order to better adapt a pre-trained language model to the XNLI problem, we also exploit a transformer encoder (Vaswani et al. 2017) to adjust the output of the pre-trained language model. Our experiments (see Table 1) confirm that the above alternations in our adversarial training framework lead to a higher accuracy in predicting NLI labels than ordinary ways.

The TMAN Framework

The architecture of TMAN is shown in Figure 1. Given a set of training tuples in the source language, TMAN translates the training set to another set of tuples in the target language and then merges these two sets to feed into an adversarial network composed by two components. The first component is an NLI model that generates the matching representation between a premise and a hypothesis. The second component consists of two discriminators that compete with each other, where one classifies the inferential relationship (namely the NLI label), and the other classifies the language used in the

matching representation.

Formalization of TMAN

The training phase of TMAN is formalized in Algorithm 1. For every training tuple (P_i, H_i, R_i, L_i), where P_i denotes the word sequence of the premise, H_i the word sequence of the hypothesis, R_i the index of the NLI label, and L_i the index of the language, TMAN employs an NLI model $\mathcal{G}$ to generate a matching representation between P_i and H_i, denoted by $\mathrm{G}(P_i, H_i)$. The NLI model is enhanced either by a pre-trained language model or by a classical NLI model, detailed in the next two subsections.

Inspired by the domain adversarial network (Ganin et al. 2016), TMAN constructs two discriminators to capture language-invariant features in matching representations. One discriminator $\mathcal{Y}$ is used to classify NLI labels, whereas the other discriminator $\mathcal{D}$ is used to classify languages. Let N_m denote the dimension of $\mathrm{G}(P_i, H_i)$ and N_r the number of NLI labels, then the output of $\mathcal{Y}$ and the output of $\mathcal{D}$ are respectively defined as

$$y_i = \mathrm{softmax}(W_y\,\mathrm{G}(P_i, H_i) + b_y) \quad (1)$$

$$d_i = \mathrm{softmax}(W_d\,\mathrm{G}(P_i, H_i) + b_d) \quad (2)$$

where $W_y \in \mathbb{R}^{N_r \times N_m}$ and $b_y \in \mathbb{R}^{N_r}$ are trainable parameters in $\mathcal{Y}$, and $W_d \in \mathbb{R}^{2 \times N_m}$ and $b_d \in \mathbb{R}^2$ are trainable parameters in $\mathcal{D}$.

In order to better capture language-invariant features in the matching representations, every mini-batch used in the training phase is required to consist of data from both the source language and the target language. In this way the implicit features related to different languages can probably be updated simultaneously and aligned into the same space. Given a mini-batch $(P_i, H_i, R_i, L_i)_{1 \le i \le N}$ of N tuples, the two discriminators $\mathcal{D}$ and $\mathcal{Y}$ are respectively trained by minimizing the cross-entropy losses $\mathcal{L}_d$ and $\mathcal{L}_y$:

$$\mathcal{L}_d = -\sum_{i=1}^{N}\sum_{j=1}^{2} I(j = L_i)\log d_{i,j} \quad (3)$$

$$\mathcal{L}_y = -\sum_{i=1}^{N}\sum_{j=1}^{N_r} I(j = R_i)\log y_{i,j} \quad (4)$$

where $d_{i,j}$ and $y_{i,j}$ are respectively the j^{th} element of d_i and y_i, and $I(C)$ returns 1 if C is true or 0 otherwise.

The model is trained by minimizing $\mathcal{L}_g = \mathcal{L}_y - \lambda\mathcal{L}_d$ for feature generation and $\mathcal{L}_d$ for language discrimination, where λ is the hyper-parameter for trade-off between the two losses. The sets of parameters θ_g for $\mathcal{G}$, θ_y for $\mathcal{Y}$ as well as θ_d for $\mathcal{D}$ can be found as either a stationary point or a point after a specified number of epochs of the following updates, where μ is the learning rate:

$$\theta_g = \theta_g - \mu\left(\frac{\partial\mathcal{L}_y}{\partial\theta_g} - \lambda\frac{\partial\mathcal{L}_d}{\partial\theta_g}\right) \quad (5)$$

$$\theta_y = \theta_y - \mu\frac{\partial\mathcal{L}_y}{\partial\theta_y} \quad (6)$$

$$\theta_d = \theta_d - \mu\left(\lambda\frac{\partial\mathcal{L}_d}{\partial\theta_d}\right) \quad (7)$$

Algorithm 1 The training phase of TMAN

Require: a translator from source language L1 to target language L2, a set of training tuples $\mathbb{S}$ in L1, and the number of epochs T.

```
Translate 𝕊 to a set of tuples 𝕋 in L2
Divide 𝕊∪𝕋 into a set of mini-batches 𝔻 where for each mini-batch 𝐁 ∈ 𝔻, 𝐁∩𝕊 ≠ ∅ and 𝐁∩𝕋 ≠ ∅
for epoch from 1 to T do
    Shuffle 𝔻
    for each mini-batch (P_i, H_i, R_i, L_i)_{1≤i≤N} in 𝔻 do
        Compute L_d and L_y by Eq. (3)–(4)
        Update θ_g, θ_y and θ_d by Eq. (5)–(7)
    end for
end for
```

During the test phase, for any test sentence pair, TMAN predicts an NLI label for it through the NLI model $\mathcal{G}$ and the discriminator $\mathcal{Y}$.

Enhancing Pre-trained Language Models

Pre-trained language models have been successfully applied to NLI (Radford et al. 2018; Devlin et al. 2019; Lample and Conneau 2019). To enhance these models for XNLI, we propose a framework shown in Figure 2.

Lexicon Encoder. The input of the enhanced model is a sentence pair (P, H), where $P = (w_1^p, ..., w_n^p)$ and $H = (w_1^h, ..., w_m^h)$ are two sequences of words. Following (Devlin et al. 2019), we create a new sequence by first concatenating P and H and then adding a special token [CLS] in front of the first token w_1^p, a special token [SEP] between w_n^p and w_1^h, and a [SEP] behind the last token w_m^h. This new sequence is fed into a lexical encoder, which can be a pre-trained language model such as BERT (Devlin et al. 2019) and XLM (Lample and Conneau 2019). By $X \in \mathbb{R}^{t \times d}$ we denote the output of the lexical encoder, where t is the number of tokens in the new sequence and d is the dimension of an arbitrary token embedding.

Transformer Encoder. Inspired by the work (Liu et al. 2019), we apply a transformer encoder (Vaswani et al. 2017) to X to obtain a more informative matching representation. The transformer encoder first computes a position embedding for every token in the new sequence.

$$P_{i,2j} = \sin(i/10000^{\frac{2j}{d}}) \quad (8)$$

$$P_{i,2j+1} = \cos(i/10000^{\frac{2j}{d}}) \quad (9)$$

where $i \in \{1, ..., t\}$ is the index of the token, and $2j \in \{1, ..., d\}$ and $2j + 1 \in \{1, ..., d\}$ are indices of the token embedding.

The position embeddings of all tokens are then element-wise added to the token embeddings, obtaining a position-enriched representation $X^\sharp$ for all tokens.

$$X^\sharp = X + P \quad (10)$$

Afterwards, the position-enriched representation is fed into a multi-head self-attention layer to obtain the contextual representation O.

$$O = \mathrm{MultiHead}(X^\sharp, X^\sharp, X^\sharp) \quad (11)$$

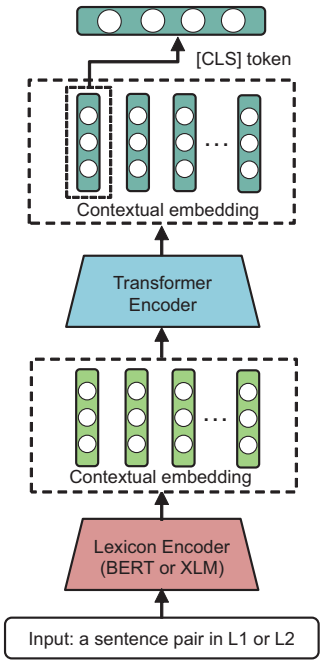


Figure 2: The framework for enhancing pre-trained language models.

where the multi-head self-attention layer is defined by

$$\mathrm{Att}(Q,K,V) = \mathrm{softmax}(\frac{QK^T}{\sqrt{d}})V \quad (12)$$

$$M_i = \mathrm{Att}(QW_i^Q, KW_i^K, VW_i^V) \quad (13)$$

$$\mathrm{MultiHead}(Q,K,V) = [M_1;...;M_h]W^O \quad (14)$$

where h is the number of heads, $W_i^Q \in \mathbb{R}^{d\times\frac{d}{h}}$, $W_i^K \in \mathbb{R}^{d\times\frac{d}{h}}$, $W_i^V \in \mathbb{R}^{d\times\frac{d}{h}}$ and $W^O \in \mathbb{R}^{d\times d}$ are trainable parameters, and [;] denotes the concatenation operator.

The output of the multi-head self-attention layer is then fed into a position-wise feed-forward network to generate the final representation U.

$$U_i = \mathrm{ReLU}(O_iW_1 + b_1)W_2 + b_2 \quad (15)$$

where i denotes the i^{th} token, and $W_1 \in \mathbb{R}^{d\times k}$, $W_2 \in \mathbb{R}^{k\times d}$, $b_1 \in \mathbb{R}^k$, and $b_2 \in \mathbb{R}^d$ are trainable parameters. Following (Vaswani et al. 2017) we set $k = 4d$ in our experiments.

All parameters of the transformer encoder are initialized randomly. Following (Devlin et al. 2019; Liu et al. 2019) we use the contextual embedding of the [CLS] token namely U_1 as the matching representation between P and H.

Enhancing Classical NLI Models

We take the well-known matching-based NLI model ESIM (Chen et al. 2017) for example to demonstrate how to enhance classical NLI models by the TMAN framework. The Enhanced Sequential Inference Model ESIM (Chen et al. 2017) exploits Bi-LSTM (Hochreiter and Schmidhuber 1997) networks to encode sentences and utilizes a co-attention mechanism to capture the interaction between two sentences.

To adapt ESIM to XNLI, we need to align the word embeddings for the target language into the vector space for the source language. To this end, we use the MUSE (Lample et al. 2018) algorithm to align the FastText (Bojanowski et al. 2017) word embeddings for the target language into the vector space for the source language. The embedding of any out-of-vocabulary (OOV) word is initialized randomly. All word embeddings are updatable during training.

Experiments

We conducted experiments on the XNLI (Conneau et al. 2018) benchmark[1], which extends the well-known MultiNLI (Williams, Nangia, and Bowman 2018) benchmark to 15 languages with human-annotated development set and test set. For each language, the training set comprises 393K annotated sentence pairs, whereas both the development set and the test set comprise 7500 annotated sentence pairs each. We applied TMAN to enhance two pre-trained models XLM and multilingual BERT as well as one classical NLI model ESIM. We call the enhanced models TMAN-XLM, TMAN-BERT and TMAN-ESIM, respectively.[2]

For TMAN-XLM, the lexical encoder was initialized by the pre-trained XLM model with 12 transformer layers, which outputs 1024D token embeddings. The transformer encoder was built with 8 heads. We applied dropout (Srivastava et al. 2014) to each layer by setting the dropout rate as 0.1. TMAN-XLM was trained by Adam (Kingma and Ba 2015) with initial learning rate 5e-6, mini-batch size 16, and two training epochs.

For TMAN-BERT, the lexical encoder was initialized by the pre-trained multilingual BERT model with 12 transformer layers, which outputs 768D token embeddings. The transformer encoder was built with 12 heads. TMAN-BERT was trained by Adam with the warmup mechanism (Devlin et al. 2019) and two training epochs, where the initial learning rate was set as 5e-5, the warmup proportion as 10%, the mini-batch size as 32, and the dropout rate as 0.1.

For TMAN-ESIM, the input word vectors were initialized by 300D FastText (Bojanowski et al. 2017) word embeddings that were aligned by the MUSE (Lample et al. 2018) algorithm. The bi-directional output dimension of all Bi-LSTM (Hochreiter and Schmidhuber 1997) networks was set as 200. The dimension of inference representations was also set as 200. We applied dropout to each layer by setting the dropout rate as 0.3. TMAN-ESIM was trained by AdaDelta (Zeiler 2012) with initial learning rate 3.0, mini-batch size 512, maximally 30 training epochs and the early stopping strategy applied according to the performance on the development set.

For above models, we set the hyper-parameter λ as 0.1 according to the performance on the development set. Every sentence is truncated to 128 tokens. To make the results reproducible, we used the translated data in XNLI instead of constructing translated data by applying translators.

[1] http://www.nyu.edu/projects/bowman/xnli/

[2] The code of our implementations is available at https://github.com/qikunxun/TMAN/.

Target language	en	fr	es	de	el	bg	ru	tr	ar	vi	th	zh	hi	sw	ur	△	σ
Machine translation baselines (TRANSLATE-TEST)																	
(Conneau et al. 2018)	73.7	70.4	70.7	68.7	69.1	70.4	67.8	66.3	66.8	66.5	64.4	68.3	64.2	61.8	59.3	67.2	3.7
Multilingual BERT	81.4	-	74.9	74.4	-	-	-	-	70.4	-	-	70.1	-	-	62.1	-	-
XLM	85.0	79.0	79.5	78.1	77.8	77.6	75.5	73.7	73.7	70.8	70.4	73.6	69.0	64.7	65.1	74.2	5.6
Machine translation baselines (TRANSLATE-TRAIN)																	
(Conneau et al. 2018)	73.7	68.3	68.8	66.5	66.4	67.4	66.5	64.5	65.8	66.0	62.8	67.0	62.1	58.2	56.6	65.4	4.2
ESIM†	72.0	67.6	68.7	70.4	70.8	70.5	67.8	66.8	66.0	69.2	66.0	68.5	63.8	62.6	58.1	67.3	3.6
ESIM‡	72.5	71.1	71.1	71.3	70.1	70.6	68.6	67.8	67.3	68.2	66.3	69.1	63.8	62.8	58.3	67.9	3.8
Multilingual BERT	81.9	-	77.8	75.9	-	-	-	-	70.7	-	-	76.6	-	-	61.6	-	-
BERT†	81.9	77.0	77.8	75.9	73.0	73.6	72.6	70.7	70.7	73.2	65.8	76.6	65.4	63.8	61.6	72.0	5.7
XLM	85.0	80.2	80.8	**80.3**	78.1	79.3	**78.1**	74.7	76.5	**76.6**	**75.5**	78.6	**72.3**	70.9	63.2	76.7	5.1
XLM‡	84.9	80.0	80.6	79.2	77.2	79.6	77.3	74.4	75.8	74.7	74.2	78.5	68.1	71.0	65.5	76.1	5.0
TMAN-ESIM	73.5	72.3	72.8	72.1	71.3	71.1	68.6	68.4	67.7	70.4	67.5	70.6	64.6	63.5	60.0	69.0	3.8
TMAN-BERT	82.3	78.2	78.5	77.4	75.1	75.8	74.5	72.7	72.0	74.6	67.3	77.5	67.8	67.4	64.3	73.7	5.1
TMAN-XLM (w/o mixing 2 languages in a mini-batch)	85.3	80.4	81.2	80.0	78.3	80.0	78.0	74.4	76.2	75.2	74.1	79.7	71.5	72.2	66.7	76.9	4.7
TMAN-XLM (w/o the transformer encoder)	85.4	80.5	81.0	79.8	78.1	80.1	77.9	**75.0**	76.3	75.2	74.4	79.8	71.5	72.0	66.8	76.9	4.6
TMAN-XLM	**85.6**	**80.8**	**81.5**	80.2	**78.2**	**80.3**	**78.1**	**75.0**	**76.8**	75.6	74.8	**79.9**	71.8	**72.4**	**66.9**	**77.2**	4.6

Table 1: Comparison results under the TRANSLATE-TRAIN/TEST setting. Every value is the test accuracy in percent. ESIM† is our implementation of the ESIM model using monolingual word embedding. ESIM‡ is the ESIM model that has no adversarial network and is trained on the merged data. BERT† shows our evaluation results on multilingual BERT. XLM‡ is the XLM model that has no adversarial network and is trained on the merged data. △ is the average accuracy for 15 languages. σ is the standard deviation of accuracy for 15 languages.

Main Results

TRANSLATE-TRAIN and TRANSLATE-TEST are two popular settings for evaluating XNLI models. Both deal with training data in the source language and test data in the target language. The difference between these two settings is that TRANSLATE-TRAIN translates the training data to the target language before training models on the training data, and that TRANSLATE-TEST translates the test data to the source language before evaluating models on the test data. TMAN follows a variant of the TRANSLATE-TRAIN setting where the training data are augmented with the original data in the source language. Table 1 reports the results for comparing TMAN-enhanced models with other XNLI models on the XNLI benchmark under the above settings, where the source language is fixed as English (en) and the target language is one of the 15 languages in the XNLI benchmark.

TMAN-ESIM achieves 69.0% accuracy on the XNLI test set averaged by 15 target languages. The classical ESIM model (denoted ESIM†) achieves 67.3% accuracy on the XNLI test set on average. It implies that TMAN pushes ESIM by an absolute gain of 1.7% accuracy on average. The difference between TMAN-ESIM and ESIM† in average accuracy is statistically significant with p-value 3.9e-5 by a one-tailed t-test.

TMAN-BERT achieves 73.7% accuracy on the XNLI test set averaged by 15 target languages. Multilingual BERT (Devlin et al. 2019) only reported the results for 5 target languages. In order to make the results complete, we conducted experiments for multilingual BERT in other target languages, shown by BERT†. It can be seen that BERT† achieves 72.0% accuracy on the XNLI test set on average. It implies that TMAN pushes multilingual BERT by an absolute gain of 1.7% accuracy on average. The difference between TMAN-BERT and BERT† in average accuracy is statistically significant with p-value 6.2e-7 by a one-tailed t-test.

TMAN-XLM achieves 77.2% accuracy on the XNLI test set averaged by 15 target languages. As reported in (Lample and Conneau 2019), XLM achieves 76.7% accuracy on the XNLI test on average. It implies that TMAN pushes XLM by an absolute gain of 0.5% accuracy on average. The difference between TMAN-XLM and XLM in average accuracy is statistically significant with p-value 4.8e-2 by a one-tailed t-test. For four languages de, vi, th and hi, TMAN-XLM achieves slightly lower accuracies than XLM. This may be caused by the OOV problem, since XLM only provides models using the vocabulary of the XNLI test set. The OOV rates in percent for all languages in the XNLI training set are respectively en(9.77), fr(12.08), es(15.25), de(18.93), el(23.29), bg(1.55), ru(3.48), tr(20.17), ar(1.33), vi(35.88), th(32.95), zh(5.07), hi(30.46), sw(5.65) and ur(4.36). The Pearson correlation coefficient between OOV rates and accuracy differences between TMAN-XLM and XLM is -0.68. This confirms our assumption that high OOV rates lead to lower performance of TMAN-XLM for the four languages.

To verify the effectiveness of adversarial training, we also evaluated the performance of ESIM and XLM when they are trained on the merged data in both the source language and target language, shown by ESIM‡ and XLM‡, respectively. The difference between TMAN-ESIM and ESIM‡ in average accuracy is statistically significant with p-value 3.7e-6 by a one-tailed t-test. The difference between TMAN-XLM and XLM‡ in average accuracy is statistically significant with p-value 2.7e-5 by a one-tailed t-test. Although training ESIM and XLM on the merged data may improve the performance for some languages such as French (fr) for ESIM and Urdu (ur) for XLM, the performance achieved by this way

Target language	en	fr	es	de	el	bg	ru	tr	ar	vi	th	zh	hi	sw	ur	$\triangle_{\text{en}\rightarrow x}$	$\triangle_{\text{en}}$
Evaluation of cross-lingual sentence encoders																	
XLM (en)	85.0	78.7	78.9	77.8	76.6	77.4	75.3	72.5	73.1	76.1	73.2	76.5	69.6	68.4	67.3	-	-
TMAN-XLM (en→fr)	84.7	**80.8**	80.4	79.0	77.2	78.5	77.1	73.3	74.6	76.5	74.1	76.4	70.9	70.5	67.0	75.0*	74.1
TMAN-XLM (en→es)	**85.6**	80.6	**81.5**	78.0	**78.6**	79.3	77.3	74.0	75.3	**77.3**	74.0	77.0	70.6	70.2	68.0	75.4*	74.0
TMAN-XLM (en→de)	84.7	79.4	79.4	**80.2**	77.4	78.0	75.9	72.3	73.4	75.1	73.1	75.7	69.8	70.0	66.8	74.3	74.1
TMAN-XLM (en→el)	84.4	78.9	79.6	77.9	78.2	78.3	76.1	71.7	73.9	75.1	72.4	76.1	70.5	69.0	66.9	74.3	74.2
TMAN-XLM (en→bg)	84.7	78.9	79.5	78.7	77.8	**80.3**	77.0	73.2	74.3	76.3	73.6	76.6	70.4	69.8	67.0	74.9*	74.2
TMAN-XLM (en→ru)	84.3	79.2	79.7	77.8	78.0	79.6	**78.1**	73.5	75.2	76.6	73.7	77.4	71.0	70.9	67.8	75.4*	74.3
TMAN-XLM (en→tr)	84.6	78.9	79.4	77.9	76.7	77.4	75.6	**75.0**	73.2	75.3	73.8	76.6	71.0	69.5	67.2	74.8*	74.5
TMAN-XLM (en→ar)	84.6	79.1	79.9	77.3	77.3	78.3	75.5	72.4	**76.8**	74.9	73.5	75.9	69.4	69.8	65.5	74.5	74.5
TMAN-XLM (en→vi)	84.1	78.2	78.5	76.9	76.2	76.2	74.7	71.5	72.1	75.6	70.4	74.3	68.9	68.5	65.9	73.3	74.3
TMAN-XLM (en→th)	83.7	78.3	78.8	76.4	76.9	77.6	75.3	71.7	73.4	75.9	**74.8**	76.2	69.2	69.9	66.2	74.3	74.5
TMAN-XLM (en→zh)	85.2	79.6	80.4	78.7	78.0	78.6	77.4	74.3	74.7	76.6	74.3	**79.9**	**72.1**	71.4	**69.2**	**75.8***	74.2
TMAN-XLM (en→hi)	84.4	77.5	78.7	77.2	76.9	77.4	75.9	71.2	73.9	75.9	73.6	76.3	71.8	69.8	66.7	74.7	74.8
TMAN-XLM (en→sw)	83.9	78.3	79.1	77.6	76.8	77.5	75.8	72.8	73.7	74.7	73.2	75.9	70.0	**72.4**	67.1	74.8	74.8
TMAN-XLM (en→ur)	83.0	77.6	78.5	76.4	76.4	77.3	75.3	72.0	72.4	74.3	72.2	76.5	70.4	69.2	66.9	74.5	74.9

Table 2: Comparison results under the cross-lingual setting. Every value is the test accuracy in percent. $\triangle_{\text{en}\rightarrow x}$ and $\triangle_{\text{en}}$ are respectively the average accuracy of TMAN-XLM (en$\rightarrow x$) and the average accuracy of XLM (en) for 13 languages including all but English and the target language x. Every average accuracy marked with * is significantly higher than the average accuracy of XLM (en) for the same 13 languages with p-value < 0.05 by a one-tailed t-test.

is still significantly lower than the performance achieved by further adding an adversarial network.

We also verified the effectiveness of the special treatments in our adversarial training framework that are different from ordinary ways. The results obtained by restricting a mini-batch to consist of data in a single language are shown by TMAN-XLM (w/o mixing 2 languages in a mini-batch). It can be seen that mixing different languages in a mini-batch improves the performance for all 15 languages and achieves an absolute gain of 0.3% accuracy on average. This improvement is statistically significant with p-value 2.0e-5 by a one-tailed t-test. The results obtained by removing the transformer encoder are shown by TMAN-XLM (w/o the transformer encoder). It can be seen that adding the transformer encoder also improves the performance for all 15 languages and achieves an absolute gain of 0.3% accuracy on average. This improvement is also statistically significant with p-value 5.0e-6 by a one-tailed t-test.

Language-invariant Evaluation

To evaluate whether an XNLI model has the ability for capturing language-invariant features, a cross-lingual setting is often used, which directly uses training data in the source language and test data in the target language without translating either the training data or the test data. As reported by (Lample and Conneau 2019), XLM achieves the best performance in this setting prior to this work. To evaluate whether TMAN advances this ability, a variant setting can be used for TMAN-enhanced models, which augments the training data in an auxiliary language by translating training data to the auxiliary language.

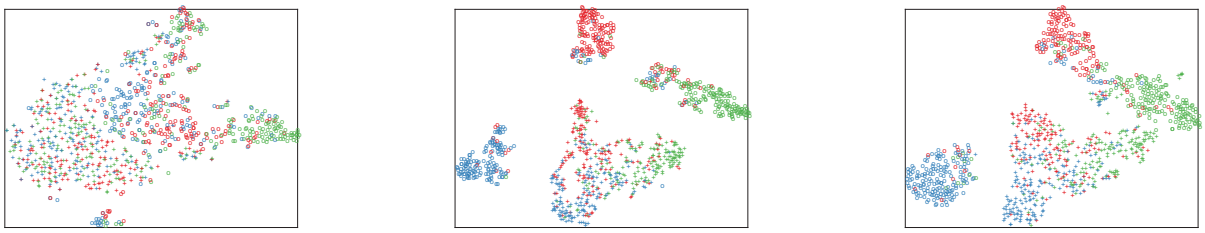
Table 2 shows the comparison results between different TMAN-XLM models that use different auxiliary languages with the best XLM model reported by (Lample and Conneau 2019), which is denoted by XLM (en), under the cross-lingual settings, where the source language is fixed as English (en) and the target language is one of the 15 languages in the XNLI benchmark. It can be seen that 10 out of 14 TMAN-XLM models using different auxiliary languages perform at least as well as XLM (en), among which 6 models significantly outperform XLM (en) with p-values < 0.05 by one-tailed t-tests. In particular, when using Chinese (zh) as the auxiliary language, TMAN-XLM achieves 75.8% average accuracy for 13 languages including all but English and Chinese, pushing XLM (en) by an absolute gain of 1.6% average accuracy for the same 13 languages. These results show that, thanks to the introduction of augmented training data and the introduction of the language discriminator to compete against the NLI label discriminator, TMAN better captures language-invariant features, giving more accurate predictions on NLI labels under the cross-lingual setting.

Variant Model	Avg acc.	p-value
Original TMAN-ESIM	**69.0**	-
(1) W/o the TMAN framework (aka ESIM†)	67.3	3.9e-5
(2) W/o the domain adversarial network (aka ESIM‡)	67.9	3.7e-6
(3) Using fixed cross-lingual word embeddings	68.7	4.7e-5
(4a) Using updatable monolingual word embeddings	68.0	2.0e-9
(4b) Using fixed monolingual word embeddings	67.7	2.4e-8
(5) Using random word embeddings	62.0	9.5e-12

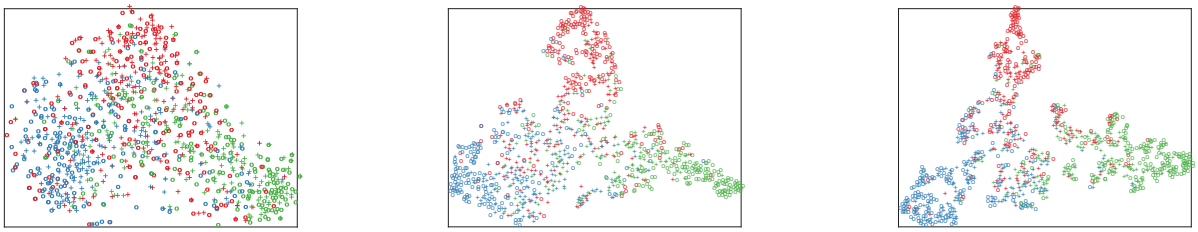
Table 3: Ablation study results on TMAN-ESIM.

Ablation Study on TMAN-ESIM

Table 3 shows the complete ablation study results on TMAN-ESIM. Besides two variants of TMAN-ESIM namely (1) ESIM† and (2) ESIM‡ whose results have been



(a) Visualization of ESIM (b) Visualization of multilingual BERT (c) Visualization of XLM



(d) Visualization of TMAN-ESIM (e) Visualization of TMAN-BERT (f) Visualization of TMAN-XLM

Figure 3: Visualization of the matching representations.

reported in Table 1, we also evaluated other variants of TMAN-ESIM that alter the usage of word embeddings. In (3) we fixed the cross-lingual word embeddings in the course of training. Results show that TMAN-ESIM achieves better performance when the word embeddings are tuned during training. In (4a) and (4b) we used monolingual word embeddings rather than cross-lingual word embeddings, where in (4a) we tuned the monolingual word embeddings during training and in (4b) we fixed the monolingual word embeddings. Results show that the usage of cross-lingual word embeddings achieves better performance than using monolingual word embeddings no matter monolingual word embeddings are updatable or not during training. In (5) we used random word embeddings that can be updated in the course of training. Results show that the use of random word embeddings leads to the worst performance among all variants of TMAN-ESIM.

Visualization Analysis

To clarify why our proposed adversarial training framework improves the accuracy in predicting NLI labels, we visualize the matching representations generated by an original model and that by the corresponding enhanced model in Figure 3, where each point stands for the matching representation of a premise-hypothesis pair, points marked "o" correspond to pairs in English, points marked "+" correspond to pairs in Urdu, blue points correspond to pairs with label "entailment", red points correspond to pairs with label "neutral", and green points correspond to pairs with label "contradiction". The figures were obtained by randomly selecting 500 premise-hypothesis pairs in English and 500 pairs in Urdu from the development set and by using t-SNE (Rauber, Falcão, and Telea 2016) to reduce the dimension of the data for visualization. Compared with the original model, the model enhanced by TMAN yields clearer distinction between different labels and more confusion between different languages. This implies that our proposed adversarial training framework encourages to align matching representations in different languages into the same space, resulting in more accurate predictions on NLI labels.

Conclusions and Future Work

In this paper we have proposed an adversarial training framework namely TMAN for cross-lingual natural language inference. TMAN enhances existing neural models for NLI mainly by training on merged data from both the source language and the target language and by introducing a language discriminator to compete against the NLI label discriminator. Experimental results show that TMAN pushes three existing NLI models namely ESIM, BERT and XLM by a significant absolute gain in classification accuracy on the XNLI benchmark. Our ablation study further confirms that all enhancements introduced by TMAN contribute to the performance improvement on the XNLI benchmark.

Our future work is two-fold. On one hand, we plan to extend TMAN to train on data from three or more different languages and to evaluate the effects. On the other hand, we plan to enhance TMAN with external multilingual resources such as BabelNet (Navigli and Ponzetto 2012) and ConceptNet (Speer, Chin, and Havasi 2017).

Acknowledgements

This work was partly supported by National Natural Science Foundation of China (61375056 and 61876204) as well as Science and Technology Program of Guangzhou (201804010496).

References

Artetxe, M., and Schwenk, H. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *TACL* 7:597–610.

Bojanowski, P.; Grave, E.; Joulin, A.; and Mikolov, T. 2017. Enriching word vectors with subword information. *TACL* 5:135–146.

Chen, Q.; Zhu, X.; Ling, Z.; Wei, S.; Jiang, H.; and Inkpen, D. 2017. Enhanced LSTM for natural language inference. In *ACL*, 1657–1668.

Chen, X.; Sun, Y.; Athiwaratkun, B.; Cardie, C.; and Weinberger, K. Q. 2018. Adversarial deep averaging networks for cross-lingual sentiment classification. *TACL* 6:557–570.

Clinchant, S.; Goutte, C.; and Gaussier, É. 2006. Lexical entailment for information retrieval. In *ERIR*, 217–228.

Conneau, A.; Rinott, R.; Lample, G.; Williams, A.; Bowman, S. R.; Schwenk, H.; and Stoyanov, V. 2018. XNLI: Evaluating cross-lingual sentence representations. In *EMNLP*, 2475–2485.

Devlin, J.; Chang, M.; Lee, K.; and Toutanova, K. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 4171–4186.

España-Bonet, C.; Varga, Á. C.; Barrón-Cedeño, A.; and van Genabith, J. 2017. An empirical analysis of nmt-derived interlingual embeddings and their use in parallel sentence identification. *J. Sel. Topics Signal Processing* 11(8):1340–1350.

Ganin, Y.; Ustinova, E.; Ajakan, H.; Germain, P.; Larochelle, H.; Laviolette, F.; Marchand, M.; and Lempitsky, V. S. 2016. Domain-adversarial training of neural networks. *Journal of Machine Learning Research* 17:59:1–59:35.

Goodfellow, I. J.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A. C.; and Bengio, Y. 2014. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, 2672–2680.

Harabagiu, S. M., and Hickl, A. 2006. Methods for using textual entailment in open-domain question answering. In *ACL*, 905–912.

Hermann, K. M., and Blunsom, P. 2014. Multilingual models for compositional distributed semantics. In *ACL*, 58–68.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural Computation* 9(8):1735–1780.

Johnson, M.; Schuster, M.; Le, Q. V.; Krikun, M.; Wu, Y.; Chen, Z.; Thorat, N.; Viégas, F. B.; Wattenberg, M.; Corrado, G.; Hughes, M.; and Dean, J. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *TACL* 5:339–351.

Kang, D.; Khot, T.; Sabharwal, A.; and Hovy, E. H. 2018. AdvEntuRe: Adversarial training for textual entailment with knowledge-guided examples. In *ACL*, 2418–2428.

Kingma, D. P., and Ba, J. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Lample, G., and Conneau, A. 2019. Cross-lingual language model pretraining. *CoRR* abs/1901.07291.

Lample, G.; Conneau, A.; Ranzato, M.; Denoyer, L.; and Jégou, H. 2018. Word translation without parallel data. In *ICLR*.

Liu, X.; He, P.; Chen, W.; and Gao, J. 2019. Multi-task deep neural networks for natural language understanding. In *ACL*, 4487–4496.

Navigli, R., and Ponzetto, S. P. 2012. Babelnet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artif. Intell.* 193:217–250.

Poliak, A.; Belinkov, Y.; Glass, J. R.; and Durme, B. V. 2018. On the evaluation of semantic phenomena in neural machine translation using natural language inference. In *NAACL-HLT*, 513–523.

Radford, A.; Narasimhan, K.; Salimans, T.; and Sutskever, I. 2018. Improving language understanding with unsupervised learning. Technical report, OpenAI.

Rauber, P. E.; Falcão, A. X.; and Telea, A. C. 2016. Visualizing time-dependent data using dynamic t-sne. In *EuroVis*, 73–77.

Schwenk, H., and Douze, M. 2017. Learning joint multilingual sentence representations with neural machine translation. In *Workshop on Representation Learning for NLP, Rep4NLP@ACL*, 157–167.

Speer, R.; Chin, J.; and Havasi, C. 2017. ConceptNet 5.5: An open multilingual graph of general knowledge. In *AAAI*, 4444–4451.

Srivastava, N.; Hinton, G. E.; Krizhevsky, A.; Sutskever, I.; and Salakhutdinov, R. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research* 15(1):1929–1958.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, 6000–6010.

Wang, W., and Pan, S. J. 2018. Transition-based adversarial network for cross-lingual aspect extraction. In *IJCAI*, 4475–4481.

Williams, A.; Nangia, N.; and Bowman, S. R. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *NAACL-HLT*, 1112–1122.

Zeiler, M. D. 2012. ADADELTA: an adaptive learning rate method. *CoRR* abs/1212.5701.

Zou, B.; Xu, Z.; Hong, Y.; and Zhou, G. 2018. Adversarial feature adaptation for cross-lingual relation classification. In *COLING*, 437–448.