FILTER: An Enhanced Fusion Method for Cross-lingual Language Understanding

Yuwei Fang*, Shuohang Wang*, Zhe Gan, Siqi Sun, Jingjing Liu
Microsoft Dynamics 365 AI Research
{yuwen,shuohang.wang,zhe.gan,siqi.sun,jingj}@microsoft.com

Abstract
Large-scale cross-lingual language models (LM), such as mBERT, Unicoder and XLM, have achieved great success in cross-lingual representation learning. However, when applied to zero-shot cross-lingual transfer tasks, most existing methods use only single-language input for LM finetuning, without leveraging the intrinsic cross-lingual alignment between different languages that proves essential for multilingual tasks. In this paper, we propose FILTER, an enhanced fusion method that takes cross-lingual data as input for XLM finetuning. Specifically, FILTER first encodes text input in the source language and its translation in the target language independently in the shallow layers, then performs cross-language fusion to extract multilingual knowledge in the intermediate layers, and finally performs further language-specific encoding. During inference, the model makes predictions based on the text input in the target language and its translation in the source language. For simple tasks such as classification, translated text in the target language shares the same label as the source language. However, this shared label becomes less accurate or even unavailable for more complex tasks such as question answering, NER and POS tagging. To tackle this issue, we further propose an additional pseudo-labels for translated text in the target language. Extensive experiments demonstrate that FILTER achieves new state of the art on XTREME and XGLUE.

Introduction
Cross-lingual low-resource adaptation has been a critical and exigent problem in the NLP field, despite recent success in large-scale language models (mostly trained on English with abundant training corpora). How to adapt models trained in high-resource languages (e.g., English) to low-resource ones (most of the 6,900 languages in the world) still remains challenging. To address the proverbial domain gap between languages, three schools of approach have been widely studied. (i) Unsupervised pre-training: to learn a universal encoder (cross-lingual language model) for different languages. For example, mBERT (Devlin et al. 2019), Unicoder (Huang et al. 2019) and XLM (Lample and Conneau 2019) have achieved strong performance on many cross-lingual tasks by successfully transferring knowledge from source language to a target one. (ii) Supervised training: to enforce models insensitive to labeled data across different languages, through teacher forcing (Wu et al. 2020) or adversarial learning (Cao, Liu, and Wan 2020). (iii) Translation: to translate either source language to the target one, or vice versa (Cui et al. 2019; Hu et al. 2020; Liang et al. 2020), so that training and inference can be performed in the same language.

The translation approach has proven highly effective on recent multilingual benchmarks. For example, the translate-train method has achieved state of the art on XTREME (Hu et al. 2020) and XGLUE (Liang et al. 2020). However, translate-train is simple data augmentation, which doubles training data by translating source text into target languages. Thus, only single-language input is considered for finetuning with augmented data, leaving out cross-lingual alignment between languages unexplored. Dual BERT (Cui et al. 2019) is recently proposed to make use of the representations learned from source language to help target language understanding. However, it only injects information from the source language into the decoder of target language, without scoping into the intrinsic relations between languages.

Motivated by this, we propose FILTER, a generic and flexible framework that leverages translated data to enforce fusion between languages for better cross-lingual language understanding. As illustrated in Figure 2(c), FILTER first (i) encodes a translated language pair separately in shallow layers; then (ii) performs cross-lingual fusion between languages in the intermediate layers; and finally (iii) encodes language-specific representations in deeper layers. Compared to the translate-train baseline (Figure 2(a)), FILTER learns additional cross-lingual alignment that is instrumental to cross-lingual representations. Furthermore, compared to simply concatenating the language pair as the input of XLM (Figure 2(b)), FILTER strikes a well-measured balance between cross-lingual fusion and individual language representation learning.

For classification tasks such as natural language inference, translated text in the target language shares the same label as the source language. However, for question answer-
Cross-lingual datasets

Cross-lingual language understanding has been investigated for many NLP tasks, where the knowledge learned from a pivot language (e.g., English) is transferred to other languages indirectly, as labeled data in low-resource languages are often scarce. There exist many multilingual corpora for diverse NLP tasks. Nivre et al. (2016) released a collection of multilingual treebanks on universal dependencies for 33 languages. Pan et al. (2017) introduced cross-lingual name tagging and linking for 282 languages. Other multilingual datasets range over tasks such as document classification, natural language inference, information retrieval, paraphrase identification, and summarization (Klementiev, Titov, and Bhattarai 2012; Cer et al. 2017; Conneau et al. 2018; Sasaki et al. 2018; Yang et al. 2019; Zhu et al. 2019).

More recent studies on open-domain question answering and machine reading comprehension also introduced cross-lingual datasets, such as MLQA (Lewis et al. 2020), XQuAD (Artetxe, Ruder, and Yogatama 2020), and TyDiQA (Clark et al. 2020). Most recently, XTREME (Hu et al. 2020) and XGLUE (Liang et al. 2020) released several datasets across multiple tasks, and set up public leaderboards for evaluating cross-lingual models. In this paper, we work on both XTREME (see Figure 1 for examples) and XGLUE to demonstrate the effectiveness of our proposed method.

Cross-lingual Models

Most previous work tackles cross-lingual problems in two fashions: (i) cross-lingual zero-shot transfer; and (ii) translate-train/test. For cross-lingual zero-shot transfer, models are trained on labeled data in the source language only, and directly evaluated on target languages. Early work focused on training multilingual word embeddings (Mikolov, Le, and Sutskever 2013; Faruqui and Dyer 2014; Xu et al. 2018), while more recent work proposed to pre-train cross-lingual language models, such as mBERT (Devlin et al. 2019), XLM (Lample and Conneau 2019), and XLM-Roberta (Conneau et al. 2020), to learn contextualized representations.

For translate-train/test, external machine translation tools are leveraged. A common approach is to augment training data by first translating all data in the source language to target languages, then train the model on translated data (Hu et al. 2020; Liang et al. 2020). Another approach is translate-test (Hu et al. 2020) or round-trip translation (Zhu et al. 2019), which translates the text in the test set of target languages into source language, so that all the models trained in the source language can be directly applied for inference, and the prediction can be translated back to the target language if needed. To enhance these translation-based pipelines, Cui et al. (2019) proposed to simultaneously model text in both languages to enrich the learned language representations. Huang, Ji, and May (2019) proposed to use adversarial transfer to enhance low-resource name tagging. And Cao, Liu, and Wan (2020) proposed to jointly learn the alignment and perform summarization across languages. FILTER follows the translate-train line of thought, but provides a better way to encode text in both source and target languages simultaneously.

Related Work

Cross-lingual Datasets

Cross-lingual language understanding has been investigated for many NLP tasks, where the knowledge learned from a pivot language (e.g., English) is transferred to other languages indirectly, as labeled data in low-resource languages are often scarce. There exist many multilingual corpora for diverse NLP tasks. Nivre et al. (2016) released a collection of multilingual treebanks on universal dependencies for 33 languages. Pan et al. (2017) introduced cross-lingual name tagging and linking for 282 languages. Other multilingual datasets range over tasks such as document classification, natural language inference, information retrieval, paraphrase identification, and summarization (Klementiev, Titov, and Bhattarai 2012; Cer et al.

Our code is released at https://github.com/yuwfan/FILTER

Figure 1: Examples from XTREME for cross-lingual natural language inference, part-of-speech tagging, and question answering tasks. The source language is English; the target language can be any other languages.
**Proposed Approach**

In this section, we first introduce the proposed FILTER model architecture, then describe the self-teaching loss for model enhancement. An overview of the framework is illustrated in Figure 2.

**FILTER Architecture**

Although the domain gap between languages has been largely reduced by translate-train method, translated text may not succeed in keeping the semantic meaning and label of the original text unchanged, due to quality constraint of translation tools. Furthermore, the source language and translated target language are usually encoded separately, without tapping into the cross-lingual relations among different languages. Therefore, we propose to use language pairs as input, and fuse the learned representations between languages through intermediate network layers, so that the model can learn cross-lingual information that is instrumental to inference in different languages.

The proposed FILTER model consists of three components: (i) “local” Transformer layers for encoding the input language pair independently; (ii) cross-lingual fusion layers for leveraging the context in different languages; and (iii) deeper domain-specific Transformer layers to shift the focus back on individual languages, after injecting information from the other language. For notation, \( \mathbf{S} \in \mathbb{R}^{d \times l_s} \) and \( \mathbf{T} \in \mathbb{R}^{d \times l_t} \) are denoted as the word embedding matrix for text input \( S \) and \( T \), respectively. Formally,

\[
\mathbf{H}^s_t = \text{Transformer-XLM}_{\text{local}}(\mathbf{S}), \\
\mathbf{H}^t_t = \text{Transformer-XLM}_{\text{local}}(\mathbf{T}),
\]

where the position embeddings are counted from 0 for both sequences, \( \mathbf{H}^s_t \in \mathbb{R}^{d \times l_s} \) and \( \mathbf{H}^t_t \in \mathbb{R}^{d \times l_t} \) are “local” representations of the sequence pair. We set the number of layers in \( \text{Transformer-XLM}_{\text{local}} \) as \( m \), which can be tuned for solving different cross-lingual tasks. The concatenation of the local representations from both languages, \( [\mathbf{H}^s_t; \mathbf{H}^t_t] \in \mathbb{R}^{d \times (l_s + l_t)} \), is the input for the next layer to learn the fusion between different languages, as follows:

\[
[\mathbf{H}^s_t; \mathbf{H}^t_t] = \text{Transformer-XLM}_{\text{fuse}}([\mathbf{H}^s_t; \mathbf{H}^t_t]),
\]

where \([::] \) denotes the concatenation of two matrices, \( \mathbf{H}^s_t \in \mathbb{R}^{d \times l_s} \) and \( \mathbf{H}^t_t \in \mathbb{R}^{d \times l_t} \) are the representations in corresponding languages. We set the number of layers in \( \text{Transformer-XLM}_{\text{fuse}} \) as \( k \), which is another hyperparameter to control the cross-lingual fusion degree. As the final goal is to predict the label in one language, we limit the top layers specifically designed to encode the text in one language, so that not too much noise is introduced from translated text in other languages. Specifically,

\[
\mathbf{H}^s_d = \text{Transformer-XLM}_{\text{domain}}(\mathbf{H}^s_f), \\
\mathbf{H}^t_d = \text{Transformer-XLM}_{\text{domain}}(\mathbf{H}^t_f),
\]

where \( \mathbf{H}^s_d \in \mathbb{R}^{d \times l_s} \) and \( \mathbf{H}^t_d \in \mathbb{R}^{d \times l_t} \) are the final representations for prediction.

As demonstrated in Figure 2, FILTER is realized by stacking the three types of transformer layer on top of each other. FILTER is a generic framework for solving multilingual tasks, where \( k \) and \( m \) can be flexibly set to different values depending on the task. For example, for classification tasks, a smaller \( k \) is desired; while for question answering,
a larger $k$ is needed for absorbing richer cross-lingual information (see Experiments for empirical evidence). Since we use XLM-R as the backbone in our framework, the number of layers in Transformer-XLM is 24 $-$ $k$ $-$ $m$. When $m = 24, k = 0$, FILTER degenerates to the translate-train baseline (Figure 2(a)). When $m = 0, k = 24$, FILTER reduces to another baseline that simply concatenates the text in different languages for XLM finetuning (Figure 2(b)).

FILTER also stacks a task-specific linear layer on top of $H^s_2$ and $H^t_2$ to compute the candidate probabilities and we simplify the whole framework as follows:

$$\mathbf{p}^s, \mathbf{p}^t = \text{FILTER} (S, T),$$

$$L^s = \text{Loss}_{\text{task}}(\mathbf{p}^s, t^s),$$

$$L^t = \text{Loss}_{\text{task}}(\mathbf{p}^t, t^t),$$

where $\mathbf{p}^s$ and $\mathbf{p}^t$ are task-specific probability vectors over candidates, used to compute the final loss based on the labels $t^s$ and $t^t$ from source and target languages, respectively. As shown in Figure 1, for natural language inference, the label can be entailment/contradiction/neutral; for question answering, the label is an answer span positions; for NER and POS tagging, the supervision becomes a sequence of labels.

Self-Teaching Loss

The teacher-student framework, or distillation loss (Hinton, Vinyals, and Dean 2015), has been widely adopted in many areas. In this paper, we propose to add self-teaching loss for training FILTER, and it can be readily adapted to all the cross-lingual tasks. As transferring the labels in source language to the corresponding translated text may introduce noise due to the word order or even semantic meaning changes after translation, the additional self-teaching loss is to bridge this gap.

The proposed training procedure is summarized in Algorithm 1. We first train a “teacher” FILTER based on clean labels in the source language and the transferred “noisy” labels in the target language (if available) with loss from Eqn. (4). This FILTER will then be used as a teacher to generate pseudo soft-labels to regularize a second FILTER (student) trained from scratch. As the noise mainly comes from translated text, we only add soft labels in the target language during the training of the second FILTER. Specifically,

$$\mathbf{p}_{tea}^s, \mathbf{p}_{tea}^t = \text{FILTER}_{tea} (S, T),$$

$$\mathbf{p}_{stu}^s, \mathbf{p}_{stu}^t = \text{FILTER}_{stu} (S, T),$$

$$L^{kl} = \text{Loss}_{KL} (\mathbf{p}_{tea}^t, \mathbf{p}_{stu}^t),$$

where $\text{Loss}_{KL}$ denotes KL divergence. The soft label $\mathbf{p}_{tea}^t$ is fixed when training the student FILTER, which is used for final prediction. When no labels can be transferred to the target language, this method helps the model receive more gradients on the target language, instead of purely on the source side, thus reducing the domain gap between languages. When labels can be transferred, it serves as a smoothing or regularization term appended to the supervised losses. By merging the self-teaching loss, our final training objective for the student FILTER is summarized as:

$$L^{final} = L^s + \lambda L^t + (1 - \lambda) L^{kl},$$

where $\lambda$ is a hyper-parameter to tune, and $\lambda$ is set to zero when no labels in the target languages can be transferred from the source language (e.g., for NER and POS tagging).

Inference

During inference, we pair the text input in the target language with the translated text in the source language, so that FILTER can fuse the information from both languages. For classification tasks, we use the probabilities from either source or target language for prediction. However, for structured prediction and question answering tasks, only the probabilities from the target language can be used for prediction, as the tagging order is different between languages, and the answers are also difficult to evaluate if in different languages. Therefore, for simplicity, we consistently use the probabilities $\mathbf{p}_{stu}^t$ from the target language for final prediction.

Experiments

In this section, we present experimental results on the XTREME and XGLUE benchmarks and provide detailed analysis on the effectiveness of FILTER.

Datasets

There are nine datasets in both XTREME (Hu et al. 2020) and XGLUE (Liang et al. 2020) benchmarks for cross-lingual language understanding, which can be grouped into four categories (Classification, Structured Prediction, QA, and Retrieval). The statistics of each dataset is summarized in Table 1. Note that cross-lingual language generation tasks in XGLUE are not included.

Cross-lingual Sentence Classification includes two common tasks: (i) Cross-lingual Natural Language Inference (XNLI) (Conneau et al. 2018), and (ii) Cross-lingual Paraphrase Adversaries from Word Scrambling (PAWS-X) (Yang et al. 2019). In XGLUE, they further include another four practical tasks selected from Search, Ads and News scenarios: News Classification, Query-Ad Matching, Web Page Ranking and QA Matching.

Cross-lingual Structured Prediction includes two tasks: POS tagging and NER. In XTREME, the Wikiann
Our implementation is based on HuggingFace’s Transformers (Wolf et al. 2019). We leverage the pre-trained XLM-R model (Conneau et al. 2020) to initialize our \textsc{Filter}, which contains 24 layers, each layer with 1,024 hidden states. For fair comparison to XLM-R, each transformer layer in \textsc{Filter} is shared for encoding both source and target languages, so that the total number of parameters are exactly the same as XLM-R.

We conduct experiments on 8 Nvidia V100-32GB GPU cards for model finetuning, and set batch size to 64 for all tasks. For self-teaching loss, we set the weight of the KL loss to 1.0 for structured prediction tasks where no labels are available in the target language. We set the weight of KL loss for classification and QA tasks to 0.5 and 0.1 respectively, by searching over $[0.1, 0.3, 0.5]$. As the official XTREME repo\(^3\) does not provide translated target language data for POS and NER, we use Microsoft Machine Translator\(^4\) for translation. More details on translation data and model hyper-parameters are provided in Appendix.

### Baselines

We compare \textsc{Filter} with previous state-of-the-art multilingual models:

- **Pre-trained models:** \textit{mBERT} (Devlin et al. 2019), \textit{XLM} (Lample and Conneau 2019), \textit{XLM-R} (Conneau et al. 2020), \textit{MMTE} (Siddhant et al. 2020), \textit{InfoXLM} (Chi et al. 2020), \textit{Unicoder} (Huang et al. 2019) pre-train Transformer models on large-scale multi-lingual dataset including machine translation data.

- **Data augmentation:** X-STILTs (Phang et al. 2020) first finetunes XLM-R on an additional intermediate auxiliary task, then further finetunes on the target task.

- **Translate-train** (Hu et al. 2020) finetunes cross-lingual pre-trained language model XLM-R on English training data and all translated data by using Google’s in-house Machine Translation system.

### Experimental Results

Table 2 and 3 summarizes our results on XTREME and XGLUE, outperforming all the leaderboard submissions. On XTREME, compared to the unpublished state-of-the-art VECO approach, \textsc{Filter} outperforms by 2.8/1.5/1.3/4.0 points on the four categories respectively, achieving an average score of 77.0, an absolute 2.2-point improvement. Compared to the XLM-R baseline, we achieve an absolute 8.8-point improvement (77.0 vs. 68.2), which is a significant

---

\(^3\)https://github.com/google-research/xtreme

\(^4\)https://azure.microsoft.com/en-us/services/cognitive-services/translator/
margin. On XGLUE, compared to the Unicoder baseline, FILTER achieves an absolute 4.0-point improvement.

<table>
<thead>
<tr>
<th>Model</th>
<th>Avg</th>
<th>NER</th>
<th>POS</th>
<th>NC</th>
<th>MLQA</th>
<th>XNLI</th>
<th>PAWS-X</th>
<th>QADSM</th>
<th>WPR</th>
<th>QAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unicoder</td>
<td>76.1</td>
<td>79.7</td>
<td>79.6</td>
<td>83.5</td>
<td>66.0</td>
<td>75.3</td>
<td>90.1</td>
<td>68.4</td>
<td>73.9</td>
<td>68.9</td>
</tr>
<tr>
<td>FILTER</td>
<td>80.1</td>
<td>82.6</td>
<td>81.6</td>
<td>83.5</td>
<td>76.2</td>
<td>83.9</td>
<td>93.8</td>
<td>71.4</td>
<td>74.7</td>
<td>73.4</td>
</tr>
</tbody>
</table>

Table 3: Results on the test set of XGLUE. FILTER achieves new state of the art at the time of submission (Sep. 14, 2020). Note that cross-lingual language generation tasks are not included. Leaderboard: https://microsoft.github.io/XGLUE.

<table>
<thead>
<tr>
<th>Model</th>
<th>Pair sentence</th>
<th>Structured prediction</th>
<th>Question answering</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>XNLI</td>
<td>PAWS-X</td>
<td>POS</td>
</tr>
<tr>
<td>mBERT</td>
<td>76.1</td>
<td>79.7</td>
<td>79.6</td>
</tr>
<tr>
<td>FILTER</td>
<td>80.1</td>
<td>82.6</td>
<td>81.6</td>
</tr>
</tbody>
</table>

Table 4: Overall test results on three different categories of cross-lingual language understanding tasks. Results of mBERT (Devlin et al. 2019), XLM (Lample and Conneau 2019) and XLM-R (Conneau et al. 2020) are from XTREME (Hu et al. 2020). InfoXLM (Chi et al. 2020) only provides results on XNLI and MLQA. We also experimented on translate-train with XLM-R as an additional baseline for fair comparison with FILTER.

<table>
<thead>
<tr>
<th>Model</th>
<th>en</th>
<th>ar</th>
<th>bg</th>
<th>de</th>
<th>el</th>
<th>es</th>
<th>fr</th>
<th>hi</th>
<th>ru</th>
<th>sw</th>
<th>th</th>
<th>tr</th>
<th>ur</th>
<th>vi</th>
<th>zh</th>
<th>avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>mBERT</td>
<td>80.8</td>
<td>64.3</td>
<td>68.0</td>
<td>70.0</td>
<td>65.3</td>
<td>73.5</td>
<td>73.4</td>
<td>58.9</td>
<td>67.8</td>
<td>49.7</td>
<td>54.1</td>
<td>60.9</td>
<td>57.2</td>
<td>69.3</td>
<td>67.8</td>
<td>65.4</td>
</tr>
<tr>
<td>MMTE</td>
<td>79.6</td>
<td>64.9</td>
<td>70.4</td>
<td>68.2</td>
<td>67.3</td>
<td>71.6</td>
<td>69.5</td>
<td>63.5</td>
<td>66.2</td>
<td>61.9</td>
<td>66.2</td>
<td>63.6</td>
<td>60.0</td>
<td>69.7</td>
<td>69.2</td>
<td>67.5</td>
</tr>
<tr>
<td>XLM</td>
<td>82.8</td>
<td>66.0</td>
<td>71.9</td>
<td>72.7</td>
<td>70.4</td>
<td>75.5</td>
<td>74.3</td>
<td>62.5</td>
<td>69.9</td>
<td>58.1</td>
<td>65.5</td>
<td>66.4</td>
<td>59.8</td>
<td>70.7</td>
<td>70.2</td>
<td>69.1</td>
</tr>
<tr>
<td>XLM-R</td>
<td>88.7</td>
<td>77.2</td>
<td>83.0</td>
<td>82.5</td>
<td>80.8</td>
<td>83.7</td>
<td>82.2</td>
<td>75.6</td>
<td>79.1</td>
<td>71.2</td>
<td>77.4</td>
<td>78.0</td>
<td>71.7</td>
<td>79.3</td>
<td>78.2</td>
<td>79.2</td>
</tr>
<tr>
<td>XLM-R (TT)</td>
<td>88.6</td>
<td>82.2</td>
<td>85.2</td>
<td>84.5</td>
<td>84.5</td>
<td>85.7</td>
<td>84.2</td>
<td>80.8</td>
<td>81.8</td>
<td>77.0</td>
<td>80.2</td>
<td>82.1</td>
<td>77.7</td>
<td>82.6</td>
<td>82.7</td>
<td>82.6</td>
</tr>
<tr>
<td>FILTER</td>
<td>89.7</td>
<td>83.2</td>
<td>86.2</td>
<td>85.5</td>
<td>85.1</td>
<td>86.6</td>
<td>85.6</td>
<td>80.9</td>
<td>83.4</td>
<td>78.2</td>
<td>82.2</td>
<td>83.1</td>
<td>77.4</td>
<td>83.7</td>
<td>83.7</td>
<td>83.6</td>
</tr>
<tr>
<td>FILTER + ST</td>
<td>89.5</td>
<td>83.6</td>
<td>86.4</td>
<td>85.6</td>
<td>85.4</td>
<td>86.6</td>
<td>85.7</td>
<td>81.1</td>
<td>83.7</td>
<td>78.7</td>
<td>81.7</td>
<td>83.2</td>
<td>79.1</td>
<td>83.9</td>
<td>83.8</td>
<td>83.9</td>
</tr>
</tbody>
</table>

Table 5: XNLI accuracy scores for each language. Results of mBERT, MMTE, XLM and XLM-R are from XTREME (Hu et al. 2020). mtl denotes translate-train in multi-task version. TT: translate train, ST: self-teaching.
Table 6: Analysis on cross-lingual transfer gap of different models on XTREME benchmark (except for retrieval task). A lower gap indicates a better cross-lingual transfer model. The average score (Avg) is calculated on all classification and QA tasks.

<table>
<thead>
<tr>
<th>Model</th>
<th>XNLI</th>
<th>PAWS-X</th>
<th>XQuAD</th>
<th>MLQA</th>
<th>TyDiQA-GoldP</th>
<th>Avg</th>
<th>POS</th>
<th>NER</th>
</tr>
</thead>
<tbody>
<tr>
<td>mBERT(Hu et al. 2020)</td>
<td>16.5</td>
<td>14.1</td>
<td>25.0</td>
<td>27.5</td>
<td>22.2</td>
<td>21.1</td>
<td>25.5</td>
<td>23.6</td>
</tr>
<tr>
<td>XLM-R(Hu et al. 2020)</td>
<td>10.2</td>
<td>12.4</td>
<td>16.3</td>
<td>19.1</td>
<td>13.3</td>
<td>14.3</td>
<td>24.3</td>
<td>19.8</td>
</tr>
<tr>
<td>Translate-train(Hu et al. 2020)</td>
<td>7.3</td>
<td>9.0</td>
<td>17.6</td>
<td>22.2</td>
<td>24.2</td>
<td>16.1</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>FILTER</td>
<td>6.0</td>
<td>5.2</td>
<td>7.3</td>
<td>15.7</td>
<td>9.2</td>
<td>8.7</td>
<td>19.7</td>
<td>16.3</td>
</tr>
</tbody>
</table>

Figure 3: Results on the dev set of PAWS-X, POS and MLQA with different $m$ and $k$ values.

Effect of Intermediate Fusion Layers Figure 3 shows the results on the dev sets with different $k$ and $m$ combinations (see Figure 2 for its definition). We perform experiments on PAWS-X, POS and MLQA, and consider them as representative datasets for classification, structured prediction and question answering tasks. For MLQA, performance is consistently improved with the number of intermediate fusion layers increasing, resulting in 2.6 points improvement from $k = 1$ to $k = 20$ when $m$ is set to 1. By contrast, the performance on PAWS-X and POS drops significantly when the number of intermediate fusion layers increases. For example, when $m$ is set to 1, accuracy decreases by 2.5/16.5 points from $k = 1$ to $k = 24$ on PAWS-X and POS datasets.

Effect of Local Transformer Layers As shown in Figure 3, for POS and MLQA, FILTER performs better when using more local transformer layers. For example, when $k$ is set to 10, we observe performance improvement by setting $m$ to 0, 1, 10 sequentially. On the contrary, for PAWS-X, when $k = 10$, the performance of setting $m = 0, 1$ is better than setting $m = 10$. This suggests that we should use more local layers for complex tasks such as QA and structured prediction, and fewer local layers for classification tasks.

Effect of Self-teaching Loss As can be seen from Table 4, for POS and NER, the use of self-teaching loss improves FILTER by 0.7 and 1.0 points. This confirms that self-teaching loss is very helpful in addressing the no-label issue for target languages. For classification and question answering tasks, we observe minor improvement, which is expected, as ground-truth labels are available for target languages, and adding the self-teaching loss only provides some label smoothing effect.

Cross-lingual Transfer Gap Table 6 shows analysis results of cross-lingual gap of different models, by calculating the difference between the performance on English test set and the average performance of other target languages. We observe that FILTER reduces the cross-lingual gap significantly among all tasks compared to mBERT, XLM-R and translate-train baselines. The transfer learning gap of FILTER is reduced by additional 2.5 and 10.6 points on average for classification and QA tasks, respectively, compared to the translate-train baseline respectively. For structured prediction tasks, the gap reduces even further, but a large gap still exists, indicating that this task demands stronger cross-lingual transfer.

Conclusion

We present FILTER, a new approach for cross-lingual language understanding that first encodes paired language input independently, then fuses them in the intermediate layers of XLM, and finally performs further language-specific encoding. An additional self-teaching loss is proposed for enhanced model training. By combining FILTER and self-teaching loss, we achieve new state of the art on the challenging XTREME and XGLUE benchmarks. Future work points to more effective ways of automatically discovering the best configuration of FILTER for different cross-lingual tasks.
References


