Freezing the Pivot for Triangular Machine Translation

Anonymous ACL submission

Abstract

Triangular machine translation is a special case of low-resource machine translation where the language pair of interest has limited parallel data, but both languages have abundant parallel data with a pivot language. Naturally, the key to triangular machine translation is the successful exploitation of such auxiliary data. In this work, we propose a transfer-learning-based approach that utilizes all types of auxiliary data. As we train auxiliary source-pivot and pivot-target translation models, we initialize some parameters of the pivot side with a pre-trained language model and freeze them to encourage both translation models to work in the same pivot language space, so that they can be smoothly transferred to the source-target translation model. Experiments show that our approach can outperform previous ones.

1 Introduction

Machine translation (MT) has achieved promising performance when large-scale parallel data is available. Unfortunately, the abundance of parallel data is largely limited to English, which leads to concerns on the unfair deployment of machine translation service across languages. In turn, researchers are increasingly interested in non-English-centric machine translation approaches (Fan et al., 2021).

Triangular MT (Kim et al., 2019; Ji et al., 2020) has the potential to alleviate some data scarcity conditions when the source and target languages both have a good amount of parallel data with a pivot language (usually English). Kim et al. (2019) have shown that transfer learning is an effective approach to triangular MT, surpassing generic approaches like multilingual MT.

However, previous works have not fully exploited all types of auxiliary data (Table 1). For example, it is reasonable to assume that the source, target, and pivot language all have much monolingual data because of the notable size of parallel data between source-pivot and pivot-target.

<table>
<thead>
<tr>
<th>approach</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
<th>X-Z</th>
<th>Z-Y</th>
<th>X-Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>no transfer</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>pivot translation</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>step-wise pre-training</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>shared target transfer</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>shared source transfer</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>simple triang. transfer</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>triangular transfer</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table 1: Data usage of different approaches (Section 3.2). X, Y, and Z represent source, target, and pivot language, respectively. Our triangular transfer uses all types of data.

In this work, we propose a transfer-learning-based approach that exploits all types of auxiliary data. During the training of auxiliary models on auxiliary data, we design parameter freezing mechanisms that encourage the models to compute the representations in the same pivot language space, so that combining parts of auxiliary models gives a reasonable starting point for finetuning on the source-target data. We verify the effectiveness of our approach with a series of experiments.

2 Approach

We first present a preliminary approach that is a simple implementation of our basic idea, for ease of understanding. We then present an enhanced version that achieves better performance. For notation purpose, we use X, Y, and Z to represent source, target, and pivot language, respectively.

2.1 Simple Triangular Transfer

We show the illustration of the preliminary approach in Figure 1, called simple triangular transfer. In Step (1), we prepare a pre-trained language model (PLM) with the pivot language monolingual data. We consider this PLM to define a representation space for the pivot language, and we would like subsequent models to stick to this representation.
We found that simple triangular transfer attains a limitation of simple triangular transfer is that it does not utilize monolingual data of the source and target languages. A naive way is to prepare source and target PLMs and use them to initialize source-pivot encoder and pivot-target decoder, respectively. However, this leads to marginal improvement for the final source-target translation performance. This is likely because the source, target, and pivot PLMs are trained independently, so their representation spaces are isolated.

Therefore, we intend to train source and target PLMs in the pivot language space as well. To this end, we design another initialization and freezing step inspired by (Zhang et al., 2021), as shown in Figure 2. In this illustration, we use BART as the PLM. Step (2) is the added step of preparing BART models in the source and target languages. As the BART body parameters are inherited from the pivot language BART and frozen, the source and target language BART embeddings are trained to lie in the pivot language space. Then in Step (3), every part of the translation models can be initialized in the pivot language space. Again, we freeze parameters in the pivot language side to ensure the representations do not drift too much.

2.3 Freezing Strategy

There are various choices when we freeze parameters in the pivot language side of the source-pivot and pivot-target translation models. Take the encoder of the pivot-target translation model as the example. In one extreme, we can freeze the embeddings only; this is good for the optimization of pivot-target translation, but may result in a space that is far away from the pivot language space given by the pivot PLM. In the other extreme, we can freeze the entire encoder, which clearly hurts the pivot-target translation performance. This is hence a trade-off. We experiment with multiple freezing strategies between the two extremes, i.e., freezing a given number of layers. We always ensure that the number of frozen layers is the same for the decoder of the source-pivot translation model.

Besides layer-wise freezing, we also try component-wise freezing inspired by (Li et al., 2021). In their study, they found that some components like layer normalization and decoder cross attention are necessary to finetune, while others can be frozen. In particular, we experiment with three strategies based on their findings of the most effective ones in their task. These strategies apply to Step (3) of triangular transfer.

LNA-E,D: All layer normalization, encoder self attention, decoder cross attention can be finetuned.
3 Experiments

3.1 Setup

We conduct experiments on French (Fr) $\rightarrow$ German (De) translation, with English (En) as the pivot language. The evaluation metric is computed by SacreBLEU\textsuperscript{1} (Post, 2018). Our translation model is Transformer base (Vaswani et al., 2017). Further details can be found in the appendix.

\begin{table}[h]
\centering
\begin{tabular}{|c|c|}
\hline
\textbf{approach} & \textbf{BLEU} \\
\hline
no transfer & 13.49 \\
pivot translation through no transfer & 18.99 \\
step-wise pre-training & 18.49 \\
shared target transfer & 18.88 \\
shared source transfer & 18.89 \\
triangular transfer & 19.91 \\
\hline
\end{tabular}
\caption{Comparison with baselines. Our triangular transfer is significantly better ($p < 0.01$) than baselines by paired bootstrap resampling (Koehn, 2004).}
\end{table}

Others are frozen.

\textbf{LNA-D}: All encoder parameters, decoder layer normalization and cross attention can be finetuned.

\textbf{LNA-e,D}: Use LNA-D when training the source-pivot model. When training the pivot-target model, freeze encoder embeddings in addition to LNA-D.

3.2 Baselines

We compare with several baselines as follows.

\textbf{No transfer}: This baseline directly trains on the source-target parallel data.

\textbf{Pivot translation}: Two-pass decoding by source-pivot and pivot-target translation.

\textbf{Step-wise pre-training}: This is one of the approaches in (Kim et al., 2019) which is simple and robust. It trains a source-pivot translation model and uses the encoder to initialize the encoder of a pivot-target translation model. In order to make this possible, these two encoders need to use a shared source-pivot vocabulary. Then the pivot-target translation model is trained while keeping its encoder frozen. Finally the model is finetuned on source-target parallel data.

\textbf{Shared target dual transfer}: Dual transfer (Zhang et al., 2021) is a general transfer learning approach to low-resource machine translation. When applied to triangular MT, it cannot utilize both source-pivot and pivot-target parallel data. Shared target dual transfer uses pivot-target auxiliary translation model and does not exploit source-pivot parallel data.

\textbf{Shared source dual transfer}: The shared source version uses source-pivot translation model for transfer and does not exploit pivot-target parallel data.

3.3 Main Results

We present the performance of our approach and the baselines in Table 2. The no transfer baseline

\textsuperscript{1}SacreBLEU signature: BLEU+case.mixed+numrefs.1+smooth.exp+tok.13a+version.1.4.12.
performs poorly because it is trained on a small amount of parallel data. The other baselines perform much better. Among them, pivot translation attains the best performance in terms of BLEU, at the cost of doubled latency. Our approach can outperform all the baselines.

### 3.4 The Effect of Freezing Strategies

From Table 3, we can observe the effect of different freezing strategies. For layer-wise freezing, we see a roughly monotonic trend of the Fr-En and En-De performance with respect to the number of frozen layers: The more frozen layers, the lower their BLEU scores. However, the best Fr-De performance is achieved with \( L = 3 \). This indicates the trade-off between the auxiliary models’ performance and the pivot space anchoring. For component-wise freezing, the Fr-En and En-De performance follows a similar trend, but the Fr-De performance that we ultimately care about is not as good.

### 3.5 Using Monolingual Data

Table 4 shows the effect of different ways of using monolingual data. The naive way is to prepare PLMs with monolingual data and initialize the encoder or decoder where needed. For pivot translation, this is known as \textit{BERT2BERT} (Rothe et al., 2020) for the source-pivot and pivot-target translation models. For dual transfer, parts of the auxiliary models can be initialized by PLMs (e.g., for shared target transfer, the pivot-target decoder is initialized). For Step (2) in simple triangular transfer, we can also initialize the pivot-target decoder and source-pivot encoder with PLMs. However, none of the above methods shows clear improvement. This is likely because these methods only help the auxiliary translation models to train, which is not necessary as they can be trained well with abundant parallel data already. In contrast, our design of Step (2) in triangular transfer additionally helps the auxiliary translation models to stay in the pivot language space.

### 3.6 Pivot-Based Back-Translation

Following (Kim et al., 2019), we generate synthetic parallel Fr-De data with pivot-based back-translation (Bertoldi et al., 2008). Results in Table 5 show that triangular transfer and dual transfer clearly outperform the no transfer baseline.

### 4 Conclusion

In this work, we propose a transfer-learning-based approach that utilizes all types of auxiliary data, including both source-pivot and pivot-target parallel data, as well as involved monolingual data. We investigate different freezing strategies for training the auxiliary models to improve source-target translation, and achieve better performance than previous approaches.
References


A Data and Preprocessing

We gather data from WMT, shown in Tables 6 and 7.

The preprocessing pipeline includes punctuation normalization, tokenization, and deduplication. Each language is encoded with byte pair encoding (BPE) (Sennrich et al., 2016) with 32k merge operations. The BPE codes and vocabularies are learned on each language’s monolingual data, and then used to segment parallel data. Sentences with more than 128 subwords are removed. Parallel sentences are cleaned with length ratio 1.5 (length counted by subwords).
Table 6: Parallel data source.

<table>
<thead>
<tr>
<th>lang.</th>
<th>source</th>
<th>train</th>
<th>dev</th>
<th>test</th>
</tr>
</thead>
<tbody>
<tr>
<td>En-De</td>
<td>WMT 2019</td>
<td>Europarl v9, News Commentary v14, Document-split Rapid corpus</td>
<td>newstest2011</td>
<td>newstest2012</td>
</tr>
<tr>
<td>Fr-De</td>
<td>WMT 2019</td>
<td>News Commentary v14, newstest2008-2010</td>
<td>newstest2011</td>
<td>newstest2012</td>
</tr>
</tbody>
</table>

Table 7: Monolingual data source.

<table>
<thead>
<tr>
<th>lang.</th>
<th>source</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>En</td>
<td>WMT 2018</td>
<td>News Crawl 2014-2017</td>
</tr>
<tr>
<td>De</td>
<td>WMT 2021</td>
<td>100m subset from WMT 2021</td>
</tr>
</tbody>
</table>

Table 8: Training data statistics.

<table>
<thead>
<tr>
<th>language code</th>
<th># sentence (pair)</th>
</tr>
</thead>
<tbody>
<tr>
<td>En-De</td>
<td>3.1m</td>
</tr>
<tr>
<td>Fr-En</td>
<td>29.5m</td>
</tr>
<tr>
<td>Fr-De</td>
<td>247k</td>
</tr>
<tr>
<td>En</td>
<td>93.9m</td>
</tr>
<tr>
<td>De</td>
<td>100.0m</td>
</tr>
<tr>
<td>Fr</td>
<td>44.6m</td>
</tr>
</tbody>
</table>

The final training data statistics is shown in Table 8.

B Hyperparameters

Our implementation is based on fairseq (Ott et al., 2019). We share decoder input and output embeddings (Press and Wolf, 2017). The optimizer is Adam. Dropout and label smoothing are both set to 0.1. The batch size is 6,144 per GPU and we train on 8 GPUs. The peak learning rate is $5 \times 10^{-4}$ for the no transfer baseline and auxiliary models, $1 \times 10^{-4}$ for the Fr→De model of stepwise pre-training and dual transfer, and $7 \times 10^{-5}$ for the Fr→De model of triangular transfer. The learning rate warms up for 4,000 steps, and then follows inverse square root decay. Early stopping happens when the development BLEU does not improve for 10 epochs.

RoBERTa and BART models use exactly the same architecture as Transformer base. The mask ratio is 15%. The batch size is 256 sentences per GPU, and each sentence contains up to 128 tokens. The learning rate warms up for 10,000 steps to the peak $5 \times 10^{-4}$, and then follows polynomial decay. They are trained for 125k steps.