Cross-lingual transfer learning considering word order difference

Haosen Zhan Jin Sakuma
The University of Tokyo
{hos, jsakuma}@tkl.iis.u-tokyo.ac.jp

Naoki Yoshinaga Masashi Toyoda
Institute of Industrial Science,
The University of Tokyo
{ynaga, toyoda}@iis.u-tokyo.ac.jp

1 Introduction

Although deep neural networks (DNNs) exhibit a remarkable performance in various natural language processing tasks, we cannot enjoy the true potential of DNNs in most of languages. This is because we do not have a large-scale annotated corpora to train the over-parametarized DNNs for the task in the target language.

The cross-lingual transfer learning mitigates this problem by training models on annotated data in the resource-rich (source) language (typically, English) so that the resulting models can be applied to the target language. We will hereafter refer to this setting as "zero-shot." The performance of zero-shot transfer learning has been drastically improved by the pre-trained multilingual models (multilingual-BERT [1]). These models are pre-trained on massive multilingual raw corpora and then fine-tuned to the target task in the source language. The resulting models are then applied in the target language. Meanwhile, it has been reported that the differences in word orders harm the performance of cross-lingual transfer between distant languages. Ahmad el al. [2] observed a substantial gap in performance across different target languages, especially when word orders in the source and target language differ.

To mitigate this challenge in cross-lingual transfer across distant languages, researchers explored two major approaches. Xia et al. [3] proposed to use dependency CNN to improve the performance of a multilingual model based on cross-lingual word embeddings in English and Chinese sentence-level classification. Liu et al. [4] fixed the positional encoding layer of BERT to avoid overfitting to the word order of the source language in fine-tuning. However, the former approach is not compatible with pre-trained multilingual models, and the latter approach is limited to the embedding layer and could not prevent the overfitting happens in transformer layers.

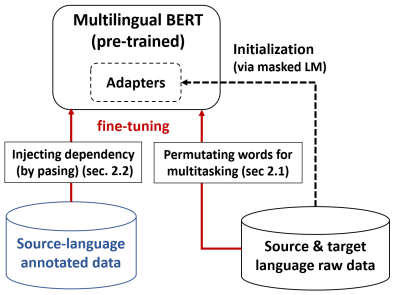

Figure 1 The overview of our framework on cross-lingual transfer learning considering word order difference.

In this study, we introduce methods to explicitly mitigate the problem of order differences in the pre-trained multilingual model for the zero-shot scenario (§ 2). Concretely, we propose two approaches to capture word orders in the fine-tuning of multilingual-BERT (mBERT): 1) multitask learning with word permutation detection task and 2) incorporating order-insensitive dependency information (Figure 1).

We evaluate our methods on mutilingual part-of-speech tagging and named entity recognition (§ 3). Experimental results confirmed that the effectiveness of our methods.

2 Proposal

In zero-shot learning, mBERT fine-tuned in the source language can overfit to the word order of the source language and fail to encode target languages with drastic order differences from the source, resulting in the loss in the performance. Figure 1 shows an overview of our framework for zero-shot cross-lingual transfer learning. In the following, we will introduce three methods to mitigate the order difference in zero-shot cross-lingual transfer learning.

2.1 Multitasking with permutation detection

To prevent the model from overfitting, we first introduce a self-supervised auxiliary task of detecting word permuta-

tion during fine-tuning in the source language. We expect that this task will identically help the model encode word orders of the source and target languages.

For each mini-batch, we sample raw corpora from both source and target languages. We randomly generate word-level permutation within one sentence with a certain probability,[1)] and the model will learn to distinguish whether each token has been permutated or not. Concretely, a token-level binary classifier is added on the top of the encoder to predict whether each token is permutated. The resulting loss is computed as

$$\mathcal{L}_{total} = \mathcal{L}_{task} + \alpha * \mathcal{L}_{perm} \quad (1)$$

where $\mathcal{L}_{task}$ and $\mathcal{L}_{perm}$ are losses of the target task and permutation detection task, respectively, and α is the hyper-parameter that represents the weight for the permutation loss, which is whether kept a constant or exponentially increased/decreased across training. We use the first subword unit in each word to calculate the loss.

2.2 Injecting dependency information

Although the multitasking with permutation detection can be applicable to any languages as long as raw corpora are available, this auxiliary task is self-supervised and may not convey sufficient supervision. Therefore, assuming a dependency parser for the target language, we next propose to explicitly inject dependency information into the transformer network to prevent the model overfit to the surface orders in the source language.

2.2.1 Injecting dependency heads

We first propose to incorporate adapters [5] into the transformer layers that aggregate each token with its dependency head. This injection is meant to promote the model to utilize language-independent information without relying on surface word orders that are specific to the source language (Figure 2).

In what follows, we explain how the output of the adapter module, x_i^{out}, is computed given the hidden state of a token x_i. Firstly, for each *subword* i, we obtain the dependency head *word* token via a dependency parser. And let $head_i$ be the first *subword* token of the head *word* token. Then, by the following function, we aggregate the outputs of feed-forward network, x_i and x_i^{head}.

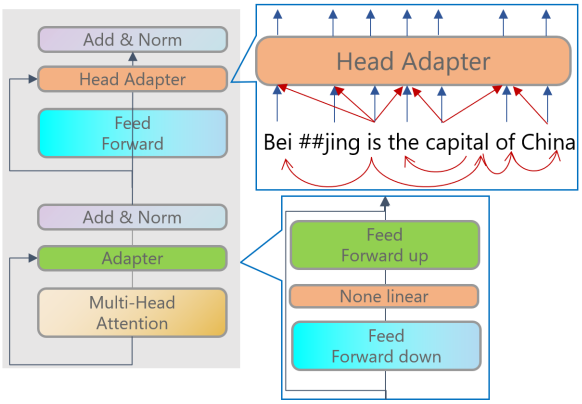

Figure 2 Transformer layer with head-adapter

$$x_i^{out} = \mathrm{FFN}_{up}(\mathrm{ReLU}(\mathrm{FFN}_{tail}(x_i) + \mathrm{FFN}_{dep}(x_i^{head}))) \quad (2)$$

where FFN_{up}, FFN_{tail}, and FFN_{dep} are single-layer feed-forward networks.

If we randomly initialize the adapter modules before fine-tuning, they are only exposed to the examples in the source language and could be overfitted. Henceforth, we first trained the adapter modules on the masked language model (MLM) task on multilingual raw corpora. Simply training the entire model on MLM task requires parsed multilingual raw corpora. However, obtaining such corpora costs a significant amount of human labour and time. Therefore, we first exploited the adapter structure proposed by Houslby [5], and then the optimized parameters are copied to our adapter modules. Concretely, we initialize both FFN_{dep} and FFN_{tail} by the trained parameters of FFN_{down}, and the rest parts are directly copied. We only inserted adapters in the first two layers of BERT, and during task fine-tuning, both adapter modules and BERT layers are trained.

2.2.2 Integrating dependency distance

Despite the high precision of the multilingual parser, the previously introduced approach introduce in § 2.2.1 can suffer from pipeline error propagation. Furthermore, despite our initialization method, the resulting adapter modules can still overfit to the source language during fine-tuning. Besides the hard encoding of dependency head information, we propose an alternative adapter module to integrate continuous representation corresponding to to-

1) In the experiments, we set this probability to 0.2.

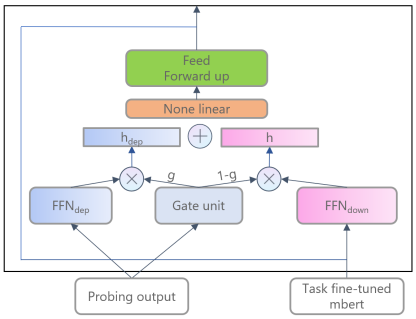

Figure 3 Transformer layer with dependency distance based adapters

kens' dependency information obtained by syntactic probing [6]. Concretely, we employ structural probing, which projects hidden states of fixed mBERT obtained by linear projection trained to recreate the distances in the dependency trees.

Here, we let x_i be the output of the feed-forward network in a transformer layer corresponding to the ith token. And let x_i^{dep} be the projected representation by structural probing corresponding to the ith token. We aggregate the h_i and h_i^{dep} to compute the output of the adapter module x_i^{out} by following

$$g_i = \sigma\left(\mathrm{FFN}_{gate}(x_i^{dep})\right) \tag{3}$$

$$h_i^{dep} = g_i \otimes \mathrm{FFN}_{dep}(x_i^{dep}) \tag{4}$$

$$h_i = (1 - g_i) \otimes \mathrm{FFN}_{down}(x_i) \tag{5}$$

$$x_i^{out} = \mathrm{FFN}_{up}(\mathrm{ReLU}(h_i^{dep} + h_i)) \tag{6}$$

where FFN_{dep}, FFN_{down}, and FFN_{gate} are feed forward layers for transforming hidden states and corresponding dependency features, g is the gate that represents how much the model considers the dependency information for the particular token, and $\otimes$ represents the element-wise multiplication. We expect that the gate module will dynamically decide whether or not to absorb the extracted dependency feature. The combined features are then projected back to the hidden dimension of mBERT by FFN_{up}.

For initialization of adapters, this approach is different from § 2.2.1 that it takes raw corpora without parsed ones as input. Hence, we use raw corpora and the MLM task to initialize all the parameters. The rest of the details work the same as § 2.2.1.

3 Experiments

We conduct experiments on the part of speech tagging (POS) task and named entity recognition (NER) task to evaluate the effectiveness of our proposed methods.

3.1 Experiment settings

Data pre-processing Injecting dependency information (§ 2.2.1) requires dependency parsers for the source and target languages. For this purpose, we leveraged Udify [7]. We use MeCab[2] and Jieba[3] to tokenize Japanese and Chinese raw corpora for permutation in § 2.1. We followed a sliding window approach [8] to process long sequences.

Models to compare To evaluate the effectiveness of our proposal, we compared the following models:

BASE is a baseline model that fine-tunes the mBERT model on the training set in the source language.

BASE+FIX-POS-EMB fixes the positional embeddings of mBERT during fine-tuning [4].

BASE+FIX-EMB fixes the both positinal and word embeddings of mBERT during fine-tuning.

We applies our methods to the above three models.

+PERM employs the multi-task learning with the auxiliary task of word permutation detection (§ 2.1).

+HEAD inserts the adapter modules that injects dependency head (§ 2.2).

+DEP-DIS inserts the adapter modules that integrate dependency distance (§ 2.2.2).

COMPARE Stands for control group for DEP-DIS, but directly taking hidden layers output without projecting to dependency-distance-space

Training All of the models were initialized from weights of "bert-base-multilingual-cased" acquired from huggingface tranformers[4] [9] and trained on the training set of the source language (English). During the fine-tuning, the checkpoints are saved every epoch, and one that performed the best on the development set in the target language is used for testing.

For models based on adapters (**+HEAD** and **+DEP-DIS**), the dimension of the hidden state within the adapters is set to 384, and we sampled 100k sentences for the initialization of adapters by the MLM task (§ 2.2.1). For the syntactic probing (**+DEP-DIS**), we trained a projection from hidden states of the 7th layer from BERT to dependencies subspace.

2) https://taku910.github.io/mecab/
3) https://github.com/fxsjy/jieba
4) https://github.com/huggingface/transformers

setting	en(✓)	es(✓)	fi(✓)	fr(✓)	et(✓)	de(✓)	lv(✓)	ru(✓)	zh(✓)	hi(✗)	ja(✗)	la(✗)
BASE	**96.99**	86.37	85.49	89.21	85.35	90.56	81.67	88.69	67.57	69.25	50.51	**71.76**
+PERM	-	86.36	85.26	89.27	85.52	90.93	81.76	88.58	66.47	**71.17**	53.97	71.33
BASE+FIX-POS-EMB	96.98	**86.51**	85.46	89.34	85.46	90.54	81.64	**88.87**	67.52	69.58	50.70	**71.76**
+PERM	-	86.27	85.24	89.24	85.56	90.85	**81.77**	88.59	66.63	71.00	54.88	71.44
BASE+FIX-EMB	96.88	86.50	**85.52**	89.32	85.42	90.57	81.60	88.77	**67.72**	69.07	50.55	**71.76**
+PERM	-	86.24	85.48	**89.42**	**85.80**	**90.96**	81.57	88.37	66.63	70.05	**55.53**	70.68

Table 1 Result for POS (accuracy). All settings were reported on 5 different random seeds. ✓ stands for languages with canonical order of Subject-Verb-Object (SOV) and ✗ stands for non-SOV languages.

Setting	en	es	nl	de
BASE	91.01	74.87	78.94	70.07
+PERM	-	74.13	79.76	71.54
+HEAD	90.98	75.38	79.78	71.62
+DEP-DIS	-	74.46	80.14	72.03
+DEP-DIS+PERM	-	74.22	80.98	72.35
BASE+FIX-POS-EMB	91.01	73.76	78.72	70.53
+PERM	-	74.71	79.24	71.63
+HEAD	91.05	**75.56**	79.77	71.86
+DEP-DIS	-	74.88	80.02	71.86
+DEP-DIS+PERM	-	74.80	80.76	72.09
BASE+FIX-EMB	90.96	75.31	79.86	71.05
+PERM	-	74.85	79.76	72.31
+HEAD	**91.12**	73.60	80.56	70.32
+DEP-DIS	-	74.42	**81.26**	71.54
+DEP-DIS+PERM	-	74.67	80.56	**72.96**
COMPARE+FIX-EMB	-	74.67	80.96	72.13

Table 2 Entity level F1 score for NER. All settings were averaged on 5 different random seeds

3.2 Part Of Speech tagging

In the POS tagging task, we leveraged the "upos" filed from Universal Dependency v2.3 treebanks,[5] the treebanks selection follows [2]. Since POS is usually solved as a subtask of dependency parsing, it is impractical to have data for parsing without POS annotated. Therefore, we only test the impact of **+PERM** in this section.

Table 1 showed the results on POS tagging. As our expectation, multi-tasking with the permutation task (**+PERM**) yielded solid improvement over **BASE** in German and Japanese (ja). In contrast, the results in Latin (la) and Chinese (zh) were opposed to our expectations. It worth mention that mBERT leveraged a brutal way of tokenization on Japanese, which could be one reason for the abnormal pattern in our results.

5) https://universaldependencies.org

3.3 Named Entity Recognition

For the NER task, we used CoNLL 2002 and 2003 NER shared task [10] containing English (en), Spanish (es), Dutch (nl), and German (de). The labels were converted into IOB2 [11] format, and we use token-level classifiers for NER prediction. The hidden state of the first subword unit of each token is used for the prediction.

Table 2 shows the results on NER. We observed that the performance in Dutch benefited from injecting dependencies the most. Among all the languages, Dutch shares the most common features with English. This is opposed to our previous assumption that leveraging dependency helps distance cross-lingual transfer. In German, the multi-task learning with the word permutation detection (**+PERM**) improved models' performance with robust increment, comparing all the settings with or without **+PERM**. German is relatively "different" from English among the target languages, showing that permutation detection is profitable for the NER task.

4 Conclusions

In this paper, we investigate three methods of mitigating order differences in source and target languages in the mBERT-based cross-lingual transfer. We observed that our proposal yields improvements in token-level classification, yet the gain is limited. In our future research, we plan to 1) leverage fine-grained dependency features and 2) find an auxiliary task that is purely related to word order, baring that current proposal permutation detection is not independent of the understanding of semantic.

Acknowledgement

This work was supported by JST CREST Grant Number JPMJCR19A4, Japan.

References

[1]Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, Minnesota, June 2019.

[2]Wasi Ahmad, Zhisong Zhang, Xuezhe Ma, Eduard Hovy, Kai-Wei Chang, and Nanyun Peng. On difficulties of cross-lingual transfer with order differences: A case study on dependency parsing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, Minnesota, June 2019.

[3]Yandi Xia, Zhongyu Wei, and Yang Liu. An efficient cross-lingual model for sentence classification using convolutional neural network. In *Proceedings of the ACL 2016 Student Research Workshop*, Berlin, Germany, August 2016.

[4]Zihan Liu, Genta Indra Winata, Samuel Cahyawijaya, Andrea Madotto, Zhaojiang Lin, and Pascale Fung. On the importance of word order information in cross-lingual sequence labeling. In *Proceedings of 35th AAAI Conference on Artificial Intelligence (AAAI-2021)*.

[5]Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *ICML*, pp. 2790–2799, 2019.

[6]Ethan A. Chi, John Hewitt, and Christopher D. Manning. Finding universal grammatical relations in multilingual BERT. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online, July 2020.

[7]Dan Kondratyuk and Milan Straka. 75 languages, 1 model: Parsing Universal Dependencies universally. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China, November 2019.

[8]Shijie Wu and Mark Dredze. Beto, bentz, becas: The surprising cross-lingual effectiveness of bert. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019.

[9]Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Online, October 2020.

[10]Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pp. 142–147, 2003.

[11]Erik F. Tjong Kim Sang and Jorn Veenstra. Representing text chunks. In *Ninth Conference of the European Chapter of the Association for Computational Linguistics*, Bergen, Norway, June 1999.

A hyper-parameters

parameter	value
adam epsilon	1e-8
learning rate	2e-5
max epoch	5
batch size	16
dropout rate	0.2
hidden perm	384
dropout perm	0.2
learning rate perm	0.5

Table 3 hyper-parameters for POS.

parameter	value
adam epsilon	1e-8
learning rate	2e-5
max epoch	6
batch size	16
dropout rate	0.2
hidden perm	384
reduction factor	2
dropout perm	0.2
learning rate perm	0.1

Table 4 hyper-parameters for NER.