COSY: COunterfactual SYntax for Cross-Lingual Understanding

Sicheng Yu[1], **Hao Zhang**[2,3], **Yulei Niu**[2], **Qianru Sun**[1], **Jing Jiang**[1]
[1]Singapore Management University, Singapore
[2]Nanyang Technological University, Singapore
[3]Agency for Science, Technology and Research, Singapore
scyu.2018@phdcs.smu.edu.sg, hao007@e.ntu.edu.sg
yn.yuleiniu@gmail.com, {qianrusun,jingjiang}@smu.edu.sg

Abstract

Pre-trained multilingual language models, *e.g.*, multilingual-BERT, are widely used in cross-lingual tasks, yielding the state-of-the-art performance. However, such models suffer from a large performance gap between source and target languages, especially in the zero-shot setting, where the models are fine-tuned only on English but tested on other languages for the same task. We tackle this issue by incorporating language-agnostic information, specifically, universal syntax such as dependency relations and POS tags, into language models, based on the observation that universal syntax is transferable across different languages. Our approach, named COunterfactual SYntax (COSY), includes the design of SYntax-aware networks as well as a COunterfactual training method to implicitly force the networks to learn not only the semantics but also the syntax. To evaluate COSY, we conduct cross-lingual experiments on natural language inference and question answering using mBERT and XLM-R as network backbones. Our results show that COSY achieves the state-of-the-art performance for both tasks, without using auxiliary dataset.[1]

1 Introduction

With the emergence of BERT (Devlin et al., 2019), large-scale pre-trained language models have become an indispensable component in the solutions to many natural language processing (NLP) tasks. Recently, large-scale multilingual transformer-based models, such as mBERT (Devlin et al., 2019), XLM (Lample and Conneau, 2019) and XLM-R (Conneau et al., 2020a), have been widely deployed as backbones in cross-lingual NLP tasks (Wu and Dredze, 2019; Pires et al., 2019; Keung et al., 2019). However, these models trained on a single resource-rich language, *e.g.*, English, all suffer from a large drop of performance when tested on different target languages, *e.g.*, Chinese and German—where the setting is called *zero-shot cross-lingual transfer*. For example, on the XQUAD dataset, mBERT achieves a 24 percentage points lower exact match score on the target language Chinese than on the training language English (Hu et al., 2020). This indicates that this model has seriously overfitted English.

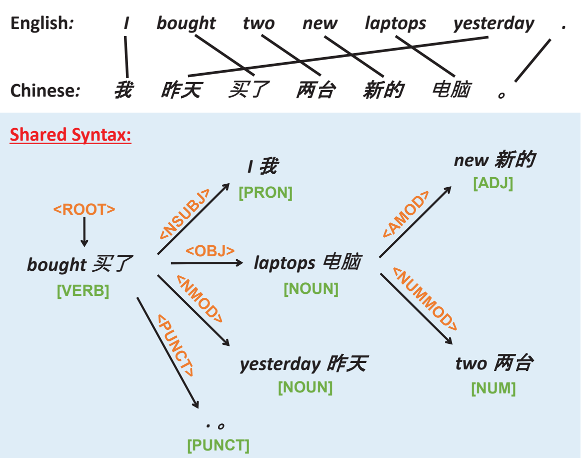

Figure 1: Examples of two sentences in English and Chinese that have the same meaning and share the same syntax in the format of dependency relations and POS tags.

An intuitive way to tackle this is to introduce language-agnostic information—the most transferable feature across languages, which is lacking in existing multilingual language models (Choenni and Shutova, 2020). In our work, we propose to exploit reliable language-agnostic information—syntax in the form of universal dependency relations and universal POS tags (de Marneffe et al., 2014; Nivre et al., 2016; Zhou et al., 2019, 2021). As illustrated in Figure 1, the sentences in Chinese and English share the same meaning but have differ-

[1] Our code is publicly available on GitHub: https://github.com/PluviophileYU/COSY

Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics
and the 11th International Joint Conference on Natural Language Processing, pages 577–589
August 1–6, 2021. ©2021 Association for Computational Linguistics

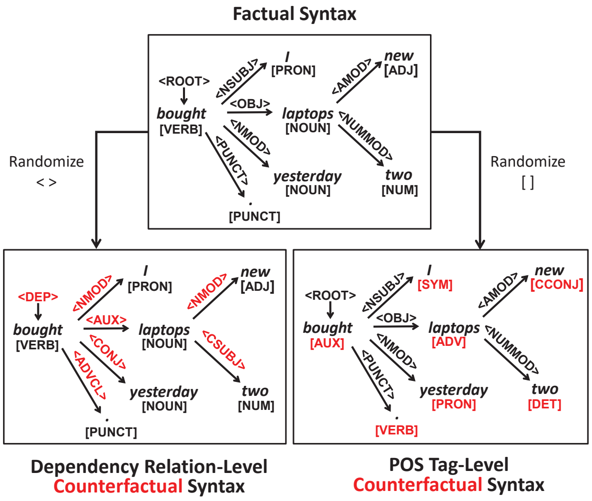

Figure 2: Illustration of counterfactual syntax generation. Red color highlights the modified syntax with randomized labels.

ent word orders. The order difference hampers the transferability between English and Chinese in conventional language models (with sequential words as input). In contrast, it is clear from Figure 1 that the two sentences share identical dependency relations and POS tags. Thus, we can incorporate such universal syntax[2] information to enhance the transferability across different languages. To achieve this learning objective in deep models, we design syntax-aware networks that incorporate the encodings of dependency relations and POS tags into the encoding of semantics.

However, we find that empirically the conventional attention-based incorporation of syntax, *e.g.*, relational graph attention networks (Ishiwatari et al., 2020), has little effect on improving the model. One possible reason is that the learning process may be dominated by the pre-trained language models due to their strength in semantic representation learning, which leads to an overfitted model. This raises the question of *how to induce the model to focus more on syntax while maintaining its original capability of representing semantics?* To this end, we propose a novel COunterfactual SYntax (COSY) method, inspired by causal inference (Roese, 1997; Pearl et al., 2009) and contrastive learning (He et al., 2020).

The intuition behind COSY is to create copies of training instances with their syntactic features altered (see the "counterfactual" syntax in Figure 2), and to force the encodings of the counterfactual instances to be different from the encodings of their corresponding factual instances. In this way, the model would learn to put more emphasis on the syntactic information when learning how to encode an instance, and such encodings are likely to perform well across languages.

We evaluate our COSY method on both question answering (QA) and natural language inference (NLI) under cross-lingual settings. Experimental results show that, without using any additional data, COSY is superior to the state-of-the-art methods.

Contributions: 1) we develop a syntax-aware network that incorporates transferable syntax in language models; 2) we propose a novel counterfactual training method that addresses the technical challenge of emphasizing syntax; and 3) extensive experiments on three benchmarks demonstrate the effectiveness of our method for cross-lingual tasks.

2 Related Work

Cross-lingual Transfer. Large-scale pre-trained language models (Devlin et al., 2019; Liu et al., 2019) have achieved sequential success in various natural language processing tasks. Recent studies (Lample and Conneau, 2019; Conneau et al., 2020a) extend the pre-trained language models to multilingual tasks and demonstrate their prominent capability on cross-lingual knowledge transfer, even under zero-shot scenario (Wu and Dredze, 2019; Pires et al., 2019; Hsu et al., 2019).

Motivated by the success of multilingual language models on cross-lingual transfer, several works explore how these models work and what their bottleneck is. On the one hand, some studies find that the shared sub-words (Wu and Dredze, 2019; Dufter and Schütze, 2020) and the parameters of top layers (Conneau et al., 2020b) are crucial for cross-lingual transfer. On the other hand, the bottleneck is attributed to two issues: (i) catastrophic forgetting (Keung et al., 2020; Liu et al., 2020), where knowledge learned in the pre-training stage is forgotten in downstream fine-tuning; (ii) lack of language-agnostic features (Choenni and Shutova, 2020; Zhao et al., 2020) or linguistic discrepancy between the source and the target languages (Wu and Dredze, 2019; Lauscher et al., 2020). In this work, we aim to tackle zero-shot and few-shot cross-lingual transfer by focusing on the second issue.

Existing works can be roughly divided into two groups. The first proposes to modify the lan-

[2] In the rest of this paper, syntax denotes universal syntax for simplicity.

guage model by aligning languages with parallel data (Zhao et al., 2020) or strengthening sentence-level representation (Wei et al., 2020). The second group focuses on the learning paradigm for fine-tuning on downstream tasks. For instance, some methods adopt meta-learning (Nooralahzadeh et al., 2020; Yan et al., 2020) or intermediate tasks training (Phang et al., 2020) to learn cross-lingual knowledge. Our COSY belongs to the second group and fills the blank of using the syntactic information in zero-shot (few-shot) cross-lingual understanding.

Counterfactual Analysis. Counterfactual analysis aims to evaluate the causal effect of a variable by considering its counterfactual scenario. Counterfactual analysis has been widely studied in epidemiology (Rothman and Greenland, 2005) and social science (Steel, 2004). Recently, counterfactual reasoning has motivated studies in applications.

In the community of computer vision, counterfactual analysis has been successfully applied in explanation (Goyal et al., 2019a,b), long-tailed classification (Tang et al., 2020a), scene graph generation (Tang et al., 2020b), and visual question answering (Chen et al., 2020; Niu et al., 2020; Abbasnejad et al., 2020).

In the community of natural language processing, counterfactual methods are also emerging recently in text classification (Choi et al., 2020), story generation (Qin et al., 2019), dialog systems (Zhu et al., 2020), gender bias (Vig et al., 2020; Shin et al., 2020), question answering (Yu et al., 2020), and sentiment bias (Huang et al., 2020). To the best of our knowledge, we are the first to conduct counterfactual analysis in cross-lingual understanding. Different from previous works (Zhu et al., 2020; Qin et al., 2019) that generate word-level or sentence-level counterfactual samples, our counterfactual analysis dives into syntax level that is more controllable than text and free from complex language generation module.

3 COSY: COunterfactual SYntax

COSY aims to leverage the syntactic information, *e.g.*, dependency relations and POS tags, to increase the transferability of cross-lingual language models. Specifically, COSY implicitly forces the networks to learn to encode the input not only based on semantic features but also based on syntactic features through syntax-aware networks and a counterfactual training method.

As illustrated in Figure 3, COSY consists of three branches with each branch based on syntax-aware networks (SAN) indicated by a distinct color. The main branch (in black) is the factual branch that uses factual syntax as input. The red and blue branches are counterfactual branches using counterfactual dependency relations and counterfactual POS tags as input, respectively. The counterfactual training method guides the black branch to put more emphasis on syntactic information with the help of other two branches. Note that the red and blue branches work for counterfactual training, and only the prediction from the black branch is used in testing.

Below, we first elaborate the modules of SAN in Section 3.1, and then introduce the counterfactual training method in Section 3.2.

3.1 Syntax-Aware Networks (SAN)

As shown in Figure 3, SAN contains four major modules: a set of feature extractors, a relational graph attention network (RGAT), fusion projection, and a classifier. In this section, we use the route in the black branch as an example to elaborate each module. The set of feature extractors include three components: a pre-trained language model, a dependency graph constructor and a POS tags extractor.

Pre-trained Language Model. Following previous work (Hu et al., 2020), we deploy a pre-trained multi-lingual language model, *e.g.*, mBERT (Devlin et al., 2019), to encode each input sentence into contextual features. Given a sequence of tokens with a length of S, we denote the derived contextual features as $\mathbf{H}=[\mathbf{h}_1, ..., \mathbf{h}_S] \in \mathbb{R}^{S\times d}$, where d is the dimensionality of each hidden vector.

Dependency Graph Constructor. We use it to construct the (factual) dependency graph for each input sentence. In this work, the Stanza toolkit (Qi et al., 2020) is used to extract the universal dependency relations as the first step. Then, the dependency graph can be represented as $G=\{V, R, E\}$, where the nodes V are tokens, the edges E denote the existence of dependency relations, and the set R contains the relation types for E. Each edge $e_{ij} \in E$ consists of a triplet (v_i, v_j, r) where $v_1, v_2 \in V$ and $r \in R$.

As shown in Figure 3, we define three kinds of relation types in R : 1) a forward syntactic relation, *e.g.*, love $\xrightarrow{\texttt{OBJ}}$ apples; 2) an inverse syntactic relation, *e.g.*, apples $\xrightarrow{\texttt{OBJ}^{-1}}$ love; and 3) a self loop

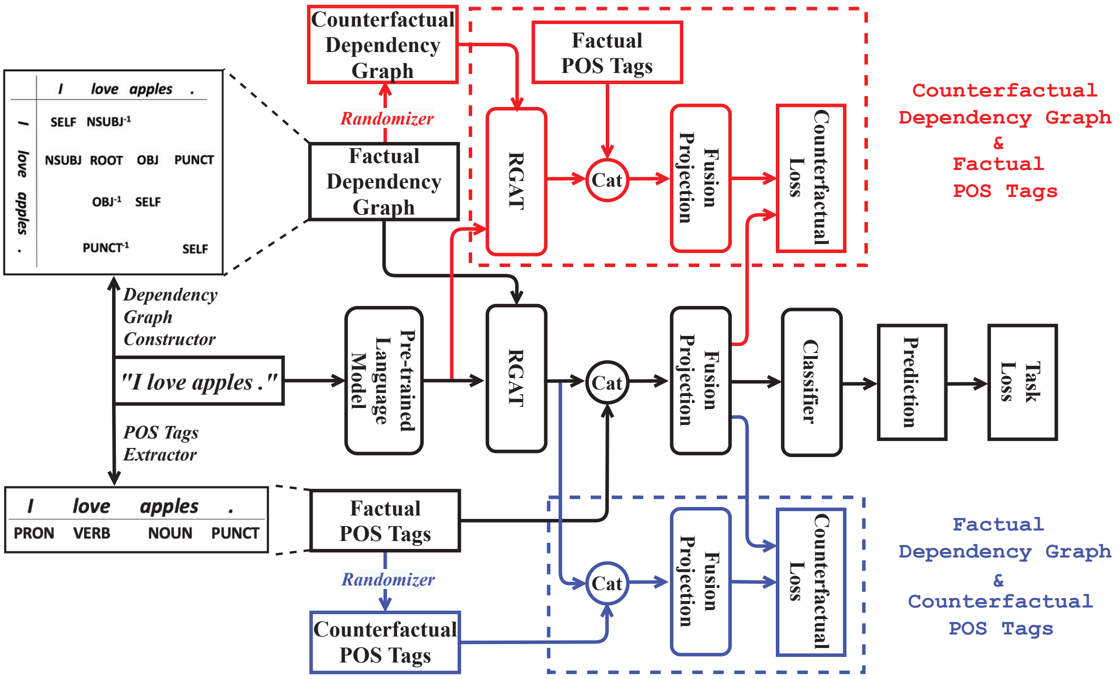

Figure 3: The overall pipeline of our COSY. We call the architecture as syntax-aware networks (Section 3.1) and the training method as counterfactual training (Section 3.2). In this architecture, there are three branches: black, red and blue. Black branch is just the normal attention-based network with additional syntactic information, and only its prediction is used in the testing stage. Red branch and blue branch are novel as they generate the counterfactual syntax samples and drive the counterfactual losses in the training stage—the key functions in COSY. RGAT stands for Relational Graph Attention Network (Ishiwatari et al., 2020; Linmei et al., 2019). **The modules of RGAT and the modules of Fusion Projection are shared across branches, *e.g.*, two RGAT modules are sharing parameters.** Cat denotes concatenation.

`SELF` that allows the information to flow from a node to itself. Note that we regard the `ROOT` relation as a self-loop. In this way, we obtain 75 different types of relations in total, and thus denote the embedding matrix as $\mathbf{R} \in \mathbb{R}^{75 \times d'}$.

POS Tags Extractor. We deploy the same Stanza toolkit (Qi et al., 2020) to assign (factual) POS tags P for all tokens. We obtain 17 different types of POS tags and denote the embedding matrix as $\mathbf{T} \in \mathbb{R}^{17 \times d'}$.

Relational Graph Attention Networks (RGAT). RGAT is one of the standard backbones to incorporate the dependency graph (Ishiwatari et al., 2020; Linmei et al., 2019). Given the (factual) dependency graph G with the contextual features of each node, RGAT can generate the relation-aware features (for each node). Details are given below. Suppose e_{ij} is the directed edge from node v_i to node v_j and the dependency relation r. The importance score of v_j from v_i is computed as:

$$s(v_i, v_j) = \mathrm{Concat}(\mathbf{e}^s_{ij}, \mathbf{e}^r_{ij}) \cdot \mathbf{W}_{Attn}, \qquad (1)$$

where $\mathbf{W}_{Attn} \in \mathbb{R}^{(d/2+d') \times 1}$ maps a vector to a scalar, $\mathbf{e}^r_{ij}$ is the embedding of the dependency relation between v_i and v_j from $\mathbf{R}$, and $\mathbf{e}^s_{ij}$ is computed by element-wise multiplication between v_i and v_j:

$$\mathbf{e}^s_{ij} = (\mathbf{h}_i \cdot \mathbf{W}_Q) \circ (\mathbf{h}_j \cdot \mathbf{W}_K), \qquad (2)$$

where $\mathbf{W}_K \in \mathbb{R}^{d \times d/2}$ and $\mathbf{W}_Q \in \mathbb{R}^{d \times d/2}$ are the learnable parameters for key and query projections (Vaswani et al., 2017), and $\mathbf{h}_i$ and $\mathbf{h}_j$ denote their contextual features extracted from pre-trained language models. Then, the importance scores are normalized across $\mathcal{N}_j$ to obtain the attention score of v_j from v_i:

$$\alpha(v_i, v_j) = \frac{\exp(s(v_i, v_j))}{\sum_{k \in \mathcal{N}_j} \exp(s(v_k, v_j))}, \qquad (3)$$

where $\mathcal{N}_j$ denotes the set of nodes pointing to v_j. The relation-aware features of v_j is computed as the weighted sum of all nodes in $\mathcal{N}_j$ with corresponding attention scores. After computing all nodes, we get the relation-aware features $\hat{\mathbf{H}} = [\hat{\mathbf{h}}_1, ..., \hat{\mathbf{h}}_S] \in \mathbb{R}^{S \times d}$.

Fusion Projection. We fuse the relation-aware features $\hat{\mathbf{H}}$ with the (factual) POS tags information before feeding them into the classifier. Given POS tags P, the fused features for each token are represented by

$$\mathbf{f}_j = \mathrm{Concat}(\hat{\mathbf{h}}_j, \mathbf{p}_j) \cdot \mathbf{W}_F, \tag{4}$$

where $\mathbf{W}_F \in \mathbb{R}^{(d+d')\times d}$ are learnable parameters of fusion projection and $\mathbf{p}_j$ is the corresponding embedding of the POS tag of the j-th token from $\mathbf{T}$. The fused features of the entire sequence are denoted as $\mathbf{F} = [\mathbf{f}_1, ..., \mathbf{f}_S] \in \mathbb{R}^{S\times d}$.

Classifier. It is designed based on the specific task, such as NLI or QA, following Devlin et al. (2019).

3.2 Counterfactual Training

Recall that the challenge in the effective utilization of syntax is how to induce the model to focus more on syntax while maintaining its original representation capability of semantics. Inspired by counterfactual analysis (Pearl et al., 2009; Pearl, 2010; Pearl and Mackenzie, 2018) and contrastive learning (Hadsell et al., 2006), we propose a counterfactual training method by incorporating counterfactual syntax (counterfactual dependency graph and counterfactual POS tags) on the red and blue branches in Figure 3. Each branch is designed to guide the model to focus on one type of syntax, *i.e.*, dependency graph or POS tags.

Counterfactual Dependency Graph is utilized on the red branch with factual POS tags in Figure 3. We build a counterfactual dependency graph by maintaining graph structure and nodes, and replacing each type of relation (except for a self-loop SELF) with a randomized (counterfactual) type. We name it G^-. We feed G^- and $\mathbf{H}$ into RGAT to obtain the counterfactual relation-aware features denoted as $\hat{\mathbf{H}}^-$. Then, we fuse $\hat{\mathbf{H}}^-$ with the factual POS tags to derive the counterfactual features $\mathbf{F}^{cf1} = [\mathbf{f}_1^{cf1}, ..., \mathbf{f}_S^{cf1}]$ on the red branch. Finally, we can calculate the similarity between the factual and the counterfactual features, by leveraging the dot-product operation, as follows,

$$\mathcal{L}_{cf1} = \frac{1}{S}\sum_{i}^{S} \mathbf{f}_i \cdot \mathbf{f}_i^{cf1}. \tag{5}$$

This counterfactual loss forces the model to emphasize the syntactic information related to dependency relations.

Counterfactual POS Tags are utilized with the factual dependency graph on the blue branch in Figure 3. We create counterfactual POS tags P^- from factual POS tags P by randomly selecting a POS tag for each token. Accordingly, we replace each embedding $\mathbf{p}_i$ by $\mathbf{p}_i^-$. Given the relation-aware features $\hat{\mathbf{H}}$ from the black branch, we then feed the embeddings of counterfactual POS tags in Eq. 4 and get the counterfactual features as $\mathbf{F}^{cf2} = [\mathbf{f}_1^{cf2}, ..., \mathbf{f}_S^{cf2}]$. Finally, we can calculate the similarity between the factual and the counterfactual features (on the blue branch) by leveraging the dot-product operation, as follows,

$$\mathcal{L}_{cf2} = \frac{1}{S}\sum_{i}^{S} \mathbf{f}_i \cdot \mathbf{f}_i^{cf2}. \tag{6}$$

This counterfactual loss forces the model to emphasize the syntactic information related to POS tags. The overall loss function used in training is as follows,

$$\mathcal{L} = \mathcal{L}_{task} + \lambda(\mathcal{L}_{cf1} + \mathcal{L}_{cf2}), \tag{7}$$

where $\mathcal{L}_{task}$ is the task-specific loss, *i.e.*, a cross-entropy loss, and λ is a scale to balance between the task-specific loss and our proposed counterfactual losses.

4 Experiments

In this section, we evaluate our COSY method for cross-lingual understanding under both zero-shot and few-shot settings. For the zero-shot setting, we use English for training and evaluate the model on different target languages. For the few-shot setting, we follow the implementation in (Nooralahzadeh et al., 2020) and use the development set of the target languages for model fine-tuning[3].

4.1 Datasets

We evaluate our method on the natural language inference (NLI) and the question answering (QA) tasks. We briefly introduce the datasets used in our experiments as follows.

Natural Language Inference (NLI). Given two sentences, NLI asks for the relationship between the two sentences, which can be entailment, contradiction or neutral. We conduct experiments on XNLI (Conneau et al., 2018) and evaluate our method on 13 target languages[4].

Question Answering (QA). In this paper, we consider the QA task that asks the model to locate the

[3] All the results and analyses are under the zero-shot settings by default, except for Table 2.

[4] We remove Thai (th) and Swahili (sw) from our experiments since these two languages are not supported by Stanza.

	Method	#T	#M	A.D.	XNLI		MLQA		XQUAD	
					en.	avg.	en.	avg.	en.	avg.
mBERT	Naive F.T.	1	1	✘	82.1	68.4	67.0 / 80.2	44.2 / 61.4	72.2 / 83.5	51.0 / 66.7
	XMAML-One	L	$O(L)$	✔	82.1	69.6	-	-	-	-
	LAKM	1	1	✔	-	-	66.8 / 80.0	-	-	-
	COSY (Ours)	1	1	✘	82.2	**70.1**	67.2 / 80.4	**45.2 / 62.1**	72.6 / 83.6	**53.2 / 68.1**
X-R$_{\text{base}}$	Naive F.T.	1	1	✘	84.6	75.1	- / 80.1	- / 65.1	71.6 / 83.1	55.9 / 71.8
	XMAML-One	L	$O(L)$	✔	-	-	- / 80.2	- / 66.1	-	-
	COSY (Ours)	1	1	✘	84.3	**75.6**	67.7 / 80.7	**48.5 / 66.5**	74.0 / 85.1	**57.3 / 73.4**
X-R$_{\text{large}}$	Naive F.T.	1	1	✘	88.7	80.0	70.6 / 83.5	53.2 / 71.6	75.7 / 86.5	60.6 / 76.8
	STILT	9	1	✔	89.6	81.6	70.8 / 84.1	54.4 / 72.8	77.4 / 88.3	63.3 / 78.7
	XMAML-One	L	$O(L)$	✔	-	-	- / 84.3	- / **73.2**	-	-
	COSY (Ours)	1	1	✘	89.2	**81.9**	70.9 / 84.2	**54.7 / 73.2**	77.7 / 88.0	**64.0 / 79.7**

Table 1: Cross-lingual **zero-shot** performance comparison between COSY and SOTA methods on three benchmark datasets. Note that we report accuracy for XNLI and Exact Match/F1 scores for MLQA and XQUAD. For each dataset, "en." denotes the results of English while "avg." is the average performance over all languages. X-R means XLM-R and Naive F.T. is the abbr. of Naive Fine-Tuning. L is the number of target languages. **#T** denotes the number of training turns, *e.g.*, STILT augments its training by using each of nine additional datasets. **#M** is the number of final models, where $1 < O(L) < L$, and **A.D.** denotes using additional datasets.

answer from a passage given a question. We conduct experiments on MLQA (Lewis et al., 2019) and XQUAD (Artetxe et al., 2020). COSY is evaluated on 7 languages on MLQA and 10 languages on XQUAD (with Thai excluded).

4.2 Implementation

In data preprocessing, we feed the same syntactic information to each of the subwords in the same word after tokenization. Our implementation of pre-trained language models (mBERT and XLM-R) is based on HuggingFaces's Transformers (Wolf et al., 2020). We select the checkpoint and set hyper-parameters, *e.g.*, learning rate and λ in the loss function, based on the performance on the corresponding development sets. We select learning rate amongst $\{7.5e{-}6, 1e{-}5, 3e{-}5\}$ and fix the batch size to 32. We select dimension d' amongst $\{100, 300\}$. λ in counterfactual loss is set to 0.1 (see Figure 4). A linear warm up strategy for learning rate is adopted with first 10% optimization steps. Adam (Kingma and Ba, 2014) is adopted as the optimizer. All experiments are conducted on a workstation with dual NVIDIA V100 32GB GPUs.

Method	en.	non-en. avg.	avg.
Naive F.T.*	81.9	70.3	71.2
XMAML-One*	82.4	70.7	71.6
COSY (Ours)	82.6	**71.9**	**72.7**

Table 2: Results of XNLI under the **few-shot** setting (mBERT). We report the testing results of English ("en."), the average results over all non-English languages ("non-en. avg.") and the average results over all languages ("avg."). * denotes the results from Nooralahzadeh et al. (2020). More details are available in Appendix.

4.3 Results

We compare our method with naive fine-tuning and the state-of-the-art methods. The overall results on three benchmarks are presented in Table 1 (zero-shot) and Table 2 (few-shot).

Comparison with Naive Fine-tuning. Naive Fine-tuning (Wu and Dredze, 2019; Liang et al., 2020; Hu et al., 2020) is to directly fine-tune the pre-trained language model on downstream tasks as in (Devlin et al., 2019). From Table 1 and Table 2, we can observe that COSY consistently outperforms the naive fine-tuning method on all datasets, *e.g.*, by average 1.9 percentage points (accuracy) and 2.9 percentage points (F1) on XNLI and XQUAD with XLM-R$_{\text{large}}$ in the zero-shot setting. These observations demonstrate the effectiveness of COSY and suggest that universal syntax as language-agnostic features can enhance the transferability for cross-lingual understanding. Furthermore, the results show that COSY is able to work with different backbones and thus is model-agnostic.

Comparison with the State of the Art. We first outline the SOTA zero-shot (few-shot) cross-lingual methods we compared with as follows: (1) XMAML-one (Nooralahzadeh et al., 2020) borrows the idea from meta-learning. Specifically, XMAML-one utilizes an auxiliary language development data in training, *e.g.*, using the development set of Spanish in training to assist German on MLQA. XMAML-One reports the results based on the most beneficial auxiliary language. (2) STILT (Phang et al., 2020) augments intermediate task training before fine-tuning on the target task, *e.g.*, adding training of HellaSwag (Zellers et al., 2019) before training on the NLI task. STILT also reports results with the most beneficial intermediate task. (3) LAKM (Yuan et al., 2020) first mines knowledge phrases along with passages from the Web. Then these Web data are used to enhance the phrase boundaries through a masked language model objective. Note that LAKM is only evaluated on three languages of MLQA.

On the one hand, we observe that COSY surpasses the compared SOTA methods over all evaluation metrics. Although meta-learning methods (Finn et al., 2017; Gu et al., 2018; Sun et al., 2019) advance the state-of-the-art performance for few-shot learning, our COSY still outperforms the meta-learning-based method, *i.e.*, XMAML-One, with 1.1 percentage points in the few-shot setting. On the other hand, the superiority of COSY is also reflected in other aspects, which are shown in Table 1. Specifically, COSY does not require additional datasets and cumbersome data selection process, which is more convenient and resources saving.

4.4 Discussion and Analysis

Ablation Study. In Table 3, we show the MLQA, XQUAD and XNLI results in 4 ablative settings, to evaluate the approach when we (1) only utilize the SAN-Black branch; (2) utilize the SAN-Black branch with an intuitive gate mechanism to control the information of pre-trained language model and syntax; (3) utilize the SAN-Black branch and SAN-Red branch; (4) utilize the SAN-Black branch and SAN-Blue branch.

Ablative Setting	**MLQA**		**XQUAD**		**XNLI**
	EM	F1	EM	F1	Acc
Naive F.T.	44.2	61.4	51.0	66.7	68.4
(1) SAN-Black	44.3	61.4	51.6	66.9	68.7
(2) SAN-Black+Gate	44.5	61.5	51.9	67.1	68.7
(3) SAN-Black, Red	44.9	61.7	52.8	67.8	69.9
(4) SAN-Black, Blue	44.7	61.8	52.2	67.4	69.7
(5) COSY	**45.2**	**62.1**	**53.2**	**68.1**	**70.1**

Table 3: The ablation study on MLQA, XQUAD and XNLI (mBERT). We report the average performance of all languages on the test set.

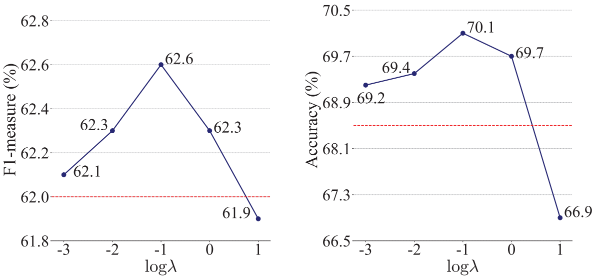

Figure 4: Left: average F1-measure (%) on target languages on MLQA development set (mBERT). Right: average accuracy (%) on target languages on XNLI development set (mBERT). Red dotted line denotes the model performance of using naive fine-tuning.

Compared to the ablative results, we can see that our full method achieves the overall top performance in all settings. Syntax features are incorporated into the models in (1)-(5) and all of them outperform the naive fine-tuning method, which demonstrates the effectiveness of universal syntax. By analyzing the settings one by one, we can observe that SAN-Black only attains limited improvement compared to naive fine-tuning since syntax is incorporated in the model by overlooked. Gate mechanism (2) fails to solve the overlooking issue. Both of (3) and (4) with counterfactual training are able to bring gains compared to (1), and the results indicate that dependency relations are more effective compared to POS labels. We also observe that our full method (5) does not accumulate the gains from (3) and (4). One explanation could be that part of the information provided by the dependency relations and POS labels overlaps. For instance, if we see an edge of relation, $\text{word}_a \xrightarrow{\texttt{AMOD}} \text{word}_b$, we may infer that word_a is `NOUN` and word_b is `ADJ`.

Effect of λ. We now study the impact of the scale value λ with counterfactual losses. For clarity, we show the results with different values of $\log\lambda$ in Figure 4. We can observe that COSY attains the

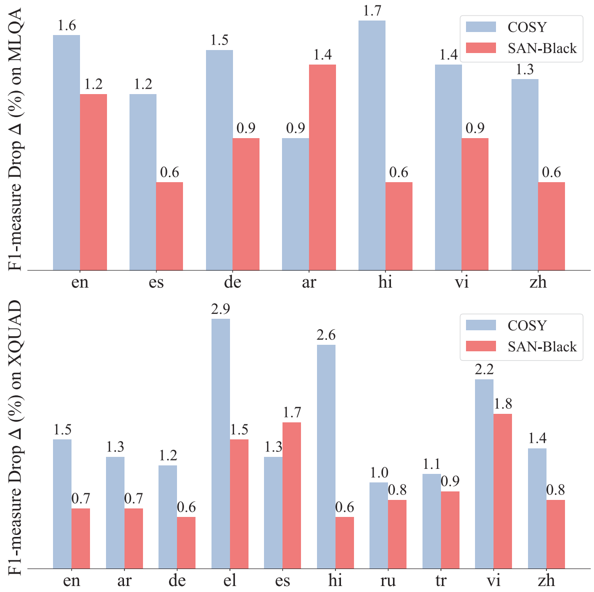

Figure 5: F1-measure drop Δ (%) with a standard normal distribution perturbation on MLQA and XQUAD (mBERT). Two colors denote COSY and SAN-Black.

highest results when $\lambda = 0.1$ on both MLQA and XNLI. As the value drops, the effect of counterfactual loss is also smaller and the performance is getting closer to that from naive fine-tuning (red dotted line). If a large value of λ is applied, *e.g.*, $\lambda = 1$, the model begins to over-emphasize the syntax and semantics are overlooked, which leads to significant decrease on performance.

Effect of COSY. In this part, we first study whether counterfactual training method indeed guides the model to focus more on syntactic information. We conduct analysis on the COSY and SAN-Black. Since it is non-trivial to measure the utilization of syntax in a straightforward way, we adopt a standard way to measure the importance of the neurons in deep models (Kádár et al., 2017). Specifically, we perturb the syntactic features with a Gaussian noise to test data and check whether our model would be more easily affected by the syntax perturbation. If so, then it verifies that our model indeed relies more on syntax.. The results are shown in Figure 5. We can discover that the performance drop of COSY is larger compared to that with SAN-Black.

Meanwhile, we also explore whether COSY is beneficial for yielding more meaningful syntax embedding than SAN-Black. Specifically, we compute the correlation score (absolute cosine similarity) between the embedding of syntactic relation and the corresponding inverse relation from the same type. For COSY, we observe that the score of the related types are $42.4\times$ larger than that of two randomly selected embeddings (average over 10000 times). However, for SAN-Black, its score is only $1.4\times$ larger than that of two randomly selected embeddings. It demonstrates that COSY attains more meaningful syntax representations than SAN-Black.

	MLQA		XQUAD	
	EM	F1	EM	F1
(1)	44.8	61.7	52.2	67.3
(2)	45.1	62.0	53.1	68.1
(3)	44.9	61.9	52.7	67.8
(4)	45.0	62.0	53.2	68.0
Current	45.2	62.1	53.2	68.1

Table 4: Results of different generation ways for generating counterfactual syntax with mBERT as backbone. "Current" means the current generation way described in Section 3. We report the average performance of all languages.

Counterfactual Syntax Generation. Here we analyze other alternative ways of counterfactual syntax generation. Specifically, we design the following variants and report the results in Table 4: (1) we not only replace edge types, but also replace connections for counterfactual dependency graph construction; (2) for each input sequence, we create 5 counterfactual dependency graphs, 5 sets of counterfactual POS tags, and the counterfactual loss is the average over the 5 sets; (3) we replace the factual syntax with a fixed type, *e.g.*, a type of padding instead of a random type from all types; (4) in each generating process, we only replace 50% of the factual syntax.

Comparing (1) with the result of "SAN-Black,Blue" in Table 3, we can see that (1) does not work. We believe that randomly changing connections in G^{-}, *e.g.*, an edge is created from the first token to the last token in a long passage, may have a significant effect to $\hat{\mathbf{H}}^{-}$, it is undesirable for further optimization of counterfactual loss. Results from (2) and (4) suggest that the number of the generated counterfactual syntax and ratio of randomizing do not play an important role in COSY. It is also discovered that randomizing with all types is better than simple replacement with a fixed type.

5 Conclusion

We study how to effectively plug in syntactic information for cross-lingual understanding. Specifically, we propose a novel counterfactual-syntax-based approach to emphasize the importance of syntax in cross-lingual models. We conduct extensive experiments on three cross-lingual benchmarks, and show that our approach can outperform the SOTA methods without additional dataset. For future work, we will combine our approach with other orthogonal methods, *e.g.*, meta-learning, to further improve its effectiveness.

Acknowledgments

This research is supported by the National Research Foundation, Singapore under its Strategic Capabilities Research Centres Funding Initiative, and partially supported by the Agency for Science, Technology and Research (A*STAR) under its AME YIRG Grant (Project No. A20E6c0101), and its AME Programmatic Fund (Project No: A18A1b0045 and No: A18A2b0046). Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore.

References

Ehsan Abbasnejad, Damien Teney, Amin Parvaneh, Javen Shi, and Anton van den Hengel. 2020. Counterfactual vision and language learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2020. On the cross-lingual transferability of monolingual representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Long Chen, Xin Yan, Jun Xiao, Hanwang Zhang, Shiliang Pu, and Yueting Zhuang. 2020. Counterfactual samples synthesizing for robust visual question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Rochelle Choenni and Ekaterina Shutova. 2020. What does it mean to be language-agnostic? probing multilingual sentence encoders for typological properties. *arXiv preprint arXiv:2009.12862*.

Seungtaek Choi, Haeju Park, Jinyoung Yeo, and Seungwon Hwang. 2020. Less is more: Attention supervision with counterfactuals for text classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Édouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020a. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Alexis Conneau, Shijie Wu, Haoran Li, Luke Zettlemoyer, and Veselin Stoyanov. 2020b. Emerging cross-lingual structure in pretrained language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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*.

Philipp Dufter and Hinrich Schütze. 2020. Identifying necessary elements for bert's multilinguality. *arXiv preprint arXiv:2005.00396*.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*.

Yash Goyal, Amir Feder, Uri Shalit, and Been Kim. 2019a. Explaining classifiers with causal concept effect (cace). *arXiv preprint arXiv:1907.07165*.

Yash Goyal, Ziyan Wu, Jan Ernst, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019b. Counterfactual visual explanations. In *International Conference on Machine Learning*.

Jiatao Gu, Yong Wang, Yun Chen, Victor OK Li, and Kyunghyun Cho. 2018. Meta-learning for low-resource neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Raia Hadsell, Sumit Chopra, and Yann LeCun. 2006. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Tsung-Yuan Hsu, Chi-Liang Liu, and Hung-yi Lee. 2019. Zero-shot reading comprehension by cross-lingual transfer learning with multi-lingual language representation model. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. Xtreme: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *International Conference on Machine Learning*.

Po-Sen Huang, Huan Zhang, Ray Jiang, Robert Stanforth, Johannes Welbl, Jack Rae, Vishal Maini, Dani Yogatama, and Pushmeet Kohli. 2020. Reducing sentiment bias in language models via counterfactual evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*.

Taichi Ishiwatari, Yuki Yasuda, Taro Miyazaki, and Jun Goto. 2020. Relation-aware graph attention networks with relational position encodings for emotion recognition in conversations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*.

Ákos Kádár, Grzegorz Chrupała, and Afra Alishahi. 2017. Representation of linguistic form and function in recurrent neural networks. *Computational Linguistics*.

Phillip Keung, Vikas Bhardwaj, et al. 2019. Adversarial learning with contextual embeddings for zero-resource cross-lingual classification and ner. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*.

Phillip Keung, Yichao Lu, Julian Salazar, and Vikas Bhardwaj. 2020. On the evaluation of contextual embeddings for zero-shot cross-lingual transfer learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*.

Anne Lauscher, Vinit Ravishankar, Ivan Vulić, and Goran Glavaš. 2020. From zero to hero: On the limitations of zero-shot cross-lingual transfer with multilingual transformers. *arXiv preprint arXiv:2005.00633*.

Patrick Lewis, Barlas Oğuz, Ruty Rinott, Sebastian Riedel, and Holger Schwenk. 2019. Mlqa: Evaluating cross-lingual extractive question answering. *arXiv preprint arXiv:1910.07475*.

Yaobo Liang, Nan Duan, Yeyun Gong, Ning Wu, Fenfei Guo, Weizhen Qi, Ming Gong, Linjun Shou, Daxin Jiang, Guihong Cao, et al. 2020. Xglue: A new benchmark datasetfor cross-lingual pre-training, understanding and generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*.

Hu Linmei, Tianchi Yang, Chuan Shi, Houye Ji, and Xiaoli Li. 2019. Heterogeneous graph attention networks for semi-supervised short text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Zihan Liu, Genta Indra Winata, Andrea Madotto, and Pascale Fung. 2020. Exploring fine-tuning techniques for pre-trained cross-lingual models via continual learning. *arXiv preprint arXiv:2004.14218*.

Marie-Catherine de Marneffe, Timothy Dozat, Natalia Silveira, Katri Haverinen, Filip Ginter, Joakim Nivre, and Christopher D Manning. 2014. Universal stanford dependencies: A cross-linguistic typology. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation*.

Yulei Niu, Kaihua Tang, Hanwang Zhang, Zhiwu Lu, Xian-Sheng Hua, and Ji-Rong Wen. 2020. Counterfactual vqa: A cause-effect look at language bias. *arXiv preprint arXiv:2006.04315*.

Joakim Nivre, Marie-Catherine De Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, et al. 2016. Universal dependencies v1: A multilingual treebank collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation*.

Farhad Nooralahzadeh, Giannis Bekoulis, Johannes Bjerva, and Isabelle Augenstein. 2020. Zero-shot cross-lingual transfer with meta learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*.

Judea Pearl. 2010. Causal inference. *Causality: Objectives and Assessment*.

Judea Pearl and Dana Mackenzie. 2018. The book of why: The new science of cause and effect.

Judea Pearl et al. 2009. Causal inference in statistics: An overview. *Statistics surveys*.

Jason Phang, Phu Mon Htut, Yada Pruksachatkun, Haokun Liu, Clara Vania, Katharina Kann, Iacer

Calixto, and Samuel R Bowman. 2020. English intermediate-task training improves zero-shot cross-lingual transfer too. *arXiv preprint arXiv:2005.13013*.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual bert? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

Lianhui Qin, Antoine Bosselut, Ari Holtzman, Chandra Bhagavatula, Elizabeth Clark, and Yejin Choi. 2019. Counterfactual story reasoning and generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*.

Neal J Roese. 1997. Counterfactual thinking. *Psychological bulletin*.

Kenneth J Rothman and Sander Greenland. 2005. Causation and causal inference in epidemiology. *American journal of public health*.

Seungjae Shin, Kyungwoo Song, JoonHo Jang, Hyemi Kim, Weonyoung Joo, and Il-Chul Moon. 2020. Neutralizing gender bias in word embedding with latent disentanglement and counterfactual generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*.

Daniel Steel. 2004. Social mechanisms and causal inference. *Philosophy of the social sciences*.

Qianru Sun, Yaoyao Liu, Tat-Seng Chua, and Bernt Schiele. 2019. Meta-transfer learning for few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Kaihua Tang, Jianqiang Huang, and Hanwang Zhang. 2020a. Long-tailed classification by keeping the good and removing the bad momentum causal effect. *arXiv preprint arXiv:2009.12991*.

Kaihua Tang, Yulei Niu, Jianqiang Huang, Jiaxin Shi, and Hanwang Zhang. 2020b. Unbiased scene graph generation from biased training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*.

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. 2020. Investigating gender bias in language models using causal mediation analysis. *Advances in Neural Information Processing Systems*.

Xiangpeng Wei, Yue Hu, Rongxiang Weng, Luxi Xing, Heng Yu, and Weihua Luo. 2020. On learning universal representations across languages. *arXiv preprint arXiv:2007.15960*.

Thomas Wolf, Julien Chaumond, Lysandre Debut, Victor Sanh, Clement Delangue, Anthony Moi, Pierric Cistac, Morgan Funtowicz, Joe Davison, Sam Shleifer, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*.

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

Ming Yan, Hao Zhang, Di Jin, and Joey Tianyi Zhou. 2020. Multi-source meta transfer for low resource multiple-choice question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Sicheng Yu, Yulei Niu, Shuohang Wang, Jing Jiang, and Qianru Sun. 2020. Counterfactual variable control for robust and interpretable question answering. *arXiv preprint arXiv:2010.05581*.

Fei Yuan, Linjun Shou, Xuanyu Bai, Ming Gong, Yaobo Liang, Nan Duan, Yan Fu, and Daxin Jiang. 2020. Enhancing answer boundary detection for multilingual machine reading comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*.

Wei Zhao, Steffen Eger, Johannes Bjerva, and Isabelle Augenstein. 2020. Inducing language-agnostic multilingual representations. *arXiv preprint arXiv:2008.09112*.

Joey Tianyi Zhou, Hao Zhang, Di Jin, and Xi Peng. 2021. Dual adversarial transfer for sequence labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Joey Tianyi Zhou, Hao Zhang, Di Jin, Hongyuan Zhu, Meng Fang, Rick Siow Mong Goh, and Kenneth Kwok. 2019. Dual adversarial neural transfer for low-resource named entity recognition. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Qingfu Zhu, Weinan Zhang, Ting Liu, and William Yang Wang. 2020. Counterfactual off-policy training for neural dialogue generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*.

Appendix

Methods	en	fr	es	de	el	bg	ru	tr	ar	vi	zh	hi	ur	Avg
mBERT														
Naive Fine-tuning[1]	82.1	73.8	74.3	71.1	66.4	68.9	69.0	61.6	64.9	69.5	69.3	60.0	58.0	68.4
XMAML-One[2]	82.1	74.4	75.1	71.8	68.0	69.5	70.2	61.2	66.1	71.8	71.1	62.2	61.5	69.6
COSY	82.2	75.2	75.5	72.2	68.9	71.1	70.1	63.1	66.7	72.4	71.3	62.4	59.7	**70.1**
XLM-R$_{base}$														
Naive Fine-tuning[3]	84.6	78.2	79.2	77.0	75.9	77.5	75.5	72.9	72.1	74.8	73.7	69.8	65.1	75.1
COSY	84.3	78.8	78.6	76.4	76.3	78.4	76.3	73.9	71.1	75.4	75.1	71.1	67.1	**75.6**
XLM-R$_{large}$														
Naive Fine-tuning[4]	88.7	82.2	83.7	82.5	80.8	83.0	79.1	78.0	77.2	79.3	78.2	75.6	71.7	80.0
STILT[5]	89.6	84.1	84.5	83.7	81.8	83.5	79.9	80.1	79.3	81.3	80.7	78.2	74.5	81.6
COSY	89.2	83.6	85.1	83.2	83.3	84.7	80.9	80.8	80.1	81.0	80.5	77.7	74.1	**81.9**

Table 5: Results on XNLI of zero-shot setting. We report the accuracy on 13 XNLI languages and the average accuracy. 1: (Wu and Dredze, 2019); 2: (Nooralahzadeh et al., 2020); 3: (Liang et al., 2020); 4: (Hu et al., 2020); 5: (Phang et al., 2020).

Methods	en	es	de	ar	hi	vi	zh	Avg
mBERT								
Naive Fine-tuning[1]	67.0 / 80.2	49.2 / 67.4	43.8 / 59.0	34.6 / 52.3	35.3 / 50.2	40.7 / 61.2	38.6 / 59.6	44.2 / 61.4
LAKM[3]	66.8 / 80.0	48.0 / 65.9	44.5 / 60.5	-	-	-	-	-
COSY	67.2 / 80.4	48.5 / 66.4	47.0 / 61.1	35.0 / 52.9	35.9 / 51.2	43.2 / 63.1	39.3 / 59.8	**45.2 / 62.1**
XLM-R$_{base}$								
Naive Fine-tuning[2]	- / 80.1	- / 67.9	- / 62.1	- / 56.4	- / 60.5	- / 67.1	- / 61.4	- / 65.1
Naive Fine-tuning*	67.1 / 80.1	50.3 / 68.0	48.3 / 62.9	37.2 / 57.0	44.5 / 62.4	47.1 / 67.4	38.4 / 62.0	47.6 / 65.7
XMAML-One[4]	- / 80.2	- / 67.5	- / 63.6	- / 58.0	- / 61.7	- / 68.0	- / 64.0	- / 66.1
COSY	67.7 / 80.7	50.9 / 68.7	49.1 / 63.4	38.7 / 57.8	45.4 / 62.7	47.9 / 68.3	39.7 / 63.6	**48.5 / 66.5**
XLM-R$_{large}$								
Naive Fine-tuning[1]	70.6 / 83.5	56.6 / 74.1	54.9 / 70.1	47.1 / 66.6	53.1 / 70.6	52.9/ 74.0	37.0 / 62.1	53.2 / 71.6
STILT[5]	70.8 / 84.1	56.8 / 75.3	52.9 / 69.6	46.4 / 67.4	54.8 / 72.5	51.7 / 70.9	47.0 / 69.4	54.4 / 72.8
XMAML-One[4]	- / 84.3	- / 74.3	- / 70.8	- / 66.6	- / 70.9	- / 74.8	- / 70.7	- / **73.2**
COSY	70.9 / 84.2	56.5 / 74.7	55.2 / 70.3	46.7 / 66.7	53.7 / 72.1	53.2 / 74.3	46.6 / 70.2	**54.7 / 73.2**

Table 6: Results on MLQA of zero-shot setting. We report the Exact Match and F1 score (EM / F1) on 7 languages. ∗: our implementation by official code; 1: (Hu et al., 2020); 2: (Liang et al., 2020); 3: (Yuan et al., 2020); 4: (Nooralahzadeh et al., 2020); 5: (Phang et al., 2020).

Methods	en	ar	de	el	es	hi	ru	tr	vi	zh	Avg
mBERT											
Naive Fine-tuning[1]	72.2 / 83.5	45.1 / 61.5	54.0 / 70.6	44.9 / 62.6	56.9 / 75.5	46.0 / 59.2	53.3 / 71.3	40.1 / 55.4	49.6 / 69.5	48.3 / 58.0	51.0 / 66.7
COSY	72.6 / 83.6	47.6 / 63.6	57.2 / 72.3	47.7 / 64.6	58.6 / 76.5	47.5 / 60.7	55.6 / 72.1	42.2 / 56.7	54.0 / 72.4	48.9 / 58.5	**53.2 / 68.1**
XLM-R$_{base}$											
Naive Fine-tuning*	71.6 / 83.1	49.9 / 66.2	56.6 / 72.5	54.2 / 72.4	58.8 / 76.6	51.3 / 67.7	57.2 / 74.1	52.5 / 68.3	53.8 / 73.6	52.6 / 63.6	55.9 / 71.8
COSY	74.0 / 85.1	51.0 / 67.8	59.2 / 75.4	55.5 / 73.2	59.0 / 77.2	51.5 / 69.1	58.5 / 75.0	52.5 / 69.5	56.0 / 74.2	56.2 / 67.3	**57.3 / 73.4**
XLM-R$_{large}$											
Naive Fine-tuning[1]	75.7 / 86.5	49.0 / 68.6	63.4 / 80.4	61.7 / 79.8	63.9 / 82.0	59.7 / 76.7	64.3 / 80.1	59.3 / 75.9	59.0 / 79.1	50.0 / 59.3	60.6 / 76.8
STILT[2]	77.4 / 88.3	59.9 / 75.9	63.6 / 80.3	62.1 / 80.3	63.2 / 81.8	59.2 / 76.1	64.1 / 80.0	59.2 / 75.8	61.2 / 80.5	61.3 / 70.8	63.3 / 78.7
COSY	77.7 / 88.0	58.7 / 76.5	65.1 / 81.4	64.4 / 81.7	64.0 / 82.5	60.6 / 77.1	64.7 / 80.9	60.7 / 76.3	61.5 / 80.7	63.0 / 72.1	**64.0 / 79.7**

Table 7: Results on XQUAD of zero-shot setting. We report the Exact Match and F1 score (EM / F1) on 10 languages. ∗: our implementation by official code; 1: (Hu et al., 2020); 2: (Phang et al., 2020).

Methods	**en**	**fr**	**es**	**de**	**el**	**bg**	**ru**	**tr**	**ar**	**vi**	**zh**	**hi**	**ur**	**Avg**
Naive Fine-tuning	81.9	75.4	75.8	73.3	69.5	71.6	70.8	64.9	67.4	73.2	73.9	64.4	63.7	71.2
XMAML-One	82.4	75.3	76.2	73.5	70.0	71.9	71.5	64.9	68.0	73.5	74.2	65.0	63.8	71.6
XMAML-Two	82.7	76.0	76.5	74.1	70.7	72.8	72.1	65.7	68.4	73.9	74.9	65.8	64.6	72.1
COSY	82.7	77.2	76.5	74.3	71.1	73.9	72.4	67.6	69.8	74.3	74.7	66.4	63.7	**72.7**

Table 8: Results on XNLI of few-shot setting with mBERT. We report the accuracy on 13 XNLI languages and the average accuracy. Results except our COSY are all from (Nooralahzadeh et al., 2020).