Constructing Code-mixed Universal Dependency Forest for Unbiased Cross-lingual Relation Extraction

Hao Fei[1], **Meishan Zhang**[2],[*] **Min Zhang**[2], **Tat-Seng Chua**[1]
[1] Sea-NExT Joint Lab, School of Computing, National University of Singapore
[2] Harbin Institute of Technology (Shenzhen), China
{haofei37,dcscts}@nus.edu.sg, mason.zms@gmail.com, zhangmin2021@hit.edu.cn

Abstract

Latest efforts on cross-lingual relation extraction (XRE) aggressively leverage the language-consistent structural features from the universal dependency (UD) resource, while they may largely suffer from biased transfer (e.g., either target-biased or source-biased) due to the inevitable linguistic disparity between languages. In this work, we investigate an unbiased UD-based XRE transfer by constructing a type of code-mixed UD forest. We first translate the sentence of the source language to the parallel target-side language, for both of which we parse the UD tree respectively. Then, we merge the source-/target-side UD structures as a unified code-mixed UD forest. With such forest features, the gaps of UD-based XRE between the training and predicting phases can be effectively closed. We conduct experiments on the ACE XRE benchmark datasets, where the results demonstrate that the proposed code-mixed UD forests help unbiased UD-based XRE transfer, with which we achieve significant XRE performance gains.

1 Introduction

Relation extraction (RE) aims at extracting from the plain texts the meaningful *entity mentions* paired with *semantic relations*. One widely-acknowledged key bottleneck of RE is called the long-range dependence (LRD) issue, i.e., the decay of dependence clues of two mention entities with increasing distance in between (Culotta and Sorensen, 2004; Zhang et al., 2018; Fei et al., 2021). Fortunately, prior work extensively reveals that the syntactic dependency trees help resolve LRD issue effectively, by taking advantage of the close relevance between the dependency structure and the relational RE pair (Miwa and Bansal, 2016; Can et al., 2019). In cross-lingual RE, likewise, the universal dependency trees (de Marneffe et al., 2021) are leveraged as effective language-persistent features

[*] Corresponding author

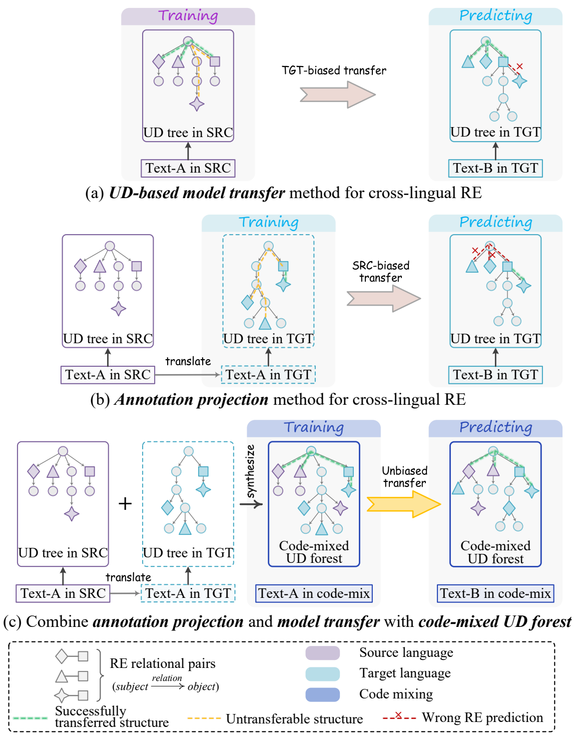


Figure 1: Model transfer fails to model the TGT-side language-specific features due to the syntactic structure discrepancy (a), while annotation projection may overlook the SRC-side effective UD features (b). This work combines the two methods and constructs code-mixed UD forests for unbiased cross-lingual RE (c).

in the latest work for better transfer from source (SRC) language to target (TGT) language (Subburathinam et al., 2019; Fei et al., 2020b; Taghizadeh and Faili, 2021).

Current state-of-the-art (SoTA) XRE work leverages the UD trees based on the model transfer paradigm, i.e., training with SRC-side UD features while predicting with TGT-side UD features (Ahmad et al., 2021; Taghizadeh and Faili, 2022). Model transfer method transfers the shareable parts of features from SRC to TGT, while unfortunately it could fail to model the TGT-side language-specific features, and thus results in a clear *TGT-side bias*. In fact, the TGT-side bias can

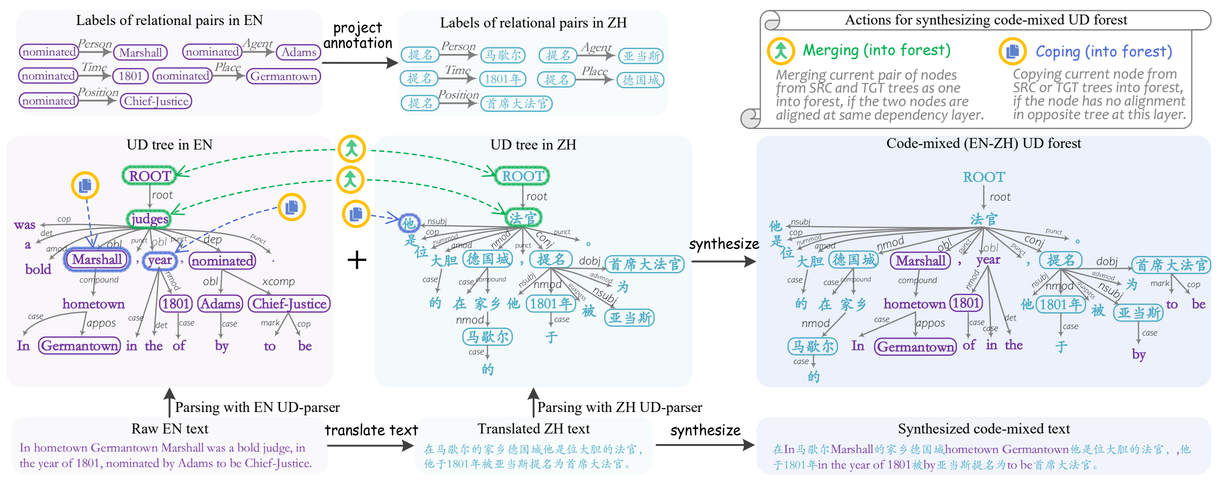

Labels of relational pairs in EN
nominated Person Marshall
nominated Agent Adams
nominated Time 1801
nominated Place Germantown
nominated Position Chief-Justice
project annotation
Labels of relational pairs in ZH
提名 Person 马歇尔
提名 Agent 亚当斯
提名 Time 1801年
提名 Place 德国城
提名 Position 首席大法官
Actions for synthesizing code-mixed UD forest
Merging (into forest)
Merging current pair of nodes from SRC and TGT trees as one into forest, if the two nodes are aligned at same dependency layer.
Coping (into forest)
Copying current node from SRC or TGT trees into forest, if the node has no alignment in opposite tree at this layer.
UD tree in EN
UD tree in ZH
Code-mixed (EN-ZH) UD forest
ROOT
root
judges
法官
synthesize
Parsing with EN UD-parser
Parsing with ZH UD-parser
Raw EN text
In hometown Germantown Marshall was a bold judge, in the year of 1801, nominated by Adams to be Chief-Justice.
translate text
Translated ZH text
在马歇尔的家乡德国城他是位大胆的法官，他于1801年被亚当斯提名为首席大法官。
synthesize
Synthesized code-mixed text
在In马歇尔Marshall的家乡德国城hometown Germantown他是位大胆的法官，，他于1801年in the year of 1801被by亚当斯提名为to be首席大法官。

Figure 2: A real example to construct a code-mixed UD forest. The raw sentence is selected from ACE05 data. We exemplify the transfer from English (EN) to Chinese (ZH).

be exacerbated in UD-based model transfer, cf. Fig. 1(a). Given that UD has a universal annotation standard, inevitably, there is still a syntax discrepancy between the two languages due to their intrinsic linguistic nature. We show (cf. §3 for more discussion) that between the parallel sentences in English and Arabic, around 30% words are misaligned and over 35% UD word-pairs have no correspondence. Such structural discrepancies consequently undermine the model transfer efficacy.

One alternative solution is using annotation projection (Padó and Lapata, 2009; Kim et al., 2010; McDonald et al., 2013; Xiao and Guo, 2015). The main idea is directly synthesizing the pseudo TGT-side training data, so that the TGT-side linguistic features (i.e., UD trees) are well preserved. However, it could be a double side of the sword in the annotation projection paradigm. It manages to learn the language-specific features, while at the cost of losing some high-efficient structural knowledge from SRC-side UD, thus leading to the SRC-biased UD feature transfer. As illustrated in Fig. 1(b), the dependence paths in the SRC UD tree that effectively solves the LRD issues for the task are sacrificed when transforming the SRC tree into the TGT tree.

This motivates us to pursue an unbiased and holistic UD-based XRE transfer by considering both the SRC and TGT UD syntax features. To reach the goal, in this work, we propose combining the view of model transfer and annotation projection paradigm, and constructing a type of code-mixed UD forests. Technically, we first project the SRC training instances and TGT predicting instances into the opposite languages, respectively. Then, we parse the parallel UD trees of both sides respectively via existing UD parsers. Next, merge each pair of SRC and TGT UD trees together into the code-mixed UD forest, in which the well-aligned word pairs are merged to the TGT ones in the forest, and the unaligned words will all be kept in the forest. With these code-mixed syntactic features, the gap between training and predicting phases can be closed, as depicted in Fig. 1(c).

We encode the UD forest with the graph attention model (GAT; Velickovic et al., 2018) for feature encoding. We perform experiments on the representative XRE benchmark, ACE05 (hristopher Walker et al., 2006), where the transfer results from English to Chinese and Arabic show that the proposed code-mixed forests bring significant improvement over the current best-performing UD-based system, obtaining the new SoTA results. Further analyses verify that 1) the code-mixed UD forests help maintain the debiased cross-lingual transfer of RE task, and 2) the larger the difference between SRC and TGT languages, the bigger the boosts offered by code-mixed forests. To our knowledge, we are the first taking the complementary advantages of annotation projection and model transfer paradigm for unbiased XRE transfer. We verify that the gap between training and predicting of UD-based XRE can be bridged by synthesizing a type of code-mixed UD forests. The resource can be found at `https://github.com/scofield7419/XLSIE/`.

2 Related Work

Different from the sequential type of information extraction (IE), e.g., named entity recognition

(NER) (Cucerzan and Yarowsky, 1999), RE not only detects the mentions but also recognizes the semantic relations between mentions. RE has long received extensive research attention within the last decades (Zelenko et al., 2002). Within the community, research has revealed that the syntactic dependency trees share close correlations with RE or broad-covering information extraction tasks in structure (Fei et al., 2021; Wu et al., 2021; Fei et al., 2022), and thus the former is frequently leveraged as supporting features for enhancing RE. In XRE, the key relational features between words need to be transferred between languages, which motivates the incorporation of UD tree features that have consistent annotations and principles across various languages. Thus, UD-based systems extensively achieve the current SoTA XRE (Lu et al., 2020; Taghizadeh and Faili, 2021; Zhang et al., 2021). This work inherits the prior wisdom, and leverages the UD features.

Model transfer (Kozhevnikov and Titov, 2013; Ni and Florian, 2019; Fei et al., 2020b) and annotation projection (Björkelund et al., 2009; Mulcaire et al., 2018; Daza and Frank, 2019; Fei et al., 2020a; Lou et al., 2022) are two mainstream avenues in structural cross-lingual transfer track. The former trains a model on SRC annotations and them make predictions with TGT instances, i.e., transferring the shared language-invariant features. The latter directly synthesizes the pseudo training instances in TGT language based on some parallel sentences, in which the TGT-specific features are retained to the largest extent. As we indicated earlier, in both two paradigms the UD tree features can be unfortunately biased during the transfer, thus leading to the underutilization of UD resource. This work considers a holistic viewpoint, integrating both the two cross-lingual transfer schemes and combining both the SRC and TGT syntax trees by code mixing.

Several prior studies have shown that combining the raw SRC and pseudo TGT (from projection) data for training helps better transfer. It is shown that although the two data are semantically identical, SRC data still can offer some complementary language-biased features (Fei et al., 2020a,b; Zhen et al., 2021). Yet we emphasize that different from regular cross-lingual text classification or sequential prediction, XRE relies particularly on the syntactic structure features, e.g., UD, and thus needs a more fine-grained approach for SRC-TGT data ensembling, instead of simply instance stacking. Thus, we propose merging the SRC and TGT syntax trees into the code-mixed forests.

Code mixing has been explored in several different NLP applications (Labutov and Lipson, 2014; Joshi et al., 2016; Banerjee et al., 2018; Samanta et al., 2019), where the core idea is creating data piece containing words from different languages simultaneously. For example, Samanta et al. (2019) introduce a novel data augmentation method for enhancing the recognition of code-switched sentiment analysis, where they replace the constituent phrases with code-mixed alternatives. Qin et al. (2020) propose generating code-switching data to augment the existing multilingual language models for better zero-shot cross-lingual tasks. While we notice that most of the works focus on the development of code-mixed sequential texts, this work considers the one for structural syntax trees. Our work is partially similar to Zhang et al. (2019) on the code-mixed UD tree construction. But ours differentiate theirs in that Zhang et al. (2019) target better UD parsing itself, while we aim to improve downstream tasks.

3 Observations on UD Bias

3.1 Bias Source Analysis

As mentioned, even though UD trees define consistent annotations across languages, it still falls short on wiping all syntactic bias. This is inevitably caused by the underlying linguistic disparity deeply embedded in the language itself. Observing the linguistic discrepancies between different languages, we can summarize them into following three levels:

1) Word-level Changes.

- **Word number.** The words referring to same semantics in different languages vary, e.g., in English one single-token word may be translated in Chinese with more than one token.
- **Part of speech.** In different languages a parallel lexicon may come with different part of speech.
- **Word order.** Also it is a common case that the word order varies among parallel sentences in different languages.

2) Phrase-level Change.

- **Modification type.** A modifier of a phrasal constituent can be changed when translating into another languages. For example, in English, 'in the distance' is often an adverbial modifier, while its counterpart in Chinese ‘遥远的’ plays a role of an attribute modifier.

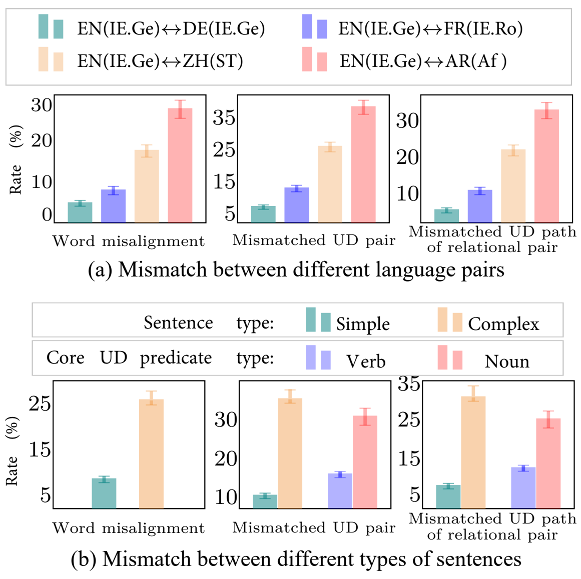
- **Change of pronouns.** English grammar has strict structure, while in some other languages the grammar structures may not strict. For example, in English, it is often case to use relative pronouns (e.g., which, that, who) to refer to the prior mentions, while in other languages, such as Chinese, the personal pronouns (e.g., which, that, who) will be used to refer the prior mentions.
- **Constituency order change.** Some constituent phrases will be reorganized and reordered from one language to another language, due to the differences in grammar rules.

3) Sentence-level Change.

- **Transformation between active and passive sentences.** In English it could be frequent to use the passive forms of sentences, while being translated into other languages the forms will be transformed into active types, where the words and phrases in the whole sentences can be reversed.
- **Transformation between clause and main sentence.** In English the attributive clauses and noun clauses are often used as subordinate components, while they can be translated into two parallel clauses in other languages.
- **Change of reading order of sentences.** The majority of the languages in this world have the reading order of from-left-to-right, such as English, French, etc. But some languages, e.g., under Afro-Asiatic family, Arabic, Hebrew, Persian, Sindhi and Urdu languages read from right to left.

3.2 UD Bias Statistics

In Fig. 3 we present the statistics of such bias between the parallel UD trees in different languages, such as the misaligned words, mismatched UD ($w_i^{\frown} w_j$) pair and UD path of ($e_s^{\frown} \cdots {}^{\frown} e_o$) relational pair. Fig. 3(a) reveals that languages under different families show distinct divergences. And the more different of languages, the greater the divergences (e.g., English to Arabic). Fig. 3(b) indicates that complex sentences (e.g., compound sentences) bring larger bias; and in the real world, complex sentences are much more ubiquitous than simple ones. Also, the mismatch goes worse when the UD core predicates are nouns instead of verbs.



(a) Mismatch between different language pairs

(b) Mismatch between different types of sentences

Figure 3: Statistics of mismatching items of UD trees.

4 Code-mixed UD Forest Construction

To eliminate such discrepancies for unbiased UD-feature transfer, we build the code-mixed UD forests, via the following six steps.

► **Step 1: translating a sentence x^{Src} in SRC language to the one x^{Tgt} in TGT language.**[1] This step is to generate a pseudo parallel sentence pair in both TGT and SRC languages. We accomplish this by using the state-of-the-art *Google Translation API*.[2] We denote the parallel sentences as <x^{Src},$\overline{x^{\text{Tgt}}}$> or <$\overline{x^{\text{Src}}}$,$x^{\text{Tgt}}$>.

► **Step 2: obtaining the word alignment scores.** Meanwhile, we employ the Awesome-align toolkit[3] to obtain the word alignment confidence M=$\{m_{i\leftrightarrow j}\}$ between word pair $w_i \in x^{\text{Src}}$ and $w_j \in \overline{x^{\text{Tgt}}}$ in parallel sentences.

► **Step 3: parsing UD trees for parallel sentences.** Then, we use the UD parsers in SRC and SRC languages respectively to parse the UD syntax trees for two parallel sentences, respectively. We adopt the UDPipe[4] as our UD parsers, which are trained separately on different UD annotated data[5]. We denote the SRC UD tree as $\mathcal{T}^{\text{Src}}$, and the pseudo TGT UD tree as $\overline{\mathcal{T}^{\text{Tgt}}}$. Note that the UD trees in all languages share the same dependency labels,

[1]Vice versa for the direction from TGT to SRC language.
[2]https://translate.google.com, Sep. 10 2022
[3]https://github.com/neulab/awesome-align
[4]https://github.com/bnosac/udpipe, Universal Dependencies 2.3 models: english-ewtud-2.3-181115.udpipe, chinese-gsd-ud-2.3-181115.udpipe, arabic-padt-ud-2.3-181115.udpipe.
[5]https://universaldependencies.org/

Algorithm 1 Process of constructing code-mixed UD forests

Input: T^{SRC}, T^{TGT}, M, threshold θ, empty forest $\mathcal{F} = \Phi$.
Output: Code-mixed UD forest $\mathcal{F}$.

```
1: def Construct ($\mathcal{T}^{\text{SRC}}$, $\mathcal{T}^{\text{TGT}}$, $M$, $\mathcal{F}$)        ▷ breadth-first top-down traverse.
2:   is_root = True        ▷ a flag for traversing the predicate only once.
3:   $\mathcal{F}.w_{cur}$ = ROOT        ▷ creating ROOT node for $\mathcal{F}$.
4:   opt_nodes = Queue.Init()        ▷ creating a queue for breadth-first search.
5:   while ($\mathcal{T}^{\text{SRC}} \neq \Phi$) or ($\mathcal{T}^{\text{TGT}} \neq \Phi$) or (opt_nodes$\neq \Phi$) do
6:     if is_root then
7:       $w_{merged}$ = Merge($\mathcal{T}^{\text{SRC}}$.ROOT, $\mathcal{T}^{\text{TGT}}$.ROOT)        ▷ merging from ROOT in $\mathcal{T}^{\text{SRC}}$ and $\mathcal{T}^{\text{TGT}}$.
8:       $w_{merged}$.next$^{\text{SRC}}$ = $\mathcal{T}^{\text{SRC}}$.ROOT.GetChildNodes()
9:       $w_{merged}$.next$^{\text{TGT}}$ = $\mathcal{T}^{\text{TGT}}$.ROOT.GetChildNodes()
10:      $\mathcal{F}.w_{cur}$.SetChild($w_{merged}$, ‘root’)
11:      opt_nodes.enqueue($w_{merged}$)
12:      is_root = False
13:    else
14:      $\mathcal{F}.w_{cur}$ = opt_nodes.dequeue()
15:      aligned_pairs, nonaligned_nodes = AlignSearch($\mathcal{F}.w_{cur}$.next$^{\text{SRC}}$, $\mathcal{F}.w_{cur}$.next$^{\text{TGT}}$, $M$)
16:      for ( $w_i^{\text{SRC}}$, $w_j^{\text{TGT}}$, $arc$ ) $\in$ aligned_pairs do
17:        $w_{merged}$ = Merge($w_i^{\text{SRC}}$, $w_j^{\text{TGT}}$)
18:        $w_{merged}$.next$^{\text{SRC}}$ = $w_i^{\text{SRC}}$.GetChildNodes()
19:        $w_{merged}$.next$^{\text{TGT}}$ = $w_j^{\text{TGT}}$.GetChildNodes()
20:        $\mathcal{F}.w_{cur}$.SetChild($w_{merged}$, $arc$)
21:        opt_nodes.enqueue($w_{merged}$)
22:      end for
23:      for $w_i \in$ nonaligned_nodes do
24:        $\mathcal{F}.w_{cur}$.SetChild($w_i$, $w_i.arc$)        ▷ action ‘Coping into forest’ for non-aligned words.
25:      end for
26:    end if
27:  end while
28:  return $\mathcal{F}$
29: def Merge ($w_a^{\text{SRC}}$, $w_b^{\text{TGT}}$)        ▷ action ‘Merging into forest’ for aligned words.
30:   return $w_b^{\text{TGT}}$        ▷ for two aligned word, returning the TGT-side word.
31: def AlignSearch (nodes_a, nodes_b, $M$)        ▷ preparing the aligned word pairs in $\mathcal{T}^{\text{SRC}}$ and $\mathcal{T}^{\text{TGT}}$.
32:   aligned_pairs = []
33:   for $m_{i\leftrightarrow j} \in M$ do
34:     if $m_{i\leftrightarrow j} > \theta$ then
35:       aligned_pairs.Append(nodes_a[i], nodes_b[j], nodes_b[i].$arc$ )
36:       nodes_a.Remove($w_i$)
37:       nodes_a.Remove($w_j$)
38:     end if
39:   end for
40:   nonaligned_nodes = nodes_a.union(nodes_b)        ▷ words with no salient alignments.
41:   return aligned_pairs, nonaligned_nodes
```

i.e., with the same (as much as possible) annotation standards. In Appendix §A we list the dependency labels which are the commonly occurred types.

► **Step 4: projecting and merging the labels of training data.** For the training set, we also need to project the annotations (relational subject-object pairs) of sentences in SRC languages to TGT pseudo sentences. Note that this step is not needed for the testing set. The projection is based on the open source[6], during which the word alignment scores at step-2 are used. We can denote the SRC annotation as y, and the pseudo TGT label as $\overline{y}$. We then merge the annotation from both SRC and TGT viewpoints, into the code-mixed one Y, for later training use. Specifically, for the node that is kept in the final code-mixed forest, we will keep its labels; and for those nodes that are filtered, the annotations are replaced by their correspondences.

▶ **Step 5: merging the SRC and TGT UD trees into a code-mixed forest.** Finally, based on the SRC UD tree and the TGT UD tree, we construct the code-mixed UD forest. We mainly perform breadth-first top-down traversal over each pair of nodes $\mathcal{T}^{\texttt{Src}}$ and $\mathcal{T}^{\overline{\texttt{Tgt}}}$, layer by layer. The traversal starts from their *ROOT* node. We first create a *ROOT* node as the initiation of the code-mixed forest. We design two types of actions for the forest merging process:

- **`Merging`** current pair of nodes $w_i \in \mathcal{T}^{\texttt{Src}}$ from SRC tree and $w_j \in \mathcal{T}^{\overline{\texttt{Tgt}}}$ from TGT tree into the forest $\mathcal{F}$, if the current two nodes are confidently aligned at same dependency layer. We check the word alignment confidence $m_{i \leftrightarrow j}$ between the two nodes, and if the confidence is above a pre-defined threshold θ, i.e., $m_{i \leftrightarrow j} > \theta$, we treat them as confidently aligned.
- **`Copying`** current node from SRC tree $\mathcal{T}^{\texttt{Src}}$ or TGT tree $\mathcal{T}^{\overline{\texttt{Tgt}}}$ into the forest $\mathcal{F}$, once the node has no significant alignment in the opposite tree at this layer.

In Algorithm 1 we formulate in detail the process of code-mixed forest construction. Also, we note that when moving the nodes from two separate UD trees into the forest, the attached dependency labels are also copied. When two nodes are merged, we only choose the label of the TGT-side node. Finally, the resulting forest $\mathcal{F}$ looks like code-mixing, and is structurally compact.

▶ **Step 6: assembling code-mixed texts.** Also we need to synthesize a code-mixed text X based on the raw SRC text $x^{\texttt{Src}}$ and the pseudo TGT text $x^{\overline{\texttt{Tgt}}}$. The code-mixed text X will also be used as inputs together with the forest, into the forest encoder. We directly replace the SRC words with the TGT words that have been determined significantly aligned at Step-5.

[6]`https://github.com/scofield7419/XSRL-ACL`

5 XRE with Code-mixed UD Forest

Along with the UD forest $\mathcal{F}^{\texttt{Src}}$, we also assemble the code-mixed sequential text $X^{\texttt{Src}}$ from the SRC and translated pseudo-TGT sentences (i.e., $x^{\texttt{Src}}$ and $x^{\overline{\texttt{Tgt}}}$), and the same for the TGT sentences $X^{\texttt{Tgt}}$. An XRE system, being trained with SRC-side annotated data (<$X^{\texttt{Src}}$, $\mathcal{F}^{\texttt{Src}}$>, $Y^{\texttt{Src}}$), needs to determine the label $Y^{\texttt{Tgt}}$ of relational pair $e_s \overset{r}{\frown} e_o$ given a TGT sentence and UD forest (<$X^{\texttt{Tgt}}$, $\mathcal{F}^{\texttt{Tgt}}$>).

The XRE system takes as input X=$\{w_i\}_n$ and $\mathcal{F}$. We use the multilingual language model (MLM) for representing the input code-mixed sentence X:

$$\boldsymbol{H} = \{\boldsymbol{h}_1, \cdots, \boldsymbol{h}_n\} = \text{MLM}(X)\,, \tag{1}$$

where X is the code-mixed sentential text. We then formulate the code-mixed forest $\mathcal{F}$ as a graph, G=<E, V>, where E=$\{e_{i,j}\}_{n\times n}$ is the edge between word pair (i.e., initiated with $e_{i,j}$=0/1, meaning dis-/connecting), V=$\{w_i\}_n$ are the words. We main the node embeddings $\boldsymbol{r}_i$ for each node v_i. We adopt the GAT model (Velickovic et al., 2018) for the backbone forest encoding:

$$\rho_{i,j} = \text{Softmax}(\text{GeLU}(\boldsymbol{U}^T[\boldsymbol{W}_1\boldsymbol{r}_i; \boldsymbol{W}_2\boldsymbol{r}_j]))\,, \tag{2}$$

$$\boldsymbol{u}_i = \sigma(\sum_j \rho_{i,j}\boldsymbol{W}_3\boldsymbol{r}_j^1)\,, \tag{3}$$

where $\boldsymbol{W}_{3/4/5}$ and $\boldsymbol{U}$ are all trainable parameters. σ is the sigmoid function. GeLU is a Gaussian error linear activation function. Note that the first-layer representations of $\boldsymbol{r}_i$ is initialized with $\boldsymbol{h}_i$. $\boldsymbol{H}$ and $\boldsymbol{U}$ are then concatenated as the resulting feature representation:

$$\hat{\boldsymbol{H}} = \boldsymbol{H} \oplus \boldsymbol{U}\,. \tag{4}$$

XRE aims to determine the semantic relation labels between two given mention entities. For example, given a sentence '*John Smith works at Google*', RE should identify that there is a relationship of "works at" between the entities "John Smith" and "Google". Our XRE model needs to predict the relation label y. We adopt the biaffine decoder (Dozat and Manning, 2017) to make prediction:

$$y = \text{Softmax}(\boldsymbol{h}_s^T \cdot \boldsymbol{W}_1 \cdot \boldsymbol{h}_o + \boldsymbol{W}_2 \cdot \text{Pool}(\hat{\boldsymbol{H}}))\,. \tag{5}$$

Here both $\boldsymbol{h}_s$ and $\boldsymbol{h}_o$ are given.

6 Experiments

6.1 Setups

We consider the ACE05 (hristopher Walker et al., 2006) dataset, which includes English (EN), Chinese (ZH) and Arabic (AR). We give the data statistics in Table 1 The multilingual BERT is used.[7]

[7]`https://huggingface.co`, base, cased version

Language	Train	Dev	Test
EN	479	60	60
ZH	507	63	63
AR	323	40	40

Table 1: Data statistics. The numbers are documents.

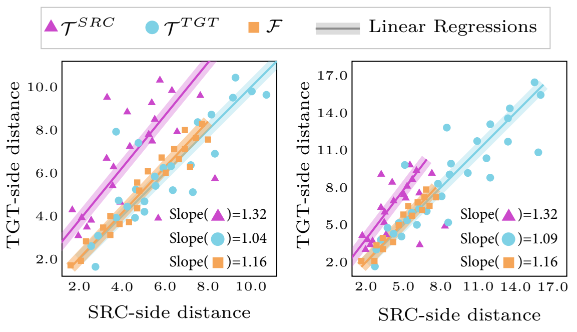
We use two-layer GAT for forest encoding, with a 768-d hidden size. We mainly consider the transfer from EN to one other language. Following most cross-lingual works (Fei et al., 2020b; Ahmad et al., 2021), we train the XRE model with fixed 300 iterations without early-stopping. We make comparisons between three setups: 1) using only raw SRC training data with the model transfer, 2) using only the pseudo TGT (via annotation projection) for training, and 3) using both the above SRC and TGT data. Each setting uses both the texts and UD tree (or forest) features. The baseline uses the same GAT model for syntax encoding, marked as *Syn-Baseline*. For setup 1)&2) we also test the transfer with only text inputs, removing the syntax features, marked as *TxtBaseline*. Besides, for setup 1) we cite current SoTA performances as references. We use F1 to measure the RE performance, following Ahmad et al. (2021). All experiments are undergone five times and the average value is reported.

6.2 Data Inspection

We also show in Table 3 the differences in average sequential and syntactic (shortest dependency path) distances between the subjects and objects of the relational triplets. As seen, the syntactic distances between subject-object pairs are clearly shortened in the view of syntactic dependency trees, which indicates the imperative to incorporate the tree structure features. However, the syntactic distances between different languages vary, i.e., more complex languages have longer syntactic distances. Such discrepancy reflects the necessity of employing our proposed UD debiasing methods to bridge the gap.

6.3 Main Results

From Table 2, we can see that UD features offer exceptional boosts (M1 vs. M2, M4 vs. M5). And annotation projection methods outperform model transfer ones (i.e., M1&M2&M3 vs. M4&M5) by offering direct TGT-side features. Interestingly, in both two transfer paradigms, the improvements from UD become weak on the language pairs with



(a) SRC(EN) → TGT(ZH)   (b) SRC(EN) → TGT(AR)

Figure 4: Change of syntax distance (shortest path) of relational pair in different UD trees.

bigger divergences. For example, the improvement on EN→DE outweighs the ones on EN→ZH. Furthermore, using our proposed code-mixed syntax forests is significantly better than using standalone SRC or TGT (or the simple combination) UD features (M7 vs. M2&M5&M6) on all transfers with big margins. For example, our system outperforms SoTA UD-based systems with averaged +4.8%(=67.2-62.4) F1. This evidently verifies the necessity to create the code-mixed forests, i.e., bringing unbiased UD features for transfer. Also, we find that the more the difference between the two languages, the bigger the improvements from forests. The ablation of code-mixed texts also shows the contribution of the sequential textual features, which indirectly demonstrates the larger efficacy of the structural code-mixed UD forests.

6.4 Probing Unbiasedness of Code-mixed UD Forest

Fig. 4 plots the change of the syntax distances of RE pairs during the transfer with different syntax trees. We see that the use of SRC UD trees shows clear bias (with larger inclination angles) during the transfer, while the use of TGT UD trees and code-mixed forests comes with less change of syntax distances. Also, we can see from the figure that the inference paths between objects and subjects of RE tasks are clearly shortened with the forests (in orange color), compared to the uses of SRC/TGT UD trees.

6.5 Change during Code-mixed UD Forest Merge

Here we make statistics of how many words are merged and kept during the UD tree merging, respectively. The statistics are shown in Table 4. We can see that the distance between EN-ZH is shorter

		SRC	TGT	EN→ZH	EN→AR	AVG
▶ **Model Transfer**						
M1	TxtBaseline	✓		55.8	63.8	59.8
M2	SynBaseline(+$\mathcal{T}$)	✓		59.2	65.2	62.2 (+2.4)
M3	SoTA XRE	✓		58.0	66.8	62.4
▶ **Annotation Projection**						
M4	TxtBaseline		✓	58.3	66.2	62.3
M5	SynBaseline(+$\mathcal{T}$)		✓	61.4	67.4	64.4 (+2.1)
▶ **Model Transfer + Annotation Projection**						
M6	SynBaseline(+$\mathcal{T}$)	✓	✓	57.8	64.0	60.9
M7 (Ours)	SynBaseline(+$\mathcal{F}$)	✓	✓	63.7	70.7	67.2 (+6.3)
M8	w/o code-mixed text	✓	✓	61.6	68.2	64.9 (-2.3)

Table 2: Main results of cross-lingual RE transfer tasks from English language to other languages, by different models and features. M6 uses two separate instances (texts and UD trees) for training, including the raw SRC one and the pseudo TGT one. M7 uses the SRC-TGT merged one as ours, i.e., code-mixed texts and forests.

EN	ZH	AR
•Sequential Distance		
4.8	3.9	25.8
•Syntactic Distance		
2.2	2.6	5.1

Table 3: Sequential and syntactic (shortest dependency path) distances (words) between the subjects and objects of the relational triplets.

than that between EN-AR. For example, the length of code-mixed EN-ZH UD forests (sentences) is 31.63, while for EN-AR the length is 40.44. Also, EN-ZH UD forests have a higher to 21.4% merging rate, while EN-AR UD forests have 16.6% merging rate. This demonstrates that the more divergences of languages, the lower the merging rate of the code-mixed forest.

6.6 Impacts of θ on Controlling the Quality of Merged Forest

In §4 of step-5, we describe that we use a threshold θ to control the aligning during the UD tree merging. Intuitively, the large the threshold θ, the lower the alignment rate. When $\theta \to 0$, most of the SRC and TGT nodes in two parallel UD trees can find their counterparts but the alignments are most likely to be wrong, thus hurting the quality of the resulting code-mixed UD forests. When $\theta \to 1$, none of the SRC and TGT nodes in two parallel UD trees can be aligned, and both two UD trees are copied and co-existed in the resulting code-mixed UD forests. In such case, the integration of such forests is equivalent to the annotation projection methods where we directly use both the raw SRC UD feature and the translated pseudo TGT UD tree feature. In Fig. 5 we now study the influences of using different code-mixed forest features generated with different merging rates (θ). We see that with a threshold of θ=0.5, the performances are consistently the best.

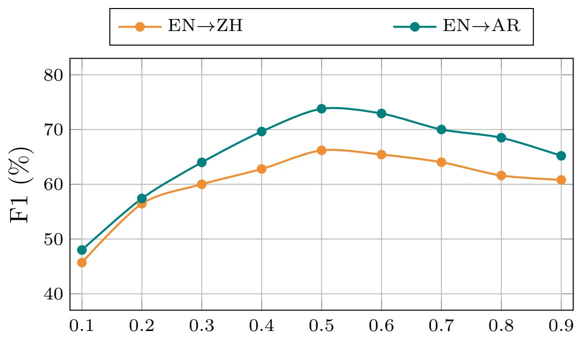


Figure 5: Transfer performances by using code-mixed forests generated with different merging rates (θ).

6.7 Performances on Different Types of Sentence

In Table 5 we show the results under different types of sentences. We directly select 500 short sentences (with length < 12) as simple sentences; and select 500 lengthy sentences (with length > 35) as complex sentences. As can be seen, with the code-mixed forest features, the system shows very notable improvements in complex sentences. For example, on the EN→ZH we obtain 15.9(=57.2-41.3)% F1 improvement, and on the EN→AR the boost increases strikingly to 25.2(=67.3-42.1)% F1. However, such enhancements are not very significant in handling simple sentences. This indicates that the code-mixed UD forest features can espe-

	Words per Sentence				
	Before Merging			After Merging	
	SRC (EN)	TGT	Sum	Code-mixed	Merged (Rate)
EN-ZH	15.32	24.91	40.23	31.63	8.6 (21.4%)
EN-AR	15.32	33.12	48.44	40.44	8.0 (16.6%)

Table 4: The statistics of the words before and after constructing code-mixed data.

	EN→ZH	EN→AR
• **Simple Sentence**		
SynBaseline(+$\mathcal{T}^{SRC}$)	66.1	78.2
SynBaseline(+$\mathcal{T}^{TGT}$)	68.7	80.6
SynBaseline(+$\mathcal{F}$)	71.3	82.4
• **Complex Sentence**		
SynBaseline(+$\mathcal{T}^{SRC}$)	39.5	37.4
SynBaseline(+$\mathcal{T}^{TGT}$)	41.3	42.1
SynBaseline(+$\mathcal{F}$)	57.2	67.3

Table 5: Comparisons under different types of sentences.

cially enhance the effectiveness on the hard case, i.e., the transfer between those pairs with greater divergences will receive stronger enhancements from our methods.

7 Conclusion and Future Work

Universal dependencies (UD) have been served as effective language-consistent syntactic features for cross-lingual relation extraction (XRE). In this work, we reveal the intrinsic language discrepancies with respect to the UD structural annotations, which limit the utility of the UD features. We enhance the efficacy of UD features for an unbiased UD-based transfer, by constructing code-mixed UD forests from both the source and target UD trees. Experimental results demonstrate that the UD forests effectively debias the syntactic disparity in the UD-based XRE transfer, especially for those language pairs with larger gaps.

Leveraging the syntactic dependency features is a long-standing practice for strengthening the performance of RE tasks. In this work, we propose a novel type of syntactic feature, code-mixed UD forests, for cross-lingual relation extraction. We note that this feature can be applied broadly to other cross-lingual structured information extraction tasks that share the same task definition besides RE, such as event detection (ED) (Halpin and Moore, 2006) and semantic role labeling (SRL) (Gildea and Jurafsky, 2000). Besides, how to further increase the utility of the UD forests with a better modeling method is a promising research direction, i.e., filtering the noisy structures in the UD forests.

Acknowledgments

This research is supported by the National Natural Science Foundation of China (No. 62176180), and also the Sea-NExT Joint Lab.

Limitations

Although showing great prominence, our proposed method has the following limitations. First of all, our method relies on the availability of annotated UD trees of both the source and target languages, as we need to use the annotations to parse the syntax trees for our own sentences. Fortunately, UD project covers over 100 languages, where most of the languages, even the minor ones, will have the UD resources. At the same time, our method will be influenced by the quality of UD parsers. Secondly, our method also uses the external translation systems to produce the pseudo parallel sentences, where our method may largely subject to the quality of the translators. Again luckily, current neural machine translation systems have been well developed and established, i.e., Google Translation. Only when handling very scare languages where the current translation systems fail to give satisfactory performances, our method will fail.

Ethics Statement

In this work, we construct a type of code-mixed UD forest based on the existing UD resources. We note that all the data construction has been accomplished automatically, and we have not created any new annotations with additional human labor. Specifically, we use the UD v2.10 resource, which is a collection of linguistic data and tools that are open-sourced. Each of treebanks of UD has its own license terms, including the *CC BY-SA 4.0*[8] and *CC BY-NC-SA*

[8] http://creativecommons.org/licenses/by-sa/4.0/

2.5-4.0[9] as well as *GNU GPL 3.0*[10]. Our use of UD treebanks comply with all these license terms is at non-commercial purpose. The software tools (i.e., UDPipe parsers) are provided under *GNU GPL V2*. Our use of UDPipe tools complies with the term.

References

Wasi Uddin Ahmad, Nanyun Peng, and Kai-Wei Chang. 2021. GATE: graph attention transformer encoder for cross-lingual relation and event extraction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 12462–12470.

Suman Banerjee, Nikita Moghe, Siddhartha Arora, and Mitesh M. Khapra. 2018. A dataset for building code-mixed goal oriented conversation systems. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3766–3780.

Anders Björkelund, Love Hafdell, and Pierre Nugues. 2009. Multilingual semantic role labeling. In *Proceedings of the CoNLL*, pages 43–48.

Duy-Cat Can, Hoang-Quynh Le, Quang-Thuy Ha, and Nigel Collier. 2019. A richer-but-smarter shortest dependency path with attentive augmentation for relation extraction. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2902–2912.

Silviu Cucerzan and David Yarowsky. 1999. Language independent named entity recognition combining morphological and contextual evidence. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*.

Aron Culotta and Jeffrey Sorensen. 2004. Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, pages 423–429.

Angel Daza and Anette Frank. 2019. Translate and label! an encoder-decoder approach for cross-lingual semantic role labeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 603–615.

Marie-Catherine de Marneffe, Christopher D. Manning, Joakim Nivre, and Daniel Zeman. 2021. Universal dependencies. *Comput. Linguistics*, 47(2):255–308.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *Proceedings of the 5th International Conference on Learning Representations*.

[9]http://creativecommons.org/licenses/by-nc-sa/4.0/

[10]http://opensource.org/licenses/GPL-3.0

Hao Fei, Fei Li, Bobo Li, and Donghong Ji. 2021. Encoder-decoder based unified semantic role labeling with label-aware syntax. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 12794–12802.

Hao Fei, Shengqiong Wu, Jingye Li, Bobo Li, Fei Li, Libo Qin, Meishan Zhang, Min Zhang, and Tat-Seng Chua. 2022. Lasuie: Unifying information extraction with latent adaptive structure-aware generative language model. In *Proceedings of the Advances in Neural Information Processing Systems, NeurIPS 2022*, pages 15460–15475.

Hao Fei, Meishan Zhang, and Donghong Ji. 2020a. Cross-lingual semantic role labeling with high-quality translated training corpus. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7014–7026.

Hao Fei, Meishan Zhang, Fei Li, and Donghong Ji. 2020b. Cross-lingual semantic role labeling with model transfer. *IEEE ACM Trans. Audio Speech Lang. Process.*, 28:2427–2437.

Daniel Gildea and Daniel Jurafsky. 2000. Automatic labeling of semantic roles. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 512–520.

Harry Halpin and Johanna D. Moore. 2006. Event extraction in a plot advice agent. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 857–864.

hristopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2006. Ace 2005 multilingual training corpus. In *Proceedings of Philadelphia: Linguistic Data Consortium*.

Aditya Joshi, Ameya Prabhu, Manish Shrivastava, and Vasudeva Varma. 2016. Towards sub-word level compositions for sentiment analysis of Hindi-English code mixed text. In *Proceedings of the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2482–2491.

Seokhwan Kim, Minwoo Jeong, Jonghoon Lee, and Gary Geunbae Lee. 2010. A cross-lingual annotation projection approach for relation detection. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 564–571.

Mikhail Kozhevnikov and Ivan Titov. 2013. Cross-lingual transfer of semantic role labeling models. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 1190–1200.

Igor Labutov and Hod Lipson. 2014. Generating code-switched text for lexical learning. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 562–571.

Chenwei Lou, Jun Gao, Changlong Yu, Wei Wang, Huan Zhao, Weiwei Tu, and Ruifeng Xu. 2022. Translation-based implicit annotation projection for zero-shot cross-lingual event argument extraction. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2076–2081.

Di Lu, Ananya Subburathinam, Heng Ji, Jonathan May, Shih-Fu Chang, Avi Sil, and Clare Voss. 2020. Cross-lingual structure transfer for zero-resource event extraction. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 1976–1981.

Ryan McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith Hall, Slav Petrov, Hao Zhang, Oscar Täckström, Claudia Bedini, Núria Bertomeu Castelló, and Jungmee Lee. 2013. Universal dependency annotation for multilingual parsing. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 92–97.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using LSTMs on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 1105–1116.

Phoebe Mulcaire, Swabha Swayamdipta, and Noah A. Smith. 2018. Polyglot semantic role labeling. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 667–672.

Jian Ni and Radu Florian. 2019. Neural cross-lingual relation extraction based on bilingual word embedding mapping. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 399–409.

Sebastian Padó and Mirella Lapata. 2009. Cross-lingual annotation projection for semantic roles. *J. Artif. Intell. Res.*, 36:307–340.

Libo Qin, Minheng Ni, Yue Zhang, and Wanxiang Che. 2020. Cosda-ml: Multi-lingual code-switching data augmentation for zero-shot cross-lingual NLP. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pages 3853–3860.

Bidisha Samanta, Niloy Ganguly, and Soumen Chakrabarti. 2019. Improved sentiment detection via label transfer from monolingual to synthetic code-switched text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3528–3537.

Ananya Subburathinam, Di Lu, Heng Ji, Jonathan May, Shih-Fu Chang, Avirup Sil, and Clare Voss. 2019. Cross-lingual structure transfer for relation and event extraction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 313–325.

Nasrin Taghizadeh and Heshaam Faili. 2021. Cross-lingual adaptation using universal dependencies. *ACM Trans. Asian Low Resour. Lang. Inf. Process.*, 20(4):65:1–65:23.

Nasrin Taghizadeh and Heshaam Faili. 2022. Cross-lingual transfer learning for relation extraction using universal dependencies. *Comput. Speech Lang.*, 71:101265.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *Proceedings of the International Conference on Learning Representations*.

Shengqiong Wu, Hao Fei, Yafeng Ren, Donghong Ji, and Jingye Li. 2021. Learn from syntax: Improving pair-wise aspect and opinion terms extraction with rich syntactic knowledge. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pages 3957–3963.

Min Xiao and Yuhong Guo. 2015. Annotation projection-based representation learning for cross-lingual dependency parsing. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*, pages 73–82.

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2002. Kernel methods for relation extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 71–78.

Meishan Zhang, Yue Zhang, and Guohong Fu. 2019. Cross-lingual dependency parsing using code-mixed TreeBank. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 997–1006.

Yuhao Zhang, Peng Qi, and Christopher D. Manning. 2018. Graph convolution over pruned dependency trees improves relation extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2205–2215.

Zhisong Zhang, Emma Strubell, and Eduard Hovy. 2021. On the benefit of syntactic supervision for cross-lingual transfer in semantic role labeling. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6229–6246.

Ranran Zhen, Rui Wang, Guohong Fu, Chengguo Lv, and Meishan Zhang. 2021. Chinese opinion role labeling with corpus translation: A pivot study. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10139–10149.

A The universal dependency labels

In Table 6, we list the dependency labels which are the commonly occurred types. Please refer to Stanford dependency[11] for more details about the dependency labels.

Dependency Label	Description
amod	adjectival modifier
advcl	adverbial clause modifier
advmod	adverb modifier
acomp	adjectival complement
auxpass	passive auxiliary
compound	compound
ccomp	clausal complement
cc	coordination
conj	conjunct
cop	copula
det	determiner
dep	dependent
dobj	direct object
mark	marker
nsubj	nominal subject
nmod	nominal modifier
neg	negation modifier
xcomp	open clausal complement

Table 6: The universal dependency labels.

[11]https://nlp.stanford.edu/software/dependencies_manual.pdf

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
7

☑ A2. Did you discuss any potential risks of your work?
7

☑ A3. Do the abstract and introduction summarize the paper's main claims?
1

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

6

☑ B1. Did you cite the creators of artifacts you used?
6

☑ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
9&Appendix-B

☑ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
9

☑ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Appendix-A

☑ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Appendix-A&B

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
6&Appendix-B

C ☑ Did you run computational experiments?

6&Appendix-B

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
6&Appendix-B

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
6&Appendix-B

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
6&Appendix-B

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
6&Appendix-B

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?
Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
Not applicable. Left blank.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Not applicable. Left blank.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
Not applicable. Left blank.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Not applicable. Left blank.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Not applicable. Left blank.