# Typology Guided Multilingual Position Representations: Case on Dependency Parsing

**Tao Ji, Yuanbin Wu, Xiaoling Wang**
School of Computer Science and Technology,
East China Normal University
{taoji@stu,ybwu@cs,xlwang@cs}.ecnu.edu.cn

## Abstract

Recent multilingual models benefit from strong unified semantic representation models. However, due to conflict linguistic regularities, ignoring language-specific features during multilingual learning may suffer from negative transfer. In this work, we analyze the relation between a language's position space and its typological characterization, and suggest deploying different position spaces for different languages. We develop a position generation network which combines prior knowledge from typology features and existing position vectors. Experiments on the multilingual dependency parsing task show that the learned position vectors exhibit meaningful hidden structures, and they can help achieving the best multilingual parsing results.

## 1 Introduction

With the recent progress on multilingual text representations, there has been a growing interest in developing unified models for NLP tasks crossing different languages (Ammar et al., 2016; Zeman et al., 2018; Conneau et al., 2020). For high-resource languages, a unified multilingual model is faster to train and easier to deploy than a bunch of independent monolingual models. For low(zero)-resource languages, a multilingual model may help building positive knowledge transfer among languages.

Words and their positions in the text are two main features of any language's sentences. For words, various multilingual pre-trained models (Devlin et al., 2019) and alignment algorithms (Lample et al., 2017) have been devoted to unifying lexical semantic spaces among languages. For positions, however, there are much less study on their roles in joint multilingual learning: models simply adopt one universal position representation for all languages. Since word positions describe word orders, a single position space implies all languages are compiled under the same word order system, which is not true according to linguistic prior. For example, adjectives are usually placed before nouns in English, while in French they are almost after nouns. Such conflicts of linguistic regularities may break the effectiveness of word position features in multilingual learning (especially for those tasks sensitive to word order (e.g., syntactic and semantic parsing (Ji et al., 2021))).

In this paper, we study the connection between a language's position space and its typological characterization (especially, word order characterization). By jointly learning position spaces with a syntactic parsing task, we first have two findings.

- When position representations are separately learned on each language, they can effectively help identifying the language's typological feature on word order (e.g., noun-adjective or adjective-noun). Therefore, by replacing the universal position space with language-specific ones, we have more room for handling different linguistic regularities.
- The distances deduced from individually learned position representations correlate well with languages' typological distances (e.g., position spaces of SVO languages and SOV languages are apart). Therefore, customized position spaces provide a clear and acknowledged path for positive transfer in multilingual learning.

We next develop methods to construct multilingual position representations. Options may include attaching language ids to the existing universal position space (Östling and Tiedemann, 2017) and learning position representations from scratch (Bjerva et al., 2019). One main concern on those approaches is on handling unseen languages: if a language doesn't appear in the training set, its position representations are totally unknown.

Our key technical contribution is a generation network for positions. It explicitly takes word typological features of a language as input and outputs



*Findings of the Association for Computational Linguistics: ACL 2023*, pages 13524–13541

a set of position vectors for that language. For unseen languages, we are free to obtain their position vectors through prior on their typology. During the generation process, we take the universal position representations as basis vectors for each language's position space, which makes the learned vectors still carry the prior of "representing a position in texts". Under this setting, we are able to examine the shift of languages' position spaces with a unified coordinate system.

We take multilingual dependency parsing as our demonstrating task. The parser is trained on 13 languages from the universal dependencies treebanks, and is tested with both languages present (13) and absent (30) in the training set. The results show that, with the typological guided position vectors, the parser is able to achieve both significant improvements on seen (+4.1 LAS) and unseen (+1.2 LAS) languages compared with using universal position representations.

## 2 Preliminary

Our multilingual models take Transformer (Vaswani et al., 2017) as basic building blocks. There are two types of position spaces, absolute and relative.

**Absolute Position Representation** Given a sequence of word vectors $\boldsymbol{x}_{0:n} = (\boldsymbol{x}_0, \boldsymbol{x}_1, \cdots, \boldsymbol{x}_n)$, for each absolute position $i$, there is a position vector $\boldsymbol{a}_i$. The vector could be obtained by a lookup function $\boldsymbol{a}_i = \texttt{lookup}(\mathrm{E}_{abs}, i)$, where $\mathrm{E}_{abs}$ is a learnable matrix (Gehring et al., 2017), or by a fix sinusoidal function (Vaswani et al., 2017),

$$\boldsymbol{a}_i[j] = \begin{cases} \texttt{sin}(\omega_k \cdot i), & \texttt{if } j = 2k \\ \texttt{cos}(\omega_k \cdot i), & \texttt{if } j = 2k+1, \end{cases} \quad (1)$$
$$\omega_k = 1/10000^{2k/d}$$

Absolution position vectors are usually added to the input word vectors.

**Relative Position Representation** Relative position is another widely applied positional feature (Shaw et al., 2018; Dai et al., 2019; Wang et al., 2020). Like absolute positions, for each relative position $i$, a relative vector $\boldsymbol{r}_i$ is obtained from a lookup function $\boldsymbol{r}_i = \texttt{lookup}(\mathrm{E}_{rel}, i)$. Commonly, relative positions are clipped in a small range $\{-k, -k+1, \ldots, k\}$. Unlike absolute positions, relative position vectors usually access the Transformer block in self-attention layers, specifically, in the computation of attention scores $\alpha_{ij}$ between two words $i, j$ and output hidden vectors $\boldsymbol{o}_i$,

$$\alpha_{ij} \propto \boldsymbol{x}_i \cdot \mathrm{W}^Q \cdot \left(\boldsymbol{x}_j \cdot \mathrm{W}^K + \boldsymbol{r}_{j-i}\right)^\top$$
$$\boldsymbol{o}_i = \sum_j \alpha_{ij} \cdot \left(\boldsymbol{x}_j \cdot \mathrm{W}^V + \boldsymbol{r}_{j-i}\right), \quad (2)$$

where $\mathrm{W}^Q, \mathrm{W}^K$ and $\mathrm{W}^V$ are parameter matrices. Relative position representation can be shared among multiple heads and layers of all self-attention modules.

**Multilingual Dependency Parsing** A dependency parser extracts arcs (head $i$, dependent $j$, relation $r$) among words in sentences. Given a set of training languages, we train a multilingual parser on the union of training languages' treebanks (high resource). In testing time, we evaluate parsing performances on two sets of languages, those are in the training set and those are not (zero-resource) An ideal multilingual parser would exhibit positive transfer on both high and zero-resource languages.

We use the Transformer network to build the parser. The input word representations are collected from mBERT (Devlin et al., 2019), after passing a Transformer, we use the biaffine scorer (Dozat and Manning, 2017) to score each possible head dependant pair. The performances are evaluated by the head-dependent labeled attachment scores (LAS).

## 3 Position Spaces for Multilingual Learning

Existing mutlilingual models use a universal position space for all languages. It is questionable that whether one position space is enough to handle languages with different linguistic constraints. In order to inspect relations between position representations and typological features, we experiment a multilingual parser with language-specific position vectors. For each language, the model assigns a set of learnable vectors for each position (absolute or relative), and the position vectors are jointly learned with the parser.

First, we examine whether the learned position vectors carry information about word order. Taking the order of subject(S), verb(V) and object(O) as an example, we merge datasets of English_en (SVO) and Hindi_hi (SOV), and train a binary probing classifier to discriminate two word orders. The classifier is a 2-layer MLP taking the mean pooling of

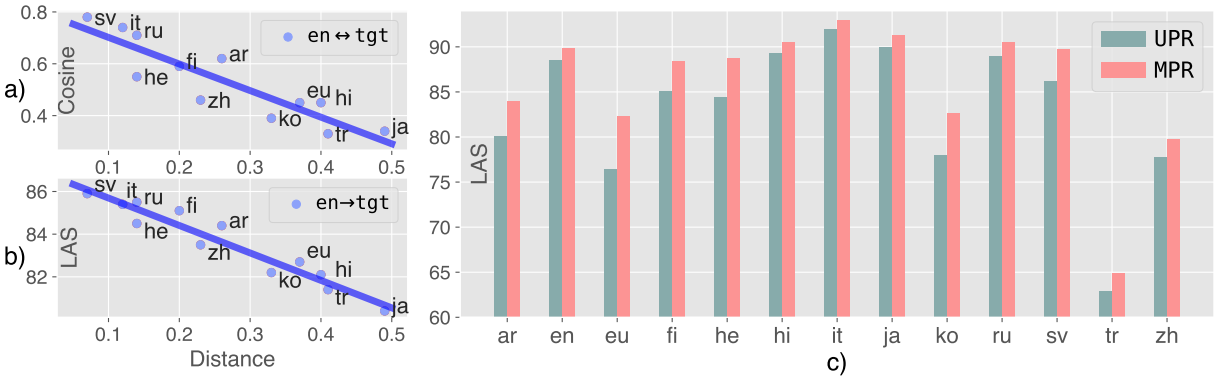


Figure 1: a) The correlation between position space similarities (to English) and linguistic distance. b) The parsing performances on English when substituting with different languages' position vectors. The x-aix is the linguistic distance defined in (Scholivet et al., 2019). c) The parsing accuracies of customized multilingual position vectors (MPR) and universal position vectors (UPR).

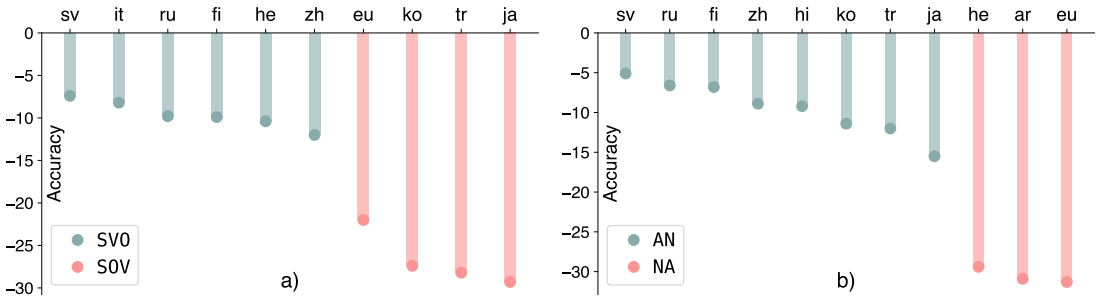


Figure 2: The probing accuracy of different language groups. a) SVO or SOV, b) NA or AN

the parser's final layer hidden vectors as input. Following the general probing workflow, we may use testing accuracies of the classifier to assert whether word order information is encoded. However, a high probing accuracy is not trustworthy here because the vocabulary overlapping of two languages is usually small, and the probing classifier is able to achieve high accuracies by simply ignoring the actual word order features and only recognizing differences of the two distinct vocabularies.

We adopt a different probing strategy. After training the probing classifier on English and Hindi, remaining languages are then divided into two groups, the SVO group (Chinese_zh, Finnish_fi, Hebrew_he, Italian_it, Russian_ru, Swedish_sv) and the SOV group (Basque_eu, Japanese_ja, Korean_ko, Turkish_tr). We replace the position vectors of English with those of the two groups, and investigate the accuracy of SVO recognition on English. If position vectors have successfully learned the concept of word order across different languages, we can expected a better probing performances when the replaced vectors are from the same group. Figure 2 shows that on English, position vectors from SVO languages performs much better than SOV languages. The results on noun-adjective order (NA or AN) are similar.

Second, we can further ask whether the distance between two position spaces reflects the typological distance between two languages. We choose a linguistic distance metric defined by Scholivet et al. (2019). For position spaces, we compute the average cosine similarities of two corresponding position vectors. Figure 1 shows that the two distances are highly correlated: similar languages have similar position spaces. It suggests that the customized position vectors may be consulted for avoiding negative transfer in multilingual learning. In fact, we perform another substitution experiment directly on the learned parser (Figure 1). By replacing English position vectors with distant languages (e.g., Japanese), the parsing performances drop a lot. Therefore if we unify languages with a universal position space, the conflict of language regularities may cause negative transfer.

Finally, we compare overall multilingual parsing performances when learning with universal position vectors and learning with different position vectors. Figure 1 shows that the latter always performs better (+2.8 average LAS).

## 4 Typology-guided Position Generation

Analyses above suggest us to apply different position spaces for different languages. However, naively assigning learnable vectors for positions can not generalized to unseen languages. In order to make the multilingual model applicable to languages not appear in the training set, we propose to generate position vectors under explicit guidance of typological features.

### 4.1 Typological Features

Our first set of typological features are extracted from World Atlas of Language Structures (WALS, (Dryer and Haspelmath, 2013)). WALS is a database of 192 structural properties (phonology, word order, lexicon, etc) of 2,676 languages. We follow (Naseem et al., 2012) to include six word order features (WALS codes 81A, 85A, 86A, 87A, 88A, 89A), which have been discussed in (Zhang and Barzilay, 2015; Ammar et al., 2016).

Table 1 lists the six features. For example, feature 81A indicates the order of subject, object and verb. It takes four values (SOV, SVO, VSO and Mixed). In English, 81A is SVO, while in Japanese, it is SOV. We assign a 3 dimension vector for each feature value. A *typological vector* $\boldsymbol{l}$ of a language is obtained by concatenating value vectors of the six features. The vectors are randomly initialized and will be learned with the multilingual model.

We also experiments with other two sets of typological features. The second set is an extension of above six features provided by (Scholivet et al., 2019) which contains 19 features from WALS. The third feature set is taken from the URIEL typology database (Littell et al., 2017), which is a collection of binary features extracted from multiple typological and phylogenetic databases (WALS, PHOIBLE (Steven et al., 2014), and Glottolog (Hammarström et al., 2021)). This set contains 103 syntactic typological features.

### 4.2 Position Generation

Given the typological vector $\boldsymbol{l}^{(l)}$ of a language $l$, we train position generation networks (joint with the multilingual model) to output position vectors customized for $l$ (absolute position $\boldsymbol{a}_i^{(l)}$ or relative position $\boldsymbol{r}_i^{(l)}$). Throughout the paper, we set the dimension of position vectors be 128, the range of absolute positions be $\{0, 1, \ldots, 127\}$, and the range of relative positions be $\{-4, -3, \ldots, 4\}$. We describe two position generation models, a simple MLP network and a self-attention network enhanced with prior on positions.

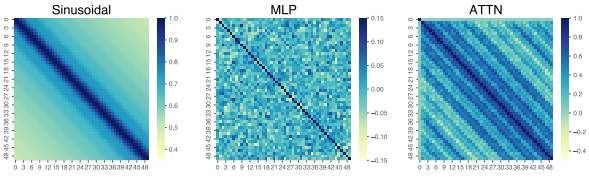


Figure 3: Cosine similarity of the first 50 position vectors. We can see that vectors learned from MLP exhibit loose similarity patterns, while those learned with prior position knowledge and self-attention (ATTN) still keep some property of the sinusoidal position prior (e.g., symmetry). We can also observe that comparing with sinusoidal vectors, self-attention vectors contain more stripes along the diagonal. It means that they contain more locality constraints: if position $i$ is similar to position $j$, it may be similar to positions around $j$.

**MLP Position Generator** For each absolute position $i$, we deploy a two-layer MLP to learn non-linear transformations from typology vector spaces to positional spaces. Specifically, the $i$-th position vector $\boldsymbol{a}_i^{(l)}$ is obtained by

$$\boldsymbol{a}_i^{(l)} = g\left(\boldsymbol{l}^{(l)} \cdot W_i^1 + \boldsymbol{b}_i^1\right) \cdot W_i^2 + \boldsymbol{b}_i^2, \quad (3)$$

where $g(\cdot)$ is a non-linear activation function and $W_i^1, W_i^2, \boldsymbol{b}_i^1$ and $\boldsymbol{b}_i^2$ are independent parameters for each position. Relative positions are generated in a similar way. Technically, for $\alpha_{ij}$ and $\boldsymbol{o}_i$ in Equation 2, we use two different position vectors $\boldsymbol{r}_i^{K,(l)}$, $\boldsymbol{r}_i^{V,(l)}$ generated by two MLPs.

**Self-attention Generator** The MLP generator learns position vectors only based on position index $i$ and typological vector $\boldsymbol{l}$. It is possible (Figure 3) that the learned vectors no longer contains the semantic of "position" (e.g., vectors of two close positions are more similar than vectors of two distant positions). Therefore, we also try to include prior knowledge on positions to regularize learned vectors. We build a new position generator based on multi-head self-attention layers.

For absolute position vectors, we assign one head of the self-attention layer for each position $i$. The typological vector $\boldsymbol{l}_l$ is considered as the query vector, and a set of prior position vectors $[\boldsymbol{c}_0, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_{127}]$ are key and value vectors. The absolute position representation $\boldsymbol{a}_i^{(l)}$ of $i$ is ob-

| Code | Description (Order of . . . ) | Value & Embeddings ($\in \mathbb{R}^3$) | | | |
|---|---|---|---|---|---|
| 81A | **S**ubject, **O**bject and **V**erb | SOV | SVO | VSO | Mixed |
| 85A | **A**dposition and **N**oun **P**hrase | NPA | ANP | Mixed | |
| 86A | **G**enitive and **N**oun | GN | NG | Mixed | |
| 87A | **A**djective and **N**oun | AN | NA | | |
| 88A | **D**emonstrative and **N**oun | DN | ND | DND | |
| 89A | **Num**eral and **N**oun | NumN | NNum | | |

| | Typological Features ($\in \mathbb{R}^{18}$) | | Typological Features ($\in \mathbb{R}^{18}$) |
|---|---|---|---|
| **Arabic** | VSO⊕ANP⊕NG⊕NA⊕DN⊕NumN | **Bulgarian** | SVO⊕ANP⊕Mixed⊕AN⊕DN ⊕NumN |

Table 1: Six word order typological features from WALS (above), and typological vectors $\boldsymbol{l}$ of Arabic and Bulgarian (below).

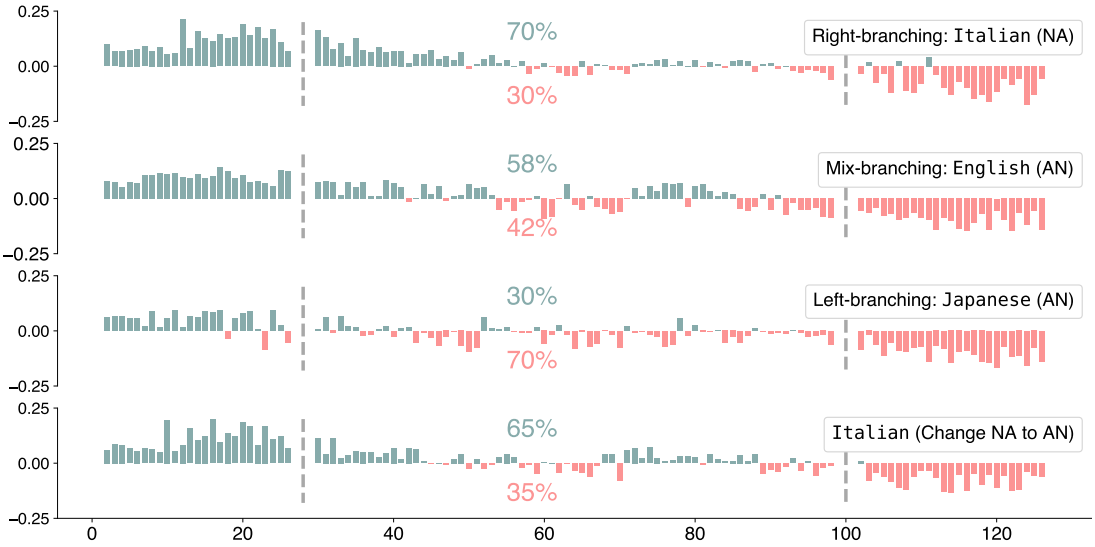


Figure 4: Attention patterns of position vectors. The x-aix is the 128 positions, and the y-aix describes differences of attention scores allocated to the right and to the left (positive values means shift right). The x-aix is divided into 3 regions, the front part contains the first 30 positions, the back part contains the last 30 positions, and the middle part contains the remaining 68 positions. Percentages report how many positions in middle parts shift left/right.

tained by weighted averaging over prior vectors.

$$\alpha_{ij}^{(l)} \propto \boldsymbol{l}^{(l)} \cdot U_i^Q \cdot \left(\boldsymbol{c}_j \cdot U_i^K\right)^\top,$$
$$\boldsymbol{a}_i^{(l)} = \sum_j \alpha_{ij}^{(l)} \cdot \left(\boldsymbol{c}_j \cdot U_i^V\right), \quad (4)$$

where $U_i^Q, U_i^K, U_i^V$ are the parameter matrices of position $i$ and are shared among all languages. The self-attention operation can be seen as a soft version of selecting a vector from an existing position vector set. In experiments, we set prior position vectors via sinusoidal functions (Equation 1). We can also build relative position vectors using the prior vectors.

Introducing prior vectors has another advantages regarding interpretability: they provide a coordinate system where we could compare the learned position spaces for different languages. In other words, for a newly learned position $i$, we can compare its attention patterns in two different languages. For example, if the learned absolute position vector $\boldsymbol{a}_i$ *shift left* (attending more on its left positions $[\boldsymbol{c}_{i-1}, \boldsymbol{c}_{i-2}, \ldots]$ than its right positions $[\boldsymbol{c}_{i+1}, \boldsymbol{c}_{i+2}, \ldots]$), this position feature may explicit guide the multilingual model to attend more on left contexts of $i$. We depict attention patterns of each position in Figure 4 and find that,

- for almost all languages, positions near the front end (0) and back end (127) always attend inwards: the front positions shift right, and the back positions shift left. Therefore, for short sentences, position vectors will always push the model to see the whole input, and for long sentences, they will suggest the model to replay the input at the end.
- for those middle positions, their attention patterns

correlate well with its language branching type: [1] for left-branching languages (i.e., head words follow their complements), they usually shift left, while for right-branching languages (i.e., head words proceed their complements) , they shift right [2].

- if we perturb the typological vector, the distribution of attention pattern can change accordingly. For example, on Italian when we freeze all its typological features but only change its noun-adjective order feature from NA to AN, 5% of its position vectors change from shifting right to shifting left.

Above observations may suggest that, guided by the typological features, position vectors are endowed with meaningful and language specific hidden structures, and these structures could be virtualized with the help of prior position vector bases.

**Training and Testing** During training, we sample one batch from 13 high-resource languages with equal probability, which increases the diversity of training for multilingual positional encoding. During testing, we first generate the corresponding positional encodings for the languages **at once** and then use them directly as position vectors in the parsing task, which means that our generative network has **almost no** additional computational cost during testing.

## 5 Experiments

**Dataset** Universal Dependencies (UD) is a framework for consistent annotation of grammar (parts of speech and syntactic dependencies) across different human languages (Zeman et al., 2018). Following Kulmizev et al. (2019); Üstün et al. (2020), We choose 13 representative training languages (high-resource) and 30 testing languages (zero-resource). Statistics for the treebanks are listed in the supplemental material. The crosslingual word representations are derived from mBERT (Devlin et al., 2019). Since the mBERT representation is subword-level, we follow previous work in taking the first subword as word-level representation.

[1] https://en.wikipedia.ahau.cf/wiki/Branching_(linguistics)

[2] It is interesting to see that the attention pattern of position vectors are different with the supervision signal from the parsing task: position vectors require a head position attends its complements' positions, while the parsing signal requires a dependent attends its head. The disagreement may suggests that the learning of position vectors does not always follow the parsing task's inductive bias. We leave a deeper discussion of this observation in future work.

**Evaluation** Parsing performance is measured with labeled attachment scores (LAS). We use the official evaluation scripts provided in the CoNLL 2018 shared tasks (Zeman et al., 2018). All of our results are averaged over three runs.

**Supplemental Material** Full results for the 30 zero-shot languages (A), experimental details (including hyperparameters, training time, model size (B)), more visualizations of position representations (C), and dataset statistics (D) are placed in the supplementary material.

### 5.1 Main Results

**Baselines** We denote $\mathtt{T}_{\mathtt{abs}}$, $\mathtt{T}_{\mathtt{rel}}$ to represent Transformer with absolute position and relative position respectively, and denote the two position generation methods as MLP and ATTN. We conduct experiments with following six baseline methods,

- udpipe (Straka, 2018), a monolingually trained multi-task parser;
- uuparser (Kulmizev et al., 2019), a monolingually trained BiLSTM parser using mBERT as additional crosslingual feature;
- udify (Kondratyuk and Straka, 2019), an all parameters fine-tuned mBERT parser could be trained both monolingually and multilingually;
- udapter (Üstün et al., 2020), a multilingually trained parser which only fine-tunes additional adapter parameters in mBERT;
- ID, it assigns each language a vector (which will be learned from scratch), and the vectors are added directly to the universal position vectors (Östling and Tiedemann, 2017);
- Feat, it direclty adds typology feature vectors (constructed in Section 4) to the universal position vectors (Scholivet et al., 2019).

**Results** We trained parsers monolingually (*one model per language*) and multilingually (*one model for all languages*) respectively (Table 2).

For the udify model which doesn't include any linguistic prior, its multilingual version underperforms its monolingual version on high-resouces, which witnesses a negative transfer. On the other side, the four methods (ID, Feat, MLP, ATTN) adding typological prior (URIEL) can reduce the gap to the best monolingual result, where language ID embeddings (ID) has the least effect, next to typological features (Feat), and our two proposed position generation methods are more effective. In particular, the ATTN method significantly improves

| | ar | en | eu | fi | he | hi | it | ja | ko | ru | sv | tr | zh | $\overline{\text{HR}}$ | $\overline{\text{ZR}}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Monolingual *(one model per language):* | | | | | | | | | | | | | | | |
| uuparser | 81.8 | 87.6 | 79.8 | 83.9 | 85.9 | 90.8 | 91.7 | 92.1 | 84.2 | 91.0 | 86.9 | 64.9 | 83.4 | 84.9 | - |
| udpipe | 82.9 | 87.0 | 82.9 | 87.5 | 86.9 | 91.8 | 91.5 | **93.7** | 84.2 | 92.3 | 86.6 | 67.6 | 80.5 | 85.8 | - |
| udify | 83.5 | 89.4 | 81.3 | 87.3 | 87.9 | 91.1 | 93.1 | 92.5 | 84.2 | 91.9 | 88.0 | 66.0 | 82.4 | 86.0 | - |
| $\text{T}_{\text{abs}}$ | 83.5 | 89.8 | 81.6 | 87.1 | 87.8 | 91.3 | 93.1 | 92.4 | 84.0 | 92.4 | 88.0 | 66.1 | 82.9 | 86.2 | - |
| $\text{T}_{\text{rel}}$ | 83.4 | 89.8 | 81.5 | 87.2 | 87.7 | 91.2 | 93.2 | 92.3 | 84.0 | 92.2 | 88.1 | 65.7 | 82.7 | 86.1 | - |
| Multilingually *(one model for all languages):* | | | | | | | | | | | | | | | |
| udify | 80.1 | 88.5 | 76.4 | 85.1 | 84.4 | 89.3 | 92.0 | 90.0 | 78.0 | 89.0 | 86.2 | 62.9 | 77.8 | 83.0 | 35.3 |
| udapter | 84.4 | 89.7 | 83.3 | 89.0 | 88.8 | 92.0 | **93.5** | 92.8 | 85.9 | 92.2 | **90.3** | 69.6 | 83.2 | 87.3 | 36.5 |
| $\text{T}_{\text{abs}}$ ID | 80.4 | 88.8 | 76.6 | 85.4 | 84.8 | 89.4 | 92.5 | 90.4 | 78.2 | 89.2 | 86.4 | 63.2 | 78.0 | 83.3 | - |
| $\text{T}_{\text{abs}}$ Feat | 80.5 | 88.8 | 76.3 | 85.3 | 84.7 | 89.5 | 92.5 | 90.4 | 79.2 | 89.3 | 86.5 | 62.9 | 78.1 | 83.4 | 35.8 |
| $\text{T}_{\text{abs}}$ MLP | 84.0 | 89.8 | 82.3 | 88.4 | 88.7 | 90.5 | 92.9 | 91.3 | 82.6 | 90.5 | 89.7 | 64.9 | 79.7 | 85.8 | 36.4 |
| $\text{T}_{\text{abs}}$ ATTN | 84.3 | **90.1** | 83.2 | 89.4 | 88.9 | 92.5 | 93.2 | 93.0 | 86.0 | **92.5** | 89.9 | 69.9 | 82.7 | 87.4 | 37.0 |
| $\text{T}_{\text{rel}}$ ID | 80.2 | 88.7 | 76.4 | 85.3 | 84.8 | 89.5 | 92.4 | 90.2 | 78.3 | 89.2 | 86.4 | 62.9 | 77.9 | 83.2 | - |
| $\text{T}_{\text{rel}}$ Feat | 81.0 | 88.2 | 77.3 | 85.6 | 85.4 | 89.6 | 92.3 | 90.5 | 80.1 | 89.4 | 86.5 | 63.5 | 79.1 | 83.7 | 35.9 |
| $\text{T}_{\text{rel}}$ MLP | 83.9 | 88.7 | 82.3 | 88.1 | 88.0 | 90.6 | 92.8 | 91.2 | 84.4 | 91.5 | 89.8 | 67.9 | 82.7 | 86.5 | 36.7 |
| $\text{T}_{\text{rel}}$ ATTN | **85.3** | 90.0 | **83.8** | **89.6** | **89.3** | **92.7** | 93.4 | 93.0 | **86.4** | **92.5** | 90.2 | **70.6** | **83.9** | **87.8** | **37.2** |

Table 2: Multilingual parsing performances. Last two columns show average LAS of 13 high-resource ($\overline{\text{HR}}$) and 30 zero-resource ($\overline{\text{ZR}}$) languages respectively.

the performance of the multilingual parser, boosting 4.0 LAS with $\text{T}_{\text{abs}}$ and 4.1 LAS on $\text{T}_{\text{rel}}$ (comparing with Feat). It also outperforms monolingual training by 1.2 LAS and 1.6 LAS. By simply adding the prior to all position vectors (approximates to a bias term), ID and Feat can hardly control the learning of the single prior parameter, so their performance gain is marginal. ATTN always outperforms MLP. It suggests that keeping a correct semantic of "position" could be crucial for learning an effective position space.

A major advantage of introducing language specific information in multilingual training is the ability to parse languages that have not been seen during training. On the 30 widely selected zero-resource languages (a subset is in Table 3), all methods except the ID method improve performances. The ATTN method still achieves the highest zero-resources parsing scores, which could be the effect of both effective way of encoding typological features (self-attention) and using a proper position prior.

The current best parser udapter is based on adapter fine-tuning. Similar to MLP, udapter uses a multi-layer perceptron to generate adapter parameters from generic typological information (URIEL). Unlike udapter, our methods focus on guiding the position vectors, which account for a smaller number of parameters in the parser. Comparing with udapter, ATTN leads 11 out of 13 high-resource languages, while for zero-resources, it further improves 0.5 LAS.

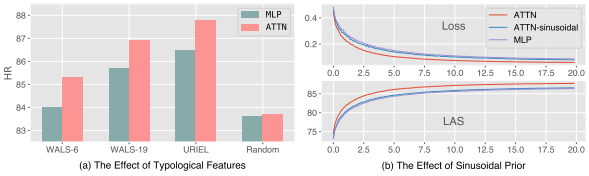


Figure 5: (a) The average LAS of 13 high-resource languages for the four different typological features. (b) The value of the loss function during training as well as the performance on the dev set.

These results suggest that explicitly associating typological information with the learning of position vectors makes a better use of typological information.

## 5.2 Analyses

**The Effect of Typological Features** To analyse the effect of typological features, we conducted experiments on four feature settings with two position generation methods (Figure 5(a)):

- WALS-6: 6 word order features from WALS (Naseem et al., 2012);
- WALS-19: 19 word order features from WALS (Scholivet et al., 2019);
- URIEL: 103 word order features from URIEL typology database (Littell et al., 2017);
- Random: random noise values instead of URIEL features.

Both MLP and ATTN methods outperform the baseline (Feat) on three meaningful sets, indicating that our method is applicable to a wide range

| | In mBERT | | | | | | | | | Out of mBERT | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | be | cy | kk | mr | ta | te | tl | yo | $\overline{\text{ZR}}$ | aii | bxr | hsb | kmr | pcm | yue | $\overline{\text{ZR}}$ |
| udify | 79.3 | **54.4** | 60.7 | 44.4 | 46.1 | 71.1 | **69.5** | 42.7 | 57.9 | 8.4 | 26.1 | 53.2 | 11.2 | 36.1 | 30.5 | 27.6 |
| udapter | 80.1 | 53.6 | 61.9 | 46.4 | 46.0 | 71.2 | 62.7 | 41.2 | 58.5 | **14.3** | **28.9** | 54.2 | **12.1** | 36.7 | 32.8 | 29.8 |
| $\mathtt{T}_{\mathtt{abs}}$ MLP | 79.6 | 53.7 | 61.6 | 46.5 | 46.3 | 71.3 | 63.4 | 41.7 | 58.0 | 12.6 | 28.4 | 53.9 | 11.7 | 36.3 | 32.5 | 29.2 |
| $\mathtt{T}_{\mathtt{abs}}$ ATTN | 80.5 | 54.2 | 62.1 | **46.6** | 46.3 | 71.3 | 64.2 | 42.7 | 58.5 | 13.8 | 28.8 | 54.3 | 12.0 | 36.7 | 32.9 | 29.8 |
| $\mathtt{T}_{\mathtt{rel}}$ MLP | 80.0 | 54.2 | 61.5 | 46.5 | 46.2 | 71.3 | 63.7 | 41.9 | 58.2 | 13.5 | 28.5 | 53.8 | 11.8 | 36.4 | 32.6 | 29.4 |
| $\mathtt{T}_{\mathtt{rel}}$ ATTN | **80.7** | **54.4** | **62.2** | 46.3 | **46.5** | **71.6** | 66.0 | **42.9** | **58.8** | 14.1 | **28.9** | **54.4** | 11.8 | **36.9** | **33.2** | **29.9** |

Table 3: We select a subset of the zero-resource languages for demonstration. Eight languages are in the pre-training process of mBERT, six languages are not (complete zero-shot).

of word order typological features. Furthermore, we can observe that the largest boost from URIEL because it has the richest typological features.

Since our method adds additional network parameters (from MLP or positional attention networks), it may make comparisons unfair. So we conduct parameter size fairness experiments by modifying the values of URIEL to *random* values. It means that our generative networks are guided by nonsensical information. The results show that even though we retain the additional parameters, the random feature values severely hurt the gain from the generative networks. Therefore, our models' performance improvments are not due to the additional parameters.

**The Effect of Sinusoidal Priors** In the self-attention position generator (Equation 4), we introduce sinusoidal prior vector $\boldsymbol{c}_t$. As the sinusoidal prior describes some properties of positions, it can avoid the generated positions deviating too far from the basic position space (Figure 3). It might also be able to speed up convergence and improving the model's inductive bias. Figure 5 (b) compares loss function curves and LAS curves on the validation set of three models, ATTN, ATTN-sinusoidal, and MLP. ATTN-sinusoidal means replacing the sinusoidal priors with random initialized learnable vectors. The results show that ATTN-sinusoidal degenerates to be comparable to MLP. This demonstrates that the sinusoidal priors not only converge faster, but also help ATTN ending up with higher performances.

## 6 Related Work

**Multilingual Parsing** Dong et al. (2015); Johnson et al. (2017) identify the (positive) transfer - (negative) interference trade-off problem in multilingual neural machine translation. Early multilingual dependency parsing studies consider word representation as a negative transfer factor and learn delexicalized parsers (McDonald et al., 2013; Naseem et al., 2012; Duong et al., 2015). Although they avoid negative transfer, valuable lexical information was lost. As a result of the development of multilingual word representations, Ammar et al. (2016); Straka (2018) train multilingual parsers using multilingual word embeddings. Kondratyuk and Straka (2019); Üstün et al. (2020) train multilingual parsers using multilingual pretrained representations (mBERT (Devlin et al., 2019)). Once word representation became positive factor, recent studies found that word order became a new negative factor. Ahmad et al. (2019); Ji et al. (2021) observe the negative transfer phenomenon of word order in a zero-shot cross-lingual scenario. Previous work simply consider word order features as input (Östling and Tiedemann, 2017; Scholivet et al., 2019; Üstün et al., 2020). Instead, we explicitly associate it with order-related parameters (i.e., position representations) in the Transformer network.

## 7 Conclusions

We studied the role of position spaces in multilingual learning. By comparing a univeral position space and language-specific position spaces, we showed the latter could either handle linguistic constraints of different language efficiently or provide a clear path for positive transfer in multilingual learning. We developped a self-attention based position space generator. We showed that by utilizing typological prior and existing position space prior, the multilingual dependency parser could enjoy positive transfer on both high-resource and zero-resource languages. One future work is to investigate whether the obtained position vectors could help other multilingual and monolingual tasks. It is also interesting to compare the position spaces induced from different multilingual tasks (supervised or unsupervised).

## Limitations

An obvious limitation is that our work relies on the typology features of languages. Some extremely rare languages might lack typology studies (its features are missing values in the WALS database). Our approach is limited for these languages. Another non-critical limitation is that the technical contribution of our work is limited. After detailed analyses of position vectors, our methods for generating position vectors are not that complex, but we believe that an effective method is not neccessarily complex, and designing experiments to reveal key properties of position features and their connection with linguistic knowledge could still make solid contributes to NLP community.

## Acknowledgments

The authors wish to thank all reviewers for their helpful comments and suggestions. The corresponding authors are Tao Ji and Yuanbin Wu. This research was (partially) supported by National Key R&D Program of China (2021YFC3340700) and NSFC(62076097).

## References

Wasi Ahmad, Zhisong Zhang, Xuezhe Ma, Eduard Hovy, Kai-Wei Chang, and Nanyun Peng. 2019. On difficulties of cross-lingual transfer with order differences: A case study on dependency parsing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2440–2452, Minneapolis, Minnesota. Association for Computational Linguistics.

Waleed Ammar, George Mulcaire, Miguel Ballesteros, Chris Dyer, and Noah A. Smith. 2016. Many languages, one parser. *Trans. Assoc. Comput. Linguistics*, 4:431–444.

Johannes Bjerva, Robert Östling, Maria Han Veiga, Jörg Tiedemann, and Isabelle Augenstein. 2019. What do language representations really represent? *Comput. Linguistics*, 45(2):381–389.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 8440–8451. Association for Computational Linguistics.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G. Carbonell, Quoc Viet Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 2978–2988. 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, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Daxiang Dong, Hua Wu, Wei He, Dianhai Yu, and Haifeng Wang. 2015. Multi-task learning for multiple language translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 1723–1732. The Association for Computer Linguistics.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Matthew S. Dryer and Martin Haspelmath, editors. 2013. *WALS Online*. Max Planck Institute for Evolutionary Anthropology, Leipzig.

Long Duong, Trevor Cohn, Steven Bird, and Paul Cook. 2015. A neural network model for low-resource universal dependency parsing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 339–348. The Association for Computational Linguistics.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1243–1252. PMLR.

Harald Hammarström, Robert Forkel, Martin Haspelmath, and Sebastian Bank. 2021. *Glottolog 4.4*. Leipzig.

Tao Ji, Yong Jiang, Tao Wang, Zhongqiang Huang, Fei Huang, Yuanbin Wu, and Xiaoling Wang. 2021.

Word reordering for zero-shot cross-lingual structured prediction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4109–4120, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda B. Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Trans. Assoc. Comput. Linguistics*, 5:339–351.

Daniel Kondratyuk and Milan Straka. 2019. 75 languages, 1 model: Parsing universal dependencies universally. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2779–2795. Association for Computational Linguistics.

Artur Kulmizev, Miryam de Lhoneux, Johannes Gontrum, Elena Fano, and Joakim Nivre. 2019. Deep contextualized word embeddings in transition-based and graph-based dependency parsing - a tale of two parsers revisited. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2755–2768, Hong Kong, China. Association for Computational Linguistics.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2017. Unsupervised machine translation using monolingual corpora only. *arXiv preprint arXiv:1711.00043*.

Patrick Littell, David R. Mortensen, Ke Lin, Katherine Kairis, Carlisle Turner, and Lori S. Levin. 2017. URIEL and lang2vec: Representing languages as typological, geographical, and phylogenetic vectors. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 2: Short Papers*, pages 8–14. Association for Computational Linguistics.

Ryan T. McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith B. 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 51st Annual Meeting of the Association for Computational Linguistics, ACL 2013, 4-9 August 2013, Sofia, Bulgaria, Volume 2: Short Papers*, pages 92–97. The Association for Computer Linguistics.

Tahira Naseem, Regina Barzilay, and Amir Globerson. 2012. Selective sharing for multilingual dependency parsing. In *The 50th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, July 8-14, 2012, Jeju Island, Korea - Volume 1: Long Papers*, pages 629–637. The Association for Computer Linguistics.

Robert Östling and Jörg Tiedemann. 2017. Continuous multilinguality with language vectors. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 2: Short Papers*, pages 644–649. Association for Computational Linguistics.

Manon Scholivet, Franck Dary, Alexis Nasr, Benoît Favre, and Carlos Ramisch. 2019. Typological features for multilingual delexicalised dependency parsing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 3919–3930. Association for Computational Linguistics.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers)*, pages 464–468. Association for Computational Linguistics.

Moran Steven, McCloy Daniel, and Wright Richard, editors. 2014. *PHOIBLE Online*. Max Planck Institute for Evolutionary Anthropology, Leipzig.

Milan Straka. 2018. Udpipe 2.0 prototype at conll 2018 UD shared task. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies, Brussels, Belgium, October 31 - November 1, 2018*, pages 197–207. Association for Computational Linguistics.

Ahmet Üstün, Arianna Bisazza, Gosse Bouma, and Gertjan van Noord. 2020. Udapter: Language adaptation for truly universal dependency parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 2302–2315. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Benyou Wang, Donghao Zhao, Christina Lioma, Qiuchi Li, Peng Zhang, and Jakob Grue Simonsen. 2020. Encoding word order in complex embeddings. In

*8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Daniel Zeman, Jan Hajic, Martin Popel, Martin Potthast, Milan Straka, Filip Ginter, Joakim Nivre, and Slav Petrov. 2018. Conll 2018 shared task: Multilingual parsing from raw text to universal dependencies. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies, Brussels, Belgium, October 31 - November 1, 2018*, pages 1–21. Association for Computational Linguistics.

Yuan Zhang and Regina Barzilay. 2015. Hierarchical low-rank tensors for multilingual transfer parsing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1857–1867. The Association for Computational Linguistics.

**Supplementary Material for**
***Typology Guided Multilingual Position Representations: Case on Dependency Parsing***

## A Zero-Shot Results

Table 4 shows LAS scores on all 30 zero-resouce languages for two types of Transformer guided by three typological methods, respectively, UDapter (Üstün et al., 2020), and udify (Kondratyuk and Straka, 2019). Languages with "*" are not present in mBERT training data. Overall, our ATTN approach achieves state-of-the-art performance, especially for the Transformer$_{abs}$ model. Our MLP approach also has visible improvements and is able to compete with udapter. This suggests that position representation guided by typological features can be successfully transferred to unseen zero-shot languages. In addition, we specifically look at languages that are not in the mBERT training set, which implies that the cross-lingual word representations are not well aligned. The performance of these languages is almost always unacceptably low. This suggests that multilingual word representations are the foundation of multilingual position representation.

## B Experimental Details

**Implementation** Our parser's implementation is based on the framework proposed by Kulmizev et al. (2019). The only difference is that we have replaced their BiLSTM context encoder with the recently popular Transformer context encoder. This is because the position representations we focus on are an important part of Transformer encoder. The hyperparameters of the parser classifier are identical to those of Udapter (Üstün et al., 2020) without applying a new hyperparameter search. Unlike Udapter, which fixes the mBERT and trains the extra added adapter modules in it, we train the extra Transformer context encoder with multilingual position encoding on top of the fixed mBERT. Together with the additional Transformer context encoder and multilingual position generation networks that are picked manually by average high-resource parsing LAS, hyper-parameters are summarized in Table 5.

**Training Time and Model size** In terms of training time, on the NVIDIA RTX3090 GPU, our parser takes about 15 minutes for one epoch over the full training set. Comparing different multilingual position generation methods, they have a similar training time. In terms of number of *trainable* parameters, our ATTN has 63.0M (24.5M for position generation, 30.7M for Transformer encoder, 7.8M for classifier) total number of parameters, and MLP has 69.3M (30.8M for position generation). As a comparison, UDify uses 191M parameters, UDapter uses 550M parameters in total (302M for adapters and 248M for classifier), and monolingual UDify models use 2.5B parameters (13x191M). The number of training parameters in our method is much smaller than the baseline parsers. In addition, we can pre-generate multilingual positions during the inference phase and the model parameters can be reduced to 38.5M, which is only 7% of the UDapter.

## C Additional Visualization

Figure 6 shows a visualization of the position vectors for three languages containing English, Italian and Chinese. We can see that they do not exhibit loose similarity patterns and still keep some properties of the sinusoidal position prior (e.g. symmetry). And these patterns clearly differ between languages. This further suggests that it is not reasonable to use the same position representation for different languages. It is necessary to guide multilingual position representations by appropriate methods (e.g. our word order features).

## D Dataset

The statistics (including treebank name, word order, language family and number of sentences) of Universal Dependency (UD) treebanks are summarized in Table 6 and Table 7. The 13 high-resource languages from UD v2.3 and 30 zero-resource languages from UD v2.5 are consistent with the selection made by Kulmizev et al. (2019); Üstün et al. (2020). The dataset UD v2.3 can be freely download from `http://hdl.handle.net/11234/1-2895`, and v2.5 can be download from `http://hdl.handle.net/11234/1-3105`.

**Typological Features** : We conducted experiments on three feature settings including

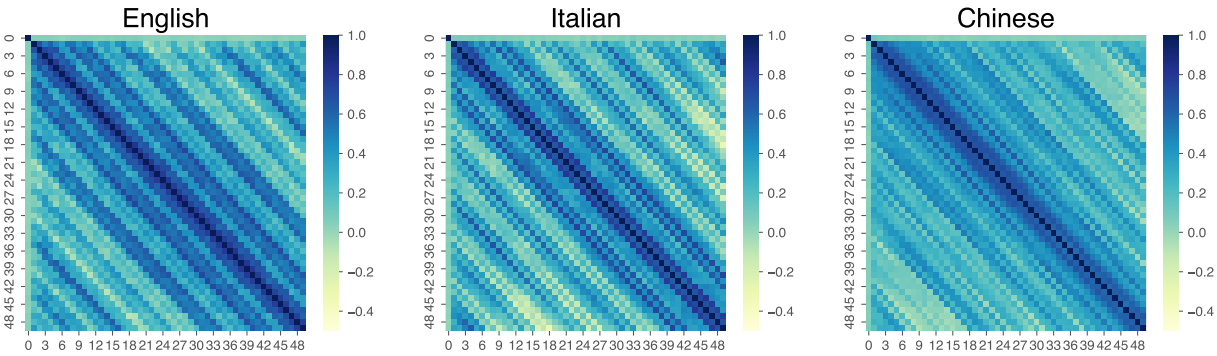


Figure 6: Cosine similarity of the first 50 position vectors. We can see that vectors learned from English, Italian and Chinese keep some property of the sinusoidal position prior (e.g., symmetry). We can also observe that the three languages have distinctly different patterns.

- WALS-6: codes 81A, 85A, 86A, 87A, 88A, 89A from WALS;
- WALS-19: codes 81A, 82A, 83A, 85A, 86A, 87A, 88A, 89A, 90A, 92A, 94A, 95A, 96A, 97A, 144A, 143A, 143E, 143F, 143G from WALS;
- URIEL: 103 word order features from URIEL typology database.

Where WALS-6 and WALS-19 can be freely download from https://wals.info, while URIEL can be freely download from https://github.com/antonisa/lang2vec.

| Code | udify | udapter | $\mathtt{T}_{\mathtt{abs}}$ MLP | $\mathtt{T}_{\mathtt{abs}}$ ATTN | $\mathtt{T}_{\mathtt{rel}}$ MLP | $\mathtt{T}_{\mathtt{rel}}$ ATTN |
|---|---|---|---|---|---|---|
| aii* | 8.4 | **14.3** | 12.6 | 13.8 | 13.5 | 14.1 |
| akk* | 4.5 | 8.2 | 8.9 | 9.4 | 9.0 | **9.8** |
| am* | 2.8 | 5.9 | 6.5 | 6.8 | 6.5 | **7.1** |
| be | 80.1 | 79.3 | 79.6 | 80.5 | 80.0 | **80.7** |
| bho* | 37.2 | 37.3 | 37.9 | 38.3 | 38.2 | **38.5** |
| bm* | 8.9 | 8.1 | 8.3 | 8.9 | 8.4 | **9.3** |
| br* | 60.5 | 58.5 | 59.3 | **60.8** | 60.4 | 60.7 |
| bxr* | 26.1 | **28.9** | 28.4 | 28.8 | 28.5 | **28.9** |
| cy | 53.6 | **54.4** | 53.7 | 54.2 | 54.2 | **54.4** |
| fo* | 68.6 | 69.2 | 69.8 | **69.9** | **69.9** | 69.7 |
| gsw* | 43.6 | 45.5 | 45.1 | 45.8 | 45.3 | **45.9** |
| gun* | 8.5 | 8.4 | 9.3 | 9.5 | 9.2 | **9.7** |
| hsb* | 53.2 | 54.2 | 53.9 | 54.3 | 53.8 | **54.4** |
| kk | 61.9 | 60.7 | 61.6 | 62.1 | 61.5 | **62.2** |
| kmr* | 11.2 | **12.1** | 11.7 | 12.0 | 11.8 | 11.8 |
| koi* | 20.8 | 23.1 | 22.8 | 23.5 | 23.5 | **23.7** |
| kpv* | 12.4 | 12.5 | 12.4 | 12.9 | 12.6 | **13.2** |
| krl* | 49.2 | 48.4 | 48.5 | 49.4 | 48.9 | **49.6** |
| mdf* | 24.7 | 26.6 | 26.9 | **27.3** | 27.1 | 27.1 |
| mr | 46.4 | 44.4 | 46.5 | **46.6** | 46.5 | 46.3 |
| myv* | 19.1 | 19.2 | 18.9 | 19.7 | 19.4 | **19.9** |
| olo* | 42.1 | 43.3 | 43.1 | 43.7 | 43.6 | **43.8** |
| pcm* | 36.1 | 36.7 | 36.3 | 36.7 | 36.4 | **36.9** |
| sa* | 19.4 | 22.2 | 22.5 | 22.9 | 22.6 | **23.2** |
| ta | 46.0 | 46.1 | 46.3 | 46.3 | 46.2 | **46.5** |
| te | 71.2 | 71.1 | 71.3 | 71.3 | 71.3 | **71.6** |
| tl | 62.7 | **69.5** | 63.4 | 64.2 | 63.7 | 66.0 |
| wbp* | 9.6 | 12.1 | 11.9 | 12.9 | 12.5 | **13.3** |
| yo | 41.2 | 42.7 | 41.7 | 42.7 | **41.9** | **42.9** |
| yue* | 30.5 | 32.8 | 32.5 | 32.9 | **32.6** | **33.2** |
| $\overline{\mathrm{ZR}}$ | 35.3 | 36.5 | 36.4 | 37.0 | 36.7 | **37.2** |

Table 4: Multilingual parsing performance on 30 zero-resource languages respectively. The "*" marker shows languages not present in mBERT training data. The last row ($\overline{\mathrm{ZR}}$) shows average LAS of each method.

| Layer | Hyper-parameter | Value |
|---|---|---|
| Input | Fixed mBERT | 768 |
| Transformer | Layer | 4 |
| | Hidden | 768 |
| | Head | 12 |
| | Dropout | 0.2 |
| Position range | Absolute | [0, 128) |
| | Relative | [-4, 4] |
| MLP with WALS-6 | $D_{in} \rightarrow D_{hid} \rightarrow D_{out}$ | 18→256→768 |
| MLP with WALS-19 | $D_{in} \rightarrow D_{hid} \rightarrow D_{out}$ | 57→256→768 |
| MLP with URIEL | $D_{in} \rightarrow D_{hid} \rightarrow D_{out}$ | 103→256→768 |
| ATTN with WALS-6 | $W^Q, W^K, W^V$ | $\mathbb{R}^{18\times768}, \mathbb{R}^{768\times768}, \mathbb{R}^{768\times768}$ |
| ATTN with WALS-19 | $W^Q, W^K, W^V$ | $\mathbb{R}^{57\times768}, \mathbb{R}^{768\times768}, \mathbb{R}^{768\times768}$ |
| ATTN with URIEL | $W^Q, W^K, W^V$ | $\mathbb{R}^{103\times768}, \mathbb{R}^{768\times768}, \mathbb{R}^{768\times768}$ |
| Classifier | Dependency tag dimension | 256 |
| | Dependency arc dimension | 768 |
| | Label smoothing | 0.03 |
| Trainer | Optimizer | Adam |
| | Learning rate | 1e-3 |
| | $(\beta_1, \beta_2)$ | (0.9, 0.98) |
| | Batch size | 80 |
| | Epochs | 80 |

Table 5: Hyper-parameters for our parser.

| Language | Code | Treebank | Family | {S,V,O} | {N,A} | #Train | #Test |
|---|---|---|---|---|---|---|---|
| Arabic | ar | PADT | Afro-Asiatic, Semitic | VSO | NA | 6.1k | 680 |
| Basque | eu | BDT | Basque | SOV | NA | 5.4k | 1799 |
| Chinese | zh | GSD | Sino-Tibetan | SVO | AN | 4.0k | 500 |
| English | en | EWT | IE, Germanic | SVO | AN | 12.5k | 2077 |
| Finnish | fi | TDT | Uralic, Finnic | SVO | AN | 12.2k | 1555 |
| Hebrew | he | HTB | Afro-Asiatic, Semitic | SVO | NA | 5.2k | 491 |
| Hindi | hi | HDTB | IE, Indic | SOV | AN | 13.3k | 1684 |
| Italian | it | ISDT | IE, Romance | SVO | NA | 13.1k | 482 |
| Japanese | ja | GSD | Japanese | SOV | AN | 7.1k | 551 |
| Korean | ko | GSD | Korean | SOV | AN | 4.4k | 989 |
| Russian | ru | SynTagRus | IE, Slavic | SVO | AN | 15k | 6491 |
| Swedish | sv | Talbanken | IE, Germanic | SVO | AN | 4.3k | 1219 |
| Turkish | tr | IMST | Turkic, Southwestern | SOV | AN | 3.7k | 975 |

Table 6: Statistics of the high-resource languages from UD v2.3. We chose the same treebank as Kulmizev et al. (2019); Üstün et al. (2020).

| Language | Code | Treebank | Family | #Test |
|---|---|---|---|---|
| Akkadian | akk | PISANDUB | Afro-Asiatic, Semitic | 1074 |
| Amharic | am | ATT | Afro-Asiatic, Semitic | 101 |
| Assyrian | aii | AS | Afro-Asiatic, Semitic | 57 |
| Bambara | bm | CRB | Mande | 1026 |
| Belarusian | be | HSE | IE, Slavic | 253 |
| Bhojpuri | bho | BHTB | IE, Indic | 254 |
| Breton | br | KEB | IE, Celtic | 888 |
| Buryat | bxr | BDT | Mongolic | 908 |
| Cantonese | yue | HK | Sino-Tibetan | 1004 |
| Erzya | myv | JR | Uralic, Mordvin | 1550 |
| Faroese | fo | OFT | IE, Germanic | 1207 |
| Karelian | krl | KKPP | Uralic, Finnic | 228 |
| Kazakh | kk | KTB | Turkic, Northwestern | 1047 |
| Komi Permyak | koi | UH | Uralic, Permic | 49 |
| Komi Zyrian | kpv | LATTICE, IKDP | Uralic, Permic | 210 |
| Kurmanji | kmr | MG | IE, Iranian | 734 |
| Livvi | olo | KKPP | Uralic, Finnic | 106 |
| Marathi | mr | UFAL | IE, Indic | 47 |
| Mbya Guarani | gun | THOMAS, DOOLEY | Tupian | 98 |
| Moksha | mdf | JR | Uralic, Mordvin | 21 |
| Naija | pcm | NSC | Creole | 948 |
| Sanskrit | sa | UFAL | IE, Indic | 230 |
| Swiss G. | gsw | UZH | IE, Germanic | 100 |
| Tagalog | tl | TRG | Austronesian, Central Philippine | 55 |
| Tamil | ta | TTB | Dravidian, Southern | 120 |
| Telugu | te | MTG | Dravidian, South Central | 146 |
| Upper Sorbian | hsb | UFAL | IE, Slavic | 623 |
| Warlpiri | wbp | UFAL | Pama-Nyungan | 54 |
| Welsh | cy | CCG | IE, Celtic | 956 |
| Yoruba | yo | YTB | Niger-Congo, Defoid | 100 |

Table 7: Statistics of the zero-resource languages from UD v2.5. We chose the same treebank as Kulmizev et al. (2019); Üstün et al. (2020).

## ACL 2023 Responsible NLP Checklist

### A For every submission:

☑ A1. Did you describe the limitations of your work?
*Line 557-571*

☑ A2. Did you discuss any potential risks of your work?
*Line 558-562*

☑ A3. Do the abstract and introduction summarize the paper's main claims?
*Line 006-012*

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

### B ☒ Did you use or create scientific artifacts?

*Left blank.*

☐ B1. Did you cite the creators of artifacts you used?
*No response.*

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
*No response.*

☐ 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)?
*No response.*

☐ 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?
*No response.*

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
*No response.*

☐ 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.
*No response.*

### C ☑ Did you run computational experiments?

*Line 829-848*

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
*Line 829-848*

*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?
*Line 829-848*

☑ 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?
*Line 829-848*

☑ 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.)?
*Line 829-848*

**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.?
*No response.*

☐ 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)?
*No response.*

☐ 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?
*No response.*

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
*No response.*

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
*No response.*