Bridging Pre-trained Language Models and Hand-crafted Features for Unsupervised POS Tagging

Houquan Zhou, Yang Li^{*}, Zhenghua Li[∞], Min Zhang

Institute of Artificial Intelligence, School of Computer Science and Technology, Soochow University, China

{hqzhou,ylinlp}@stu.suda.edu.cn; {zhli13,minzhang}@suda.edu.cn

Abstract

In recent years, large-scale pre-trained language models (PLMs) have made extraordinary progress in most NLP tasks. But, in the unsupervised POS tagging task, works utilizing PLMs are few and fail to achieve stateof-the-art (SOTA) performance. The recent SOTA performance is yielded by a Guassian HMM variant proposed by He et al. (2018). However, as a generative model, HMM makes very strong independence assumptions, making it very challenging to incorporate contexualized word representations from PLMs. In this work, we for the first time propose a neural conditional random field autoencoder (CRF-AE) model for unsupervised POS tagging. The discriminative encoder of CRF-AE can straightforwardly incorporate PLM word representations. Moreover, inspired by featurerich HMM, we reintroduce hand-crafted features into the decoder of CRF-AE. Finally, experiments clearly show that our model outperforms previous state-of-the-art models by a large margin on Penn Treebank and multilingual Universal Dependencies treebank v2.0.

1 Introduction

Unsupervised learning has been an important yet challenging research direction in NLP (Klein and Manning, 2004; Liang et al., 2006; Seginer, 2007). Training models directly from unlabeled data can relieve painful data annotation and is thus especially attractive for low-resource languages (He et al., 2018). As three typical tasks related to syntactic analysis, unsupervised part-of-speech (POS) tagging (or induction), dependency parsing, and constituency parsing have attracted intensive interest during the past three decades (Pereira and Schabes, 1992; Christodoulopoulos et al., 2010, *inter alia*). Compared with tree-structure dependency and constituency parsing, POS tagging corresponds

PRP	VBD	IN	PRP\$	NN	
Ι	looked	at	my	watch	.)

Figure 1: Example of POS tagging.

to simpler sequential structure, and aims to assign a POS tag to each word, as depicted in Figure 1. Besides the alleviation of labeled data, unsupervised POS tagging is particularly valuable for child language acquisition study because every child manages to induce syntactic categories without access to labeled data (Yuret et al., 2014).

Nowadays, supervised POS tagging models trained on large-scale labeled data can already achieve extremely high accuracy, for example over 97.5% on English Penn Treebank (PTB) texts (Huang et al., 2015; Bohnet et al., 2018; Zhou et al., 2020). However, unsupervised POS tagging, though having attracted a lot of research interest (Lin et al., 2015; Tran et al., 2016; He et al., 2018; Stratos, 2019; Gupta et al., 2020), can only achieve at most 80.8% many-to-one (M-1) accuracy, where M-1 means multiple induced tags can be mapped to a single ground-truth tag when evaluating the model on the test data.

The generative Hidden Markov Models (HMMs) are the most representative and successful approach for unsupervised POS tagging (Merialdo, 1994; Graça et al., 2009). By treating POS tags as latent variables, a first-order HMM factorizes the joint probability of a sentence and a tag sequence $p(\mathbf{x}, \mathbf{y})$ into independent emission probabilities $p(x_i | y_i)$ and transition probabilities $p(y_{i-1} | y_i)$. The training objective is to maximize the marginal probability $p(\mathbf{x})$, which can be solved by the EM algorithm or direct gradient descent (Salakhutdinov et al., 2003). Berg-Kirkpatrick et al. (2010) propose a feature-rich HMM (FHMM), which further parameterizes $p(x_i | y_i)$ with many hand-crafted

^{*} Houquan and Yang make equal contributions to this work. Zhenghua is the corresponding author.

morphological features, greatly boosting M-1 accuracy to 75.5 from 63.1 of the basic HMM.

In the DL era, researchers have paid a lot of attention to HMMs for unsupervised POS tagging. Lin et al. (2015) propose a Gaussian HMM (GHMM), where $p(x_i | y_i)$ corresponds to the probability of the pre-trained word embedding (fixed during training) of x_i against the Gaussian distribution of y_i . Tran et al. (2016) propose a neural HMM model (NHMM), where $p(x_i \mid y_i)$ and $p(y_{i-1} \mid y_i)$ are all computed via neural networks with POS tag and word embeddings as inputs. He et al. (2018) extend the Gaussian HMM of Lin et al. (2015) by introducing an invertible neural projection (INP) component for the pre-trained word embeddings, which has a similar effect of tuning word embeddings during training. Their INP Gaussian HMM (INP-GHMM) approach achieves state-of-the-art (SOTA) M-1 accuracy (80.8) on PTB so far.

The major weakness of HMMs is the strong independence assumption in emission probabilities $p(x_i | y_i)$, which directly hinders the use of contextualized word representations from powerful pre-trained language models (PLMs) such as ELMo/BERT (Peters et al., 2018; Devlin et al., 2019). It is a pity since PLMs are able to greatly boost performance of many NLP tasks.

In this work, we for the first time propose a neural conditional random field autoencoder (CRF-AE) model for unsupervised POS tagging, inspired by Ammar et al. (2014) who propose a non-neural CRF-AE model. In the discriminative encoder of CRF-AE, we straightforwardly incorporate ELMo word representations. Moreover, inspired by feature-rich HMM (Berg-Kirkpatrick et al., 2010), we reintroduce hand-crafted features into the decoder of CRF-AE. In summary, this work makes the following contributions:

- We for the first time propose a neural CRF-AE model for unsupervised POS tagging.
- We successfully bridge PLMs and hand-crafted features in our CRF-AE model.
- Our model achieves new SOTA M-1 accuracy of 83.21 on the 45-tag English PTB data and outperforms the previous best result by 2.41.
- After a few straightforward adjustments, our model achieves new SOTA M-1 accuracy on the 12-tag multilingual Universal Dependencies treebank v2.0 (UD), surpassing the previous best results by 4.97 on average.

We release our code at https://github.c



Figure 2: Illustration of CRF-AE.

om/Jacob-Zhou/FeatureCRFAE, including our re-implemented HMM and FHMM models.

2 Vanilla CRF-AE

In this work, we adopt the CRF-AE approach as our basic model for unsupervised POS tagging. The non-neural CRF-AE model is first proposed by Ammar et al. (2014) for unsupervised sequence labeling tasks, inspired by neural network autoencoders. Cai et al. (2017) also extend the idea to non-neural unsupervised dependency parsing. The basic idea is first producing latent structures, i.e., POS tag sequences, with a discriminative CRF over the observed sentence, and then reconstructing the original sentence given each latent structure. The two steps correspond to the encoder and the decoder respectively.

Training loss. We denote a sentence as $\mathbf{x} = x_1, x_2, \dots, x_i, \dots, x_n$, and a POS tag sequence as $\mathbf{y} = y_1, y_2, \dots, y_i, \dots, y_n$. Given an unlabeled dataset \mathcal{D} which does not contain any POS tag sequences, the training loss is:

$$\mathcal{L}(\mathcal{D}; \boldsymbol{\phi}, \boldsymbol{\theta}) = -\sum_{\mathbf{x} \in \mathcal{D}} \log \mathbb{E}_{\mathbf{y} \sim p(\mathbf{y} \mid \mathbf{x}; \boldsymbol{\phi})} p(\mathbf{x} \mid \mathbf{y}; \boldsymbol{\theta}) + \lambda \left(\|\boldsymbol{\phi}\|_2^2 + \|\boldsymbol{\theta}\|_2^2 \right), \quad (1)$$

where $p(\mathbf{y} | \mathbf{x}; \boldsymbol{\phi})$ is the CRF encoder; $p(\mathbf{x} | \mathbf{y}; \boldsymbol{\theta})$ is the decoder; $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ are model parameters.

This training loss encourages the model to meet the intuition that a high-probability POS sequence should also permit reconstruction of the sentence with a high probability.

Ammar et al. (2014) adopt the *Expectation-Maximization* (EM) algorithm for training. In this work, we directly compute the training loss via the *Forward* algorithm. Then, we employ the powerful *AutoGrad* function of deep learning to compute the gradient of each parameter. Our preliminary experiments on HMM and feature-rich HMM show that this gradient-based approach is consistently superior to EM in both efficiency and performance.

Inference. During evaluation, we follow Ammar et al. (2014) and use both the CRF and the reconstruction probabilities to obtain the optimal tag sequence:

$$\mathbf{y}^* = \arg \max_{\mathbf{y}} p(\mathbf{y} \mid \mathbf{x}; \boldsymbol{\phi}) p(\mathbf{x} \mid \mathbf{y}; \boldsymbol{\theta}), \quad (2)$$

which can be solved by the Viterbi algorithm.

CRF Encoder: $p(\mathbf{y} | \mathbf{x}; \boldsymbol{\phi})$. As a discriminative log-linear model, the CRF encoder defines a conditional probability:

$$p(\mathbf{y} \mid \mathbf{x}; \boldsymbol{\phi}) = \frac{\exp\left(S(\mathbf{x}, \mathbf{y}; \boldsymbol{\phi})\right)}{Z(\mathbf{x}; \boldsymbol{\phi}) \equiv \sum_{\mathbf{y}} \exp\left(S(\mathbf{x}, \mathbf{y}; \boldsymbol{\phi})\right)},$$
(3)

where $Z(\mathbf{x})$ is the partition function, also known as the normalization term.

The score of \mathbf{y} given \mathbf{x} is decomposed into bigram scores:

$$S(\mathbf{x}, \mathbf{y}; \boldsymbol{\phi}) = \sum_{i=1}^{n} s(\mathbf{x}, y_{i-1}, y_i; \boldsymbol{\phi}).$$
(4)

Ammar et al. (2014) use hand-crafted discrete features to obtain bigram scores.

$$s(\mathbf{x}, y_{i-1}, y_i; \boldsymbol{\phi}) = \boldsymbol{\phi}^{\mathsf{T}} g(\mathbf{x}, y_{i-1}, y_i, i).$$
 (5)

Decoder: $p(\mathbf{x} | \mathbf{y}; \boldsymbol{\theta})$. The decoder computes the reconstruction probability of \mathbf{x} given a POS tag sequence \mathbf{y} , which is factorized into position-wise generation probabilities based on a strong independence assumption.

$$p(\mathbf{x} \mid \mathbf{y}; \boldsymbol{\theta}) = \prod_{i=1}^{n} p(x_i \mid y_i; \boldsymbol{\theta}).$$
(6)

Ammar et al. (2014) use a categorical distribution matrix θ , which is updated via EM training, to maintain all generation probabilities $p(x_i | y_i)$, i.e., a word x_i generated by a tag y_i .

3 Proposed Approach

In this work, we for the first time propose a neural CRF-AE and leverage **PLM representations** and **hand-crafted features** for unsupervised POS tagging.

3.1 CRF Encoder w/ PLM Representations

As discussed in §2, the CRF-AE framework consists of two major components, i.e., the CRF encoder and the decoder for sentence reconstruction. We first introduce how to enhance the CRF encoder.



Figure 3: Model architecture of proposed model. ① is the "CRF encoder w/ ELMo representations" and ② is the "reconstruction w/ hand-crafted features".

The major challenge of the CRF encoder is how to induce latent sequences more accurately via effective contextual representations. Like most works before the DL era, Ammar et al. (2014) employ manually designed features to represent contexts.

One of the major advances brought by DL is the strong capability of contextual representation via neural networks like LSTM and Transformer. Furthermore, pre-trained language models, such as ELMo and BERT, greatly amplify this advantage and are shown to be able to substantially improve performance for almost all NLP tasks.

However, few works have tried to utilize such neural contextualized encoders for unsupervised POS tagging, except Tran et al. (2016) and Gupta et al. (2020). Most importantly, according to our knowledge, there is no work so far that successfully employ PLMs for unsupervised POS tagging.

In this work, we propose to employ the contextual representations from PLM to enhance the CRF encoder of the CRF-AE model. Here we use ELMo (Peters et al., 2018) to illustrate our method, which is the same for other PLMs like BERT.

ELMo outputs. The encoder of ELMo consists of three layers (Peters et al., 2018). The bottom layer computes context-free word representations via word-wise character-level convolutional neural networks. The top two layers, each with two unidi-

rectional LSTMs (forward and backward), obtain context-aware word representations by concatenating the forward and backward representations.

After feeding an input sentence into ELMo, each word x_i has three representation vectors, i.e., $(\mathbf{h}_i^0, \mathbf{h}_i^1, \mathbf{h}_i^2)$, corresponding to three encoder layers respectively. Following the standard practice, we take the weighted arithmetic mean (ScalarMix) of output vectors as the final contextualized word representation \mathbf{r}_i for \mathbf{x}_i :

$$\mathbf{r}_i = \gamma \sum_{k=0}^{K-1} \omega_k \mathbf{h}_i^k, \tag{7}$$

where ω_k $(0 \le k < K)$ are softmax-normalized weights¹ and K is the layer number; γ is the scale factor of the entire contextualized word representation. In our final model, we only use \mathbf{h}_i^1 and \mathbf{h}_i^2 , since including \mathbf{h}_i^0 degrades performance (see Table 2).

Minus operation. Apart from specific information of the focused word x_i , the contextualized word representation \mathbf{r}_i from ELMo also contains a lot of common contextual information shared by neighbour words (Ethayarajh, 2019). Therefore, inspired by previous works on constituent parsing (Wang and Chang, 2016; Cross and Huang, 2016), we adopt the minus operation for representations as follows:

$$\mathbf{m}_{i} = \begin{bmatrix} \overrightarrow{\mathbf{r}}_{i} \\ \overleftarrow{\mathbf{r}}_{i} \end{bmatrix} - \begin{bmatrix} \overrightarrow{\mathbf{r}}_{i-1} \\ \overleftarrow{\mathbf{r}}_{i+1} \end{bmatrix}, \qquad (8)$$

where $\vec{\mathbf{r}}_i$ is the forward part of the final contextualized word representation \mathbf{r}_i and $\overleftarrow{\mathbf{r}}_i$ is backward one. \mathbf{m}_i is the word representation of x_i after the minus operation.

Bottleneck MLP. The ELMo adopts large dimensions d, i.e., 1024, to encode as much information as possible. Representations from ELMo contains syntax clues and even semantic ones besides the information about the POS. Inspired by supervised dependency parsing models (Dozat and Manning, 2017; Li and Eisner, 2019), we adopt a bottleneck MLP (MLP^x), whose output vector has a very low dimension. Because of the low dimension of the MLP output, redundant and irrelevant

Feature	John	75th	two-tiered
Word	John	$0 t h^{\dagger}$	UNK [‡]
Uni-gram Suffix	n	h	d
Bi-gram Suffix	hn	th	ed
Tri-gram Suffix	ohn	Oth	red
Has Digit	X	1	X
Has Hyphen	X	X	1
Capitalized	1	X	X

Table 1: Feature templates for feature-rich reconstruction. †: before extracting features, we replace continuous digits into a single "0" in each word. ‡: features appeared less then 50 times in the training data are replaced with a special UNK feature.

information will be stripped away:

$$\mathbf{c}_{i} = \mathsf{MLP}^{\mathtt{x}}(\mathbf{m}_{i})$$

= LeakyReLU ($\mathbf{W}^{\mathtt{x}} \cdot \texttt{LayerNorm}(\mathbf{m}_{i}) + \mathbf{b}^{\mathtt{x}}$),
(9)

where the bottleneck size $d' \ll d$ is output dimensions of the bottleneck projection weight $\mathbf{W}^{\mathtt{x}} \in \mathbb{R}^{d \times d'}$ and the bias $\mathbf{b}^{\mathtt{x}} \in \mathbb{R}^{d'}$.

Scorer. The definition of a POS tagging sequence y given x is identical to equation 4. But the definition of bigram scores is different from the vanilla CRF-AE. Here, a bigram score consists of two parts: a unigram score $s(\mathbf{x}, y_i; \boldsymbol{\phi})$ estimated from ELMo representations and a matrix-maintained transition score $t(y_{i-1}, y_i; \boldsymbol{\phi})$.

$$s(\mathbf{x}, y_{i-1}, y_i; \boldsymbol{\phi}) = s(\mathbf{x}, y_i; \boldsymbol{\phi}) + t(y_{i-1}, y_i; \boldsymbol{\phi}).$$
(10)

Specifically, $s(\mathbf{x}, y_i)$ is calculated as follows:

$$s(\mathbf{x}, y_i; \boldsymbol{\phi}) = \texttt{LayerNorm} (\mathbf{W}^s \cdot \mathbf{c}_i + \mathbf{b}^s) [y_i],$$

(11)

where $\mathbf{W}^{s} \in \mathbb{R}^{d' \times |\mathcal{Y}|}$ is the projection weight of scoring, $\mathbf{b}^{s} \in \mathbb{R}^{|\mathcal{Y}|}$ is the scoring bias, and \mathcal{Y} is the POS tag set. $[y_i]$ is the index selection operation.

3.2 Reconstruction w/ Hand-crafted Features

In Ammar et al. (2014), the reconstruction probabilities are stored and updated as a matrix. The conditional probability $p(x_i | y_i)$, i.e., generating x_i given y_i , is modeled at the whole-word level. This leads to the data sparseness problem. For rare words, the probabilities are usually unreliable.

Therefore, we borrow the idea of feature-rich HMM by Berg-Kirkpatrick et al. (2010). The idea

¹The weights are trained only in the second stage of our training method.

is to utilize rich morphological information to learn more reliable generation probability. For example, suffixes usually provide strong clues to POS categories. In this work, we adopt the feature templates proposed by Berg-Kirkpatrick et al. (2010), as shown in Table 1.

With the hand-crafted features, we then parameterize tag-to-word emission probabilities as local multinomials:

$$p(x_i \mid y_i; \boldsymbol{\theta}) = \frac{\exp\left(\boldsymbol{\theta} \cdot f\left(x_i, y_i\right)\right)}{\sum_{x' \in \mathcal{V}} \exp\left(\boldsymbol{\theta} \cdot f\left(x', y_i\right)\right)} \quad (12)$$

where θ is the feature weight vector and \mathcal{V} is the vocabulary set.

4 Experiments on English PTB

4.1 Settings

Data. Following previous works on unsupervised POS tagging, we conduct experiments on the Wall Street Journal (WSJ) data from PTB, yet with two distinct data settings.

(1) *WSJ-All*. Almost all previous works train and evaluate their models on the entire WSJ data. We report results on WSJ-All for comparison with previous works. However, this data setting is very unfriendly for selecting hyper-parameters, such as stopping and best epoch numbers, M-1 mappings, learning rates, network dimensions, etc. It is probable that some previous works make modeling choices by directly looking at the evaluation performance, since training loss (e.g., data likelihood) is quite loosely correlated with performance. Such details are usually omitted or only implicitly discussed in previous works.

(2) *WSJ-Split*. We follow the practice in unsupervised dependency parsing and divide the WSJ dataset into train (sections 02-21), dev (section 22) and test (section 23). We tune hyper-parameters and study the contributions of individual model components by referring to performance on WSJ-Dev. Moreover, we determine the best many-to-one mappings on WSJ-Dev, which are directly used to compute many-to-one accuracy (M-1) on both WSJ-Dev and WSJ-Test.

We strongly suggest that in future researchers can adopt the WSJ-Split setting. First, the WSJ-Split setting is more realistic because it is able to evaluate a model's generalization ability with out-of-vocabulary words. Second, it is more reasonable and fairer to use WSJ-dev to choose hyper-parameters and it is usually feasible to manually annotate a dev data, even if very small-scale.

Layer	M-1	1-1	VM	LL
0	$79.98 \\ {\scriptstyle \pm 0.3}$	59.13 ±3.4	$73.06 \\ {\scriptstyle \pm 0.9}$	-73.06
1	$\underset{\pm 0.8}{82.61}$	$\underset{\pm 5.0}{63.03}$	$\underset{\pm 1.2}{\textbf{76.98}}$	$-79.52 \\ {}_{\pm 0.8}$
2	$\underset{\pm 1.0}{82.45}$	$\underset{\pm 3.2}{\textbf{60.29}}$	$\underset{\pm 1.0}{\textbf{76.27}}$	$-83.35 \\ {\scriptstyle \pm 0.5}$
{0, 1, 2}	$\underset{\pm 0.3}{\textbf{81.53}}$	$\underset{\pm 4.1}{64.03}$	$\underset{\pm 0.7}{\textbf{76.21}}$	-76.89 $_{\pm0.4}$
$\{1,2\}^{\oplus}$	$\underset{\pm 1.3}{\textbf{82.28}}$	$\underset{\pm 4.5}{63.54}$	$\underset{\pm 1.3}{\textbf{76.91}}$	$\textbf{-78.96}_{\pm 0.4}$
{1, 2}	83.20 ±0.7	65.17 ±2.3	77.69	-80.49

Table 2: Results of utilizing different layers of the ELMo on WSJ-Dev. \oplus means directly concatenating the representation vectors of different layers.

Evaluation metrics. Following previous works, we mainly adopt many-to-one accuracy, and also report one-to-one accuracy (1-1) and validity-measure (VM) values for better comparison. To reduce the effect of performance vibration, we follow previous works, run each model for five times with different seeds, and report the mean and standard deviation. Please see Appendix A for details.

Hyper-parameters. We set the number of predicted POS tags to 45 and the output dimensions of MLP^x to 5. We train each model on the training data for at most 50 epochs, and select the best epoch based on data log-likelihood (LL). Please see Appendix B for full details of hyper-parameters.

Three-stage Training procedure. Unsupervised models are very sensitive to parameter initialization. Inspired by previous works (Han et al., 2017; He et al., 2018), we adopt a three-step progressive training procedure. 1) We train a feature-rich HMM model from random initialization, and produce the 1-best prediction from it for each training sentence. 2) The feature-rich HMM model is used as a teacher to pre-train the CRF-AE model. More concretely, we train the CRF encoder on the pseudo-labeled training data in a supervised fashion for 5 epochs; meanwhile we directly copy the feature weights from the feature-rich HMM model to the decoder of the CRF-AE model. 3) We train our full CRF-AE model on unlabeled training data with parameters obtained in the second step as initialization.

4.2 Model Development on WSJ-Split

Using which ELMo layers. As mentioned above, ELMo produces three representation vectors for each word x, corresponding to its three

encoder layers. Since the usefulness of information contained in different ELMo layers is unknown for our task at hand, we conduct experiments to study which layers to use and how to use them. Table 2 shows the results. When using single-layer representations, it is obvious that using one of the top two layers (1/2) is superior to using the bottom 0-th layer². This is in line with our expectation considering that the 0-th layer corresponds to context-independent word type embeddings. The first layer is superior to the second one, which is consistent with Peters et al. (2018), who also conclude that the information contained by the first layer are more suitable for POS tagging than the second layer.

Then we try to combine multiple layers by using aforementioned ScalarMix in Equation 7. It is clear that using the top two contextualized layers ({1, 2}) achieves best performance. We find that the weight contribution of layer 1 and 2 is about 92% vs. 8%, confirming again that the first contextualized layer provides the majority of syntactic information, while the second layer is more concerned with high-level semantics. We can also see that replacing ScalarMix with simple concatenation leads to large performance drop.

Comparing M-1, 1-1, VM, and LL, we can see that M-1, 1-1 and VM are highly correlated, whereas LL is quite loosely correlated with model performance, suggesting that training loss cannot be used for selecting models or tuning hyperparameters.

In the following, we try to understand **the contribution of different components** by removing one from the full CRF-AE model at a time. Table 3 shows the results.

Usefulness of hand-crafted features. In order to measure the effectiveness of hand-crafted features in the reconstruction part, we revert to the vanilla matrix-maintained version. We can see that rich hand-crafted features are critical and not using them leads to the largest performance drop.

Usefulness of PLMs. We first replace pretrained ELMo with a conventional three-layer BiL-STM encoder that is trained from scratch. We use pre-trained word embeddings of He et al. (2018) as encoder inputs. As expected, performance also declines a lot. It shows that ELMo does provide

Model	M-1	1-1	VM
Full CRF-AE	83.20 ±0.7	65.17 ±2.3	77.69
w/o Features	$\underset{\pm 1.4}{\textbf{76.74}}$	$\underset{\pm 3.5}{61.34}$	$73.55 \\ {\scriptstyle \pm 1.0}$
w/o PLM Repr.	$\underset{\pm 0.9}{\textbf{78.40}}$	$\underset{\pm 4.9}{61.31}$	$72.55 \\ {\scriptstyle \pm 1.6}$
w/o Minus Op.	$\underset{\pm 1.5}{\textbf{81.28}}$	$\underset{\pm 2.8}{63.07}$	$\underset{\pm 1.1}{\textbf{76.04}}$
w/o 3-stage Train	$\underset{\pm 3.4}{\textbf{80.21}}$	$\underset{\pm 2.9}{59.50}$	$75.70 _{\pm 1.2}$
$ELMo \rightarrow BERT$	$\underset{\pm 1.0}{82.30}$	$\underset{\scriptstyle{\pm 5.8}}{62.78}$	$\underset{\pm 1.6}{\textbf{76.13}}$

Table 3: The contribution of different components on WSJ-Dev by removing one component at a time.

very useful information. We have also tried to replace ELMo with BERT without much hyperparameter tuning, as shown in the bottom row, but found that the performance decreases. The results are similar on the multilingual UD data in Table 6. We suspect the reasons are two-fold. First, we did not carefully tune the hyper-parameters for using BERT due to time and resource limitation. Second, we suspect the ELMo word representations suffice and are even more suitable for unsupervised POS tagging. The POS tag of a word usually heavily depends on neighboring words within a small window, which makes the BiLSTM encoder superior to Transformer. The latter is more powerful to capture long-distance dependencies.

Usefulness of minus operation. Besides the minus operation in Equation 8, the default choice is directly using the ELMo output, i.e., \mathbf{r}_i . As shown in the fourth row, models without the minus operation are inferior to the models with the minus operation. We believe it is because the original ELMo representations have a lot of common contextual information shared by neighbour words, and the minus operation can remove them.

Usefulness of the three-stage training procedure. To find out the effect of our three-step progressive training procedure, we randomly initialize model parameters. The result shows that the randomly initialization decreases model performances substantially. It proves that the three-stage training procedure helps models find relatively good initial parameters.

4.3 Results on WSJ-Test

We report results on WSJ-Test in Table 4 and hope future researchers adopt the WSJ-Split setting. Considering that INP-GHMM is the current SOTA

²Minus operations do not apply to vectors at the 0-th layer, i.e., context-independent word type embeddings, which are directly used as \mathbf{m}_i .

Model	M-1	1-1	VM
HMM (re-Impl.)	$\underset{\pm 2.0}{65.25}$	$\underset{\pm 2.6}{47.62}$	$\underset{\pm 1.0}{58.16}$
FHMM (re-Impl.)	$\underset{\pm 1.0}{73.91}$	$\underset{\pm 6.4}{59.13}$	$\underset{\pm 2.5}{\textbf{68.69}}$
INP-GHMM (re-Run)	$\underset{\pm 1.5}{\textbf{76.10}}$	$\underset{\pm 3.2}{53.83}$	$\underset{\pm 0.8}{72.19}$
Ours	82.89 ±0.7	65.32 ±2.5	$\underset{\pm 0.8}{\textbf{78.06}}$

Table 4: Results on WSJ-Test. We re-implement HMM and Feature-rich HMM and re-run INP-GHMM.

model on English PTB, we re-run their open-source code³ with default configuration on WSJ-Split. We re-implement vanilla HMM and feature-rich HMM of Berg-Kirkpatrick et al. (2010), and train them with Adam algorithm via direct gradient descent. Results show that our model is superior to the previous best one, and achieves current SOTA results.

4.4 Performance Comparison on WSJ-All

In order to compare with previous works, we report results on WSJ-All in Table 5. We directly use all hyper-parameters obtained from WSJ-Split.

We can see that our proposed model outperforms all previous works by large margin. The INP-GHMM model (He et al., 2018) achieves the previous best performance on WSJ-All. Our model outperforms theirs by 2.41 and 3.54 on M-1 and VM, respectively.

5 Experiments on Multilingual UD

5.1 Settings

Data. For more thorough comparison with previous works, we also report results⁴ on the Multilingual Universal Dependencies treebank v2.0 (UD), consisting of 10 languages (McDonald et al., 2013). Similar to experiments on English PTB, we adopt two settings for the UD data, i.e., UD-Split and UD-All. For UD-Split, we adopt the default partition of the UD data.

Hyper-parameters. We directly adopt most hyper-parameters obtained for PTB with three important exceptions. First, The number of predicted POS tags is changed to 12. Second, since the scale of data for each language diverge a lot, we adjust the feature cutoff threshold to be proportional to the token number against English partition. For example, the "de" data contains about 293k tokens,

Model	M-1	1-1	VM
HMM (B'10)	63.1 ±1.3	_	_
FHMM (B'10)	$\underset{\pm 1.1}{\textbf{75.5}}$	_	_
FHMM (re-Impl.)	$\underset{\pm 2.2}{74.70}$	$\underset{\pm 4.3}{60.88}$	$\underset{\pm 2.1}{\textbf{68.53}}$
Brown (C'10)	76.1	60.7	68.8
S-CODE (Y'12)	$\underset{\pm 0.7}{80.23}$	_	$72.07 _{\pm 0.4}$
GHMM (L'15)	$\underset{\pm 1.0}{75.4}$	_	$\underset{\pm 0.5}{68.5}$
NHMM (T'16)	79.1	60.7	71.7
INP-GHMM (H'18)	80.8 ±1.3	-	$\underset{\pm 0.7}{\textbf{74.1}}$
MIM (S'19)	$\underset{\pm 0.8}{\textbf{78.1}}$	_	_
SyntDEC (G'20)	$\underset{\pm 0.9}{78.2}$	_	_
Ours	83.21 ±1.2	65.78 ±2.8	77.64

Table 5: Results on WSJ-All. Here, B'10 is for Berg-Kirkpatrick et al. (2010), C'10 for Christodoulopoulos et al. (2010), Y'12 for Yatbaz et al. (2012), L'15 for Lin et al. (2015), T'16 for Tran et al. (2016), H'18 for He et al. (2018), S'19 for Stratos (2019), and, G'20 for Gupta et al. (2020).

which is about 28% of that of "en" (1M), and therefore we set the threshold to 14 ($28\% \times 50$). Third, we adjust the hand-crafted features to accommodate the 12-tag UD standard and characteristics of different languages, detailed in the following.

Modifications on hand-crafted features. The fine-grained 45-tag WSJ standard is greatly different from the coarse-grained 12-tag UD standard adopted by the multilingual UD datasets (Petrov et al., 2012). Therefore, we start from the features of Berg-Kirkpatrick et al. (2010) in Table 1 as the base, and make adjustments from two aspects.

(1) Adjustments for UD. We remove the "Capitalized" feature, which is originally purposed to distinguish proper and common nouns which correspond to a single UD tag. Moreover, we replace all punctuation marks with a special "PUNCT" word form, add a new feature template "is-Punctuation", as UD uses a single tag for punctuation marks.

(2) Adjustments for specific languages⁵. The UD tag set doesn't distinguish inflections such as numbers, tenses, and genders. We find this can be accommodated by customizing suffix uni/bi/trigram features. We simply remove a certain number of ending characters (related to inflectional affixes)

³https://github.com/jxhe/struct-learning-with-flow

⁴We run each model for five times with different random seeds.

⁵We only adopt language-specific adjustments for "de", "en", "es", "fr", "it" and "pt-br".

UD-Dev	de*	en*	es*	fr*	id	it*	ja	ko	pt-br*	sv	Mean
Full CRF-AE	$76.18 \\ {\scriptstyle \pm 4.0}$	80.30 ±2.2	81.76 ±1.0	82.56 ±0.4	80.99 ±0.5	80.32 ±1.8	$\underset{\pm 1.1}{\textbf{86.61}}$	$73.00 \\ {\scriptstyle \pm 3.0}$	81.38 ±2.2	74.93	79.80
w/o Features	$\underset{\pm 1.2}{\textbf{70.18}}$	$73.64 \\ {\scriptstyle \pm 1.3}$	$\underset{\pm 6.0}{73.91}$	$72.51 \\ {}_{\pm 2.8}$	$\underset{\pm 2.3}{73.09}$	$\underset{\pm 1.3}{\textbf{68.18}}$	$75.96 \\ {\scriptstyle \pm 3.8}$	$\underset{\pm 3.3}{63.36}$	$\underset{\pm 6.0}{68.95}$	$\underset{\scriptscriptstyle\pm4.4}{65.81}$	70.56
w/o UD Adjust.	$\underset{\pm 1.8}{73.94}$	$\underset{\pm 4.0}{73.63}$	$\underset{\pm 3.1}{77.95}$	$\underset{\pm 3.1}{\textbf{76.05}}$	$\underset{\pm 1.4}{\textbf{76.57}}$	$72.12 \\ {}_{\pm 4.7}$	$\underset{\pm 1.1}{\textbf{82.19}}$	$\underset{\pm 3.2}{74.37}$	$\underset{\pm 3.2}{74.47}$	$\underset{\pm 3.3}{64.96}$	74.62
w/o Language Adjust.	$75.99 \atop \scriptstyle \pm 1.2$	$\underset{\pm 2.3}{\textbf{78.97}}$	$\underset{\pm 2.1}{\textbf{79.66}}$	$\underset{\pm 2.0}{\textbf{79.60}}$	=	$71.71 \\ {\scriptstyle \pm 6.5}$	=	=	$\underset{\pm 1.9}{73.72}$	—	77.52
w/o PLM Repr.	75.11	$\underset{\pm 1.4}{\textbf{76.50}}$	$\underset{\pm 0.9}{\textbf{78.78}}$	$\underset{\pm 1.7}{\textbf{82.16}}$	$\underset{\pm 1.5}{77.96}$	$70.54 \\ {\scriptstyle \pm 2.6}$	$\underset{\pm 1.0}{82.26}$	$\underset{\pm 1.4}{\textbf{65.47}}$	$\underset{\pm 2.6}{\textbf{79.11}}$	$\underset{\pm 1.7}{\textbf{68.94}}$	75.68
w/o 3-stage Training	77.52 ±3.1	$72.18 \\ {\scriptstyle \pm 3.6}$	$\underset{\pm 3.8}{74.70}$	$\underset{\pm 3.2}{\textbf{78.26}}$	$78.62 \\ {\scriptstyle \pm 2.8}$	$70.85 \\ {}_{\pm 2.2}$	$\underset{\pm 1.4}{\textbf{83.93}}$	76.01 ±2.7	77.26	$\underset{\pm 5.5}{68.50}$	75.78
$ELMo \rightarrow mBERT$	$75.96 \\ {\scriptstyle \pm 4.2}$	$\underset{\pm 1.3}{\textbf{78.12}}$	$\underset{\pm 1.6}{\textbf{79.67}}$	$\underset{\pm 0.7}{81.09}$	$\underset{\pm 0.5}{\textbf{80.13}}$	$75.66 \atop {\scriptstyle \pm 3.0}$	86.92 ±1.5	$\underset{\pm 3.1}{\textbf{73.18}}$	$\underset{\pm 2.4}{\textbf{80.69}}$	$\underset{\pm 2.8}{\textbf{72.87}}$	78.43
UD-Test	de	en	es	fr	id	it	ja	ko	pt-br	sv	Mean
HMM (re-Impl.)	$60.13_{\pm 1.2}$	$\underset{\scriptstyle\pm2.4}{63.85}$	$\underset{\pm 3.8}{64.68}$	$\underset{\pm 4.5}{65.50}$	$\underset{\pm 2.1}{\textbf{66.23}}$	$\underset{\pm 1.8}{\textbf{66.14}}$	$\underset{\pm 0.4}{60.02}$	$\underset{\pm0.6}{46.55}$	57.65 ±6.3	57.07 ±5.1	60.78
FHMM (re-Impl.)	$70.95 \\ {}_{\pm 2.7}$	$\underset{\pm 0.7}{75.58}$	$\underset{\pm 1.2}{\textbf{76.26}}$	$\underset{\pm 1.8}{\textbf{77.33}}$	$73.67 \atop \scriptstyle \pm 1.0$	$\underset{\pm 2.5}{74.73}$	$72.47 _{\pm 0.5}$	$\underset{\pm 1.7}{63.77}$	$77.67 \\ {}_{\pm 2.1}$	$\underset{\pm 2.3}{\textbf{67.99}}$	73.04
GHMM (re-Run)	$\underset{\pm 1.2}{81.95}$	$\underset{\pm 1.5}{75.49}$	$\underset{\pm 1.7}{\textbf{78.92}}$	$\underset{\scriptscriptstyle{\pm 7.4}}{73.48}$	$\underset{\pm 4.3}{\textbf{76.09}}$	$72.87 \\ {\scriptstyle \pm 4.5}$	$\underset{\scriptscriptstyle{\pm1.1}}{75.41}$	$\underset{\pm 1.7}{\textbf{68.31}}$	$\underset{\pm 5.5}{74.84}$	$72.15 \\ {}_{\pm 3.6}$	74.95
INP-GHMM (re-Run)	$\underset{\pm 1.1}{\textbf{82.79}}$	$\underset{\pm 1.5}{\textbf{75.93}}$	$\underset{\pm 2.9}{\textbf{79.61}}$	$\underset{\pm 7.2}{73.55}$	$\underset{\pm 3.6}{\textbf{76.92}}$	$73.60 \\ {\scriptstyle \pm 4.8}$	$\underset{\pm 1.2}{\textbf{76.32}}$	$\underset{\pm 2.3}{\textbf{67.85}}$	$75.43 _{\pm 5.0}$	$\underset{\pm 3.5}{74.33}$	75.63
Ours	77.46	79.60	80.46	79.36	80.77	80.82	79.93	75.48	81.23	76.29	79.14
	±4.5	±2.2	±0.9	±0.6	± 0.5	±2.2	±2.5	± 3.1	± 2.3	±2.0	

Table 6: M-1 accuracy on UD-Split. **Upper Part:** The contribution of different components on UD-Dev by removing one component at a time. * means adopting the language-specific suffix features for this language. "=" means the result is identical to that of Full CRF-AE. Lower Part: Performance comparison on UD-Test.

for a word form before extracting suffix features. We remove the last character for "it", and the last two characters for "de". For "fr", "es", and "pt-br", we remove last two characters if the word ends with "s", and the last one otherwise. For "en", we only remove the last "s" letter if applicable.

5.2 Results on UD-Split

Table 6 shows the M-1 results. For 1-1 and VM-results, please refer to Table 10 and Table 9 in the Appendix.

We perform ablation study on UD-Dev. Most of the results show the same trend as on WSJ-Dev. In particular, we find that our two adjustment strategies for the UD data are very helpful, and the UD adjustment is more helpful. After observation, we find that without UD-specific adjustments, punctuation marks are more likely to be divided into multiple tags. For example, models may assign three different tags to periods, commas, and quotation marks. Moreover, with the removal of the "Capitalized" feature, which is one of the UD adjustments, the models no longer distinguish common and proper nouns and assign one tag to them.⁶ Without language-specific adjustments, highly inflected languages, e.g., Italian (it) and Brazilian Portuguese (pt-br), are more likely to distinguish words by their number or gender rather than part-ofspeech. For example, in English, models without making language-specific adjustments will tend to split nouns into two classes: single nouns and plural nouns ending with "s".

We report the M-1 results on UD-Test in Table 6. We run our implemented vanilla HMM and featurerich HMM, and the latter adopt the same features after UD and linguage adjustments. Unfortunately, we are unable to re-run SyntDEC, the current SOTA on UD-All, since its authors (Gupta et al., 2020) have not yet released their code. We also re-run INP-GHMM (He et al., 2018) with their released code, which is the current SOTA on WSJ-All. We take context-free word representations (0-th layer) of ELMo as inputs of INP-GHMM, which should be better than Skip-Gram embeddings. Please see Appendix C for details of hyper-parameters.

Results show that our models achieve the highest M-1 accuracy on 9 out of 10 languages, except "de". After investigation on why our models fail to outperform INP-GHMM on "de", we find that the direct reason is that INP-GHMM is initialized

⁶However, we find that some models still divide nouns into multiple tags by some unknown criteria.

with GHMM, and the simple GHMM is already more superior to our model. Therefore, we replace FHMM with GHMM in the first stage of our training procedure. Results show that our models are substantially improved in "de". However, we still do not understand the reason behind these results, which we leave for future investigation due to time limitation.

5.3 Performance Comparison on UD-All

To compare with previous works, we report results on UD-All in Table 11 in the Appendix. For thorough comparison, we also re-run GHMM and INP-GHMM on UD-All. The results show identical trends as those on UD-Split.

6 Related Works

Unsupervised POS tagging. In addition to HMMs and the CRF-AE, other approaches for unsupervised POS tagging are as follows.

(1) Clustering. The clustering approach, as a mainstream unsupervised learning technique, is also investigated for unsupervised POS tagging Yatbaz et al. (2012); Yuret et al. (2014); Gupta et al. (2020). All these works adopt the k-means algorithm to divide word tokens into different groups. The main difference among them is how to represent words. Yatbaz et al. (2012) propose to learn context-free word embeddings by minimizing the distance between each word and its substituted words. Substituted words are selected according to a n-gram language model. Yuret et al. (2014) extend their previous work to produce contextsensitive word embeddings. Gupta et al. (2020) adopt a deep clustering approach that uses a feedforward neural network to transform word representations from mBERT into a lower-dimension clustering-friendly space. Transformation with reconstruction loss and clustering are jointly trained. Unfortunately, all three works have not released their source code.

(2) *Mutual information maximization*. The mutual information maximization approach is proposed by Stratos (2019). The idea is that we can predict POS tags in two ways (using the words themselves or their context), and predictions from these two ways should agree as more as possible.

Utilizing PLMs for unsupervised tagging or parsing. As discussed earlier, SyntDEC (Gupta et al., 2020) is the only work that employs PLMs for unsupervised POS tagging based on deep clustering. As for unsupervised parsing, Wu et al. (2020) propose a perturbed masking technique to estimate inter-word correlations and then induce syntax trees from those correlations. Kim et al. (2020) extract constituency trees from the PLMs through capturing syntactical proximity between representations of two adjacent words (or subwords). If the proximity is loose, then it is likely that the middle position of the two words corresponds to some constituent boundary. Cao et al. (2020) successfully exploit PLMs for unsupervised constituency parsing based on constituency test, achieving SOTA performance.

Utilizing CRF-AE. Cai et al. (2017) apply CRF-AE to unsupervised dependency parsing. They use the encoder to generate a most likely dependency tree and then force the decoder to reconstruct the input sentence from the tree. Zhang et al. (2017) propose a neural CRF-AE for semi-supervised learning on sequence labeling problems (including POS tagging) and Jia et al. (2020) adopt a neural CRF-AE for semi-supervised semantic parsing.

7 Conclusions

This work bridges PLMs and hand-crafted features for unsupervised POS tagging. Based on the CRF-AE framework, we employ powerful contextualized representations from PLMs in the CRF encoder, and incorporate rich morphological features for better reconstruction. Our proposed approach achieves new SOTA on 45-tag English PTB and 12tag multilingual UD datasets, outperforming previous results by large margin. Experiments and analysis show that rich features and PLM representations are critical for the superior performance of our model. Meanwhile, simple adjustments of hand-crafted features are key for the success of our model on languages other than English.

Acknowledgments

We thank the anonymous reviewers for the helpful comments. We are very grateful to Wei Jiang for his early-stage exploration on unsupervised POS tagging. We also thank Chen Gong, Yu Zhang, Ying Li, Qingrong Xia, Yahui Liu, and Tong Zhu for their help in paper writing and polishing. This work was supported by National Natural Science Foundation of China (Grant No. 62176173, 61876116) and a Project Funded by the Priority Academic Program Development (PAPD) of Jiangsu Higher Education Institutions.

References

- Waleed Ammar, Chris Dyer, and Noah A. Smith. 2014. Conditional random field autoencoders for unsupervised structured prediction. In *Proc. of NeurIPS*, pages 3311–3319.
- Taylor Berg-Kirkpatrick, Alexandre Bouchard-Côté, John DeNero, and Dan Klein. 2010. Painless unsupervised learning with features. In Proc. of NAACL-HLT, pages 582–590, Los Angeles, California.
- Bernd Bohnet, Ryan McDonald, Gonçalo Simões, Daniel Andor, Emily Pitler, and Joshua Maynez. 2018. Morphosyntactic tagging with a meta-BiLSTM model over context sensitive token encodings. In *Proc. of ACL*, pages 2642–2652, Melbourne, Australia.
- Jiong Cai, Yong Jiang, and Kewei Tu. 2017. CRF autoencoder for unsupervised dependency parsing. In *Proc. of EMNLP*, pages 1638–1643, Copenhagen, Denmark.
- Steven Cao, Nikita Kitaev, and Dan Klein. 2020. Unsupervised parsing via constituency tests. In *Proc. of EMNLP*, pages 4798–4808, Online.
- Wanxiang Che, Yijia Liu, Yuxuan Wang, Bo Zheng, and Ting Liu. 2018. Towards better UD parsing: Deep contextualized word embeddings, ensemble, and treebank concatenation. In *Proc. of CoNLL*, pages 55–64, Brussels, Belgium.
- Christos Christodoulopoulos, Sharon Goldwater, and Mark Steedman. 2010. Two decades of unsupervised POS induction: How far have we come? In *Proc. of EMNLP*, pages 575–584, Cambridge, MA.
- James Cross and Liang Huang. 2016. Span-based constituency parsing with a structure-label system and provably optimal dynamic oracles. In *Proc. of EMNLP*, pages 1–11, Austin, Texas.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proc. of NAACL-HLT*, pages 4171– 4186, Minneapolis, Minnesota.
- Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *Proc. of ICLR*.
- Kawin Ethayarajh. 2019. How contextual are contextualized word representations? comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In *Proc. of EMNLP*, pages 55–65, Hong Kong, China.
- João Graça, Kuzman Ganchev, Ben Taskar, and Fernando C. N. Pereira. 2009. Posterior vs parameter sparsity in latent variable models. In *Proc. of NeurIPS*, pages 664–672.
- Vikram Gupta, Haoyue Shi, Kevin Gimpel, and Mrinmaya Sachan. 2020. Clustering contextualized representations of text for unsupervised syntax induction. ArXiv preprint, abs/2010.12784.

- Wenjuan Han, Yong Jiang, and Kewei Tu. 2017. Dependency grammar induction with neural lexicalization and big training data. In *Proc. of EMNLP*, pages 1683–1688, Copenhagen, Denmark.
- Junxian He, Graham Neubig, and Taylor Berg-Kirkpatrick. 2018. Unsupervised learning of syntactic structure with invertible neural projections. In *Proc. of EMNLP*, pages 1292–1302, Brussels, Belgium.
- Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *ArXiv preprint*, abs/1508.01991.
- Zixia Jia, Youmi Ma, Jiong Cai, and Kewei Tu. 2020. Semi-supervised semantic dependency parsing using CRF autoencoders. In *Proc. of ACL*, pages 6795–6805, Online.
- Taeuk Kim, Jihun Choi, Daniel Edmiston, and Sanggoo Lee. 2020. Are pre-trained language models aware of phrases? simple but strong baselines for grammar induction. In *Proc. of ICLR*.
- Dan Klein and Christopher Manning. 2004. Corpusbased induction of syntactic structure: Models of dependency and constituency. In *Proc. of ACL*, pages 478–485, Barcelona, Spain.
- Xiang Lisa Li and Jason Eisner. 2019. Specializing word embeddings (for parsing) by information bottleneck. In *Proc. of EMNLP*, pages 2744–2754, Hong Kong, China.
- Percy Liang, Ben Taskar, and Dan Klein. 2006. Alignment by agreement. In *Proc. of NAACL-HLT*, pages 104–111, New York City, USA.
- Chu-Cheng Lin, Waleed Ammar, Chris Dyer, and Lori Levin. 2015. Unsupervised POS induction with word embeddings. In *Proc. of NAACL-HLT*, pages 1311–1316, Denver, Colorado.
- 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 *Proc. of ACL*, pages 92–97, Sofia, Bulgaria.
- Bernard Merialdo. 1994. Tagging English text with a probabilistic model. *Computational Linguistics*, 20(2):155–171.
- Fernando Pereira and Yves Schabes. 1992. Insideoutside reestimation from partially bracketed corpora. In Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York, February 23-26, 1992.
- Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL-HLT*, pages 2227– 2237, New Orleans, Louisiana.

- Slav Petrov, Dipanjan Das, and Ryan McDonald. 2012. A universal part-of-speech tagset. In *Proc. of LREC*, pages 2089–2096, Istanbul, Turkey.
- Andrew Rosenberg and Julia Hirschberg. 2007. Vmeasure: A conditional entropy-based external cluster evaluation measure. In *Proc. of EMNLP*, pages 410–420, Prague, Czech Republic.
- Ruslan Salakhutdinov, Sam T. Roweis, and Zoubin Ghahramani. 2003. Optimization with EM and expectation-conjugate-gradient. In *Proc. of ICML*, pages 672–679.
- Yoav Seginer. 2007. Fast unsupervised incremental parsing. In *Proc. of ACL*, pages 384–391, Prague, Czech Republic.
- Karl Stratos. 2019. Mutual information maximization for simple and accurate part-of-speech induction. In *Proc. of NAACL-HLT*, pages 1095–1104, Minneapolis, Minnesota.
- Karl Stratos, Michael Collins, and Daniel Hsu. 2016. Unsupervised part-of-speech tagging with anchor hidden Markov models. *Transactions of the Association for Computational Linguistics*, 4:245–257.
- Ke M. Tran, Yonatan Bisk, Ashish Vaswani, Daniel Marcu, and Kevin Knight. 2016. Unsupervised neural hidden Markov models. In *Proc. of the Workshop on Structured Prediction for NLP*, pages 63–71, Austin, TX.
- Wenhui Wang and Baobao Chang. 2016. Graph-based dependency parsing with bidirectional LSTM. In *Proc. of ACL*, pages 2306–2315, Berlin, Germany.
- Zhiyong Wu, Yun Chen, Ben Kao, and Qun Liu. 2020. Perturbed masking: Parameter-free probing for analyzing and interpreting BERT. In *Proc. of ACL*, pages 4166–4176, Online.
- Mehmet Ali Yatbaz, Enis Sert, and Deniz Yuret. 2012. Learning syntactic categories using paradigmatic representations of word context. In *Proc. of EMNLP*, pages 940–951, Jeju Island, Korea.
- Deniz Yuret, Mehmet Ali Yatbaz, and Enis Sert. 2014. Unsupervised instance-based part of speech induction using probable substitutes. In *Proc. of COLING*, pages 2303–2313, Dublin, Ireland.
- Xiao Zhang, Yong Jiang, Hao Peng, Kewei Tu, and Dan Goldwasser. 2017. Semi-supervised structured prediction with neural CRF autoencoder. In *Proc. of EMNLP*, pages 1701–1711, Copenhagen, Denmark.
- Houquan Zhou, Yu Zhang, Zhenghua Li, and Min Zhang. 2020. Is POS tagging necessary or even helpful for neural dependency parsing? In *Proc. of NLPCC*, pages 179–191.

A Details of Evaluation Metrics

The core issue of the unsupervised POS tagging evaluation is that we can not directly compute the tagging accuracy since the correspondence between ground truth tags and predicted tag indexes (indexto-tag mapping) is unknown and varies from model to model. The different evaluation metrics handle this issue in different way.

A.1 Many-to-One Accuracy (M-1)

M-1 is the most commonly used evaluation metric. It addresses the problem of correspondence by assigning each predicted tag index $j \in \mathcal{P}$ to its most frequent co-occurring ground truth tag $g_i \in \mathcal{G}$:

$$M-1(\mathbf{A}) = \sum_{j} \max_{g_i} \mathbf{A}_{g_i,j}, \qquad (13)$$

where $\mathbf{A} \in \mathbb{R}^{n \times n}$ is contingency matrix and the matrix item $\mathbf{A}_{g_i,j}$ is the number of words which are annotated as a g_i and predicted as a j by the model to be evaluated. This metric, obviously, allows different predicted indexes to map to the same ground truth tag⁷.

A.2 One-to-One Accuracy (1-1)

Different from M-1 that we allows a ground truth tag g_i corresponding to multiple predicted indexes, 1-1 only allows one predicted index can be assigned to a ground truth tag, and vice versa. Calculating 1-1 is a typical assignment problem that finding a optimal bijection function $f : \mathcal{P} \to \mathcal{G}$ that maximums the correct matching count from all possible bijection functions \mathcal{F} :

$$1-1(\mathbf{A}) = \max_{f \in \mathcal{F}} \sum_{j} \mathbf{A}_{f(j),j}.$$
 (14)

In this paper we solve this assignment problem with the Hungarian algorithm 8 .

A.3 Validity-Measure (VM)

VM (Rosenberg and Hirschberg, 2007) is an entropy-based measure, which do not require the index-to-tag mapping and considers two criteria: homogeneity h and completeness c. The homogeneity of a predicted index indicates the purity of its co-occurring ground truth tags. The predicted index j results the highest homogeneity when it

⁷In the WSJ-Split data setting, the index-to-tag mapping of metrics for WSJ-Dev and WSJ-Test are both observed from WSJ-Dev.

⁸https://en.wikipedia.org/wiki/Hungar ian_algorithm

only co-occur with g_i , i.e., $\mathbf{A}_{g_i,j} = \sum_{g_{i'}} \mathbf{A}_{g_{i',j}}$, and has a low homogeneity it appears with different ground truth tags randomly. The homogeneity of a model is the simply the sum of the homogeneity of all index indicates. The completeness is symmetrical to homogeneity, merely exchanging the position of predicted indexes and ground truth tags. VM employs the conditional entropy to measure the value of homogeneity and completeness:

$$H(\mathcal{G} \mid \mathcal{P}, \mathbf{A}) = -\sum_{j} \sum_{i} \frac{\mathbf{A}_{g_{i},j}}{N} \log \frac{\mathbf{A}_{g_{i},j}}{\sum_{g_{i'}} \mathbf{A}_{g_{i'},j}},$$
(15)
$$H(\mathcal{P} \mid \mathcal{G}, \mathbf{A}) = -\sum_{i} \sum_{j} \frac{\mathbf{A}_{g_{i},j}}{N} \log \frac{\mathbf{A}_{g_{i},j}}{\sum_{j'} \mathbf{A}_{g_{i},j'}},$$
(16)

where $\mathbf{A} \in \mathbb{R}^{n \times n}$ is contingency matrix and the matrix item $\mathbf{A}_{g_i,j}$ is the number of words which are annotated as a g_i and predicted as a j.

To alleviate the impact of the size of the dataset and the numbers of the POS class, the conditional entropy is normalized by the entropy of ground truth POS tag $H(\mathcal{G}, \mathbf{A})$ and $H(\mathcal{P}, \mathbf{A})$ for homogeneity and completeness, respectively:

$$h(\mathbf{A}) = 1 - \frac{H(\mathcal{G} \mid \mathcal{P}, \mathbf{A})}{H(\mathcal{G}, \mathbf{A})}, \qquad (17)$$

$$c(\mathbf{A}) = 1 - \frac{H(\mathcal{P} \mid \mathcal{G}, \mathbf{A})}{H(\mathcal{P}, \mathbf{A})},$$
 (18)

where

$$H(\mathcal{G}, \mathbf{A}) = -\sum_{i} \frac{\sum_{j} \mathbf{A}_{g_{i}, j}}{N} \log \frac{\sum_{j} \mathbf{A}_{g_{i}, j}}{N}, \quad (19)$$

$$H(\mathcal{P}, \mathbf{A}) = -\sum_{j} \frac{\sum_{g_i} \mathbf{A}_{g_i, j}}{N} \log \frac{\sum_{g_i} \mathbf{A}_{g_i, j}}{N}.$$
 (20)

Completeness is symmetrical to homogeneity, merely exchanging \mathcal{G} and \mathcal{P} in the formulas.

In order to balance the significance between homogeneity and completeness, VM is defined as the weighted harmonic mean of homogeneity and completeness:

$$VM(\mathbf{A}) = \frac{(1+\beta)h(\mathbf{A})c(\mathbf{A})}{\beta h(\mathbf{A}) + c(\mathbf{A})}, \qquad (21)$$

where β are set to 1 in experiments.

B Details of Hyper-parameters

B.1 Model

The number of predicted POS tags is 45 for experiments on WSJ and 12 for Multilingual

experiments. The ELMo parameters we use for experiments on WSJ are "Original (5.5B)"⁹ from AllenNLP. The parameters for Multilingual are from "ELMoForManyLangs"¹⁰ (Che et al., 2018). We use "bert-base-cased" (BERT) and "bert-base-multilingual-cased" (mBERT)¹¹ for the ablation study of PLMs on WSJ and UD respectively. We do not fine-tune ELMo parameters. The dropout value is uniformly set to 0.33, and the negative slope of the activation function Leaky-ReLU is set to 1×10^{-2} . The seeds we selected for experiments are 0, 1, 2, 3, 4.

B.2 Feature

We set the feature cutoff threshold to 50, which means that all features that appear in the training data less than 50 times are replaced with a special "UNK" feature.

B.3 Training

We use a mini-batch update strategy with a batch size of 5000 words and optimize models with Adam. The learning rate used in the training of the FHMM in the first step is 0.5. The CRF encoder is then trained on pseudo-labeled data for 5 epochs with a learning rate of 2×10^{-3} in the subsequent pre-training step. In the final step, the CRF encoder has a learning rate of 1×10^{-2} , and we set the reconstruction learning rate to 2×10^{-1} . Other hyperparameters are identical among all three steps in training procedure, The β_1 and β_2 are both 0.9. The learning rate decay is 0.75 per 45 epochs, the gradient clipping value is 5, and the weight decay value λ is 1×10^{-5} .

C Details of Re-run INP-GHMM Hyper-parameters

We use the word-wise character-level convolutional layer (0-th layer) of ELMo to extract word embeddings. We use 8 coupling layers. To accelerate the training of INP-GHMM, we increase the batch sizes from 32 to 512 sentences. We also decrease the learning rate from 1×10^{-3} to 5×10^{-4} , as we found that high learning rates lead to performance decreases as training progresses.

⁹https://allennlp.org/elmo

¹⁰https://github.com/HIT-SCIR/ELMoForM
anyLangs

¹¹https://github.com/google-research/b
ert

	Gender	Singular	Plural	Gloss
Adi	М.	rosso	rossi	rad
Auj.	F.	F. rossa rosse M. lo li F. la le	Ieu	
Dron	M.	1 0	li	him/
FIOII.	F.	la	le	her/them
Noun	M.	bambin o	bambin i	hou/girl
Noun	F.	bambin a	bambin e	boy/giri

Table 7: Examples of inflections of Italian adjective, pronoun, and noun. "M." means the gender Masculine and "F." means Feminine.

Langs.	Uni-gram	Bi-gram	Tri-gram
it	mus <u>e</u> o	mu <u>se</u> o	m <u>use</u> o
π	mus <u>e</u> i	mu <u>se</u> i	m <u>use</u> i
da	mus <u>e</u> um	mu <u>se</u> um	m <u>use</u> um
ue	mus <u>e</u> en	mu <u>se</u> en	m <u>use</u> en
fr	mus <u>é</u> e	mu <u>sé</u> e	m <u>usé</u> e
	mus <u>é</u> es	mu <u>sé</u> es	m <u>usé</u> es
86	mus <u>e</u> o	mu <u>se</u> o	m <u>use</u> o
63	mus <u>e</u> os	mu <u>se</u> os	m <u>use</u> os
nt_hr	mus <u>e</u> u	mu <u>se</u> u	m <u>use</u> u
pt-or	mus <u>e</u> us	mu <u>se</u> us	m <u>use</u> us
on	museu <u>m</u>	muse <u>um</u>	mus <u>eum</u>
	museu <u>m</u> s	muse <u>um</u> s	mus <u>eum</u> s

Table 8: Language-specific suffix features for the UD datasets. The underlined <u>characters</u> represent extracted suffix features.

D Explanation of adjustments for specific languages on UD

Most of European languages are inflected languages. Some words are inflected for number, gender, tense, aspect and so on. For example in English nouns are inflected for number (singular or plural); verbs for tense. A major way to inflect words is adding inflectional suffixes to the end of words, e.g., English nouns inflected for number with suffix "s" ("museum" \rightarrow "museums"). Therefore, in some languages suffixes is more closely related to inflections than coarse-grained POS. For instance, as shown in Table 7, the last letter of Italian words is highly corresponding to gender and number, and haves little connection to coarse-grained POS. In this work, we simply remove a certain number of ending characters for a word form before extracting suffix features, as shown in Table 8.

UD-Dev	de*	en*	es*	fr^*	id	it*	ja	ko	pt-br*	sv	Mean
Full CRF-AE	$68.52 \\ \pm 2.1$	68.64 ±3.0	69.55 ±1.9	70.30 ±0.3	$\underset{\pm 0.8}{\textbf{61.94}}$	69.77 ±1.6	$\underset{\pm 1.4}{47.91}$	$\underset{\pm 2.2}{\textbf{36.61}}$	68.14 ±2.5	66.99 ±2.9	62.84
w/o Features	$\underset{\pm 1.3}{60.12}$	$\underset{\pm 0.8}{61.11}$	$\underset{\scriptstyle\pm4.0}{64.11}$	$\underset{\pm 0.6}{64.28}$	$\underset{\scriptscriptstyle{\pm 2.5}}{54.26}$	$\underset{\pm 1.5}{60.40}$	$\underset{\pm 3.1}{35.63}$	$\underset{\pm 3.0}{24.29}$	$\underset{\pm 3.5}{57.72}$	$\underset{\pm 4.2}{59.98}$	54.19
w/o UD Adjust.	$\underset{\pm 2.0}{64.30}$	$\underset{\pm 2.7}{60.13}$	$\underset{\pm 3.4}{\textbf{66.68}}$	$\underset{\scriptscriptstyle\pm4.0}{64.10}$	$\underset{\pm 1.7}{56.45}$	$\underset{\pm 3.2}{\textbf{60.98}}$	$\underset{\pm 1.9}{42.05}$	$\underset{\pm 2.4}{\textbf{37.48}}$	$\underset{\pm 2.6}{60.64}$	$\underset{\pm 1.9}{57.98}$	57.08
w/o Language Adjust.	$\underset{\pm 1.5}{\textbf{67.26}}$	$\underset{\pm 1.4}{\textbf{67.85}}$	$\underset{\pm 2.5}{65.83}$	$\underset{\pm 2.3}{\textbf{66.63}}$	=	$\underset{\pm 6.9}{60.90}$	=	=	$\underset{\pm 1.8}{58.53}$	—	60.05
w/o PLM Repr.	$\underset{\pm 1.9}{65.36}$	$\underset{\pm 1.5}{64.28}$	$\underset{\pm 1.6}{64.18}$	$\underset{\pm 1.3}{\textbf{67.76}}$	$\underset{\pm 1.3}{56.26}$	$\underset{\pm 1.8}{61.64}$	$\underset{\scriptscriptstyle{\pm1.4}}{42.15}$	$\underset{\pm 1.0}{26.29}$	$\underset{\pm 2.6}{64.21}$	$\underset{\pm 1.0}{58.27}$	57.04
w/o 3-stage Training	69.37 ±2.3	$\underset{\pm 3.8}{61.44}$	$\underset{\pm 1.9}{\textbf{65.31}}$	$\underset{\scriptstyle{\pm 2.8}}{66.52}$	$\underset{\pm 2.9}{61.02}$	$\underset{\pm 1.4}{61.95}$	$\underset{\pm 2.3}{44.78}$	38.96 ±1.9	$\underset{\pm 3.2}{\textbf{66.11}}$	$\underset{\pm 4.2}{61.72}$	59.72
$ELMo \rightarrow mBERT$	$\underset{\pm 2.5}{\textbf{67.54}}$	$\underset{\pm 1.6}{\textbf{66.71}}$	$\underset{\pm 2.5}{\textbf{66.74}}$	$\underset{\pm 0.7}{\textbf{68.42}}$	$\underset{\pm 0.8}{60.55}$	$\underset{\pm 2.0}{65.79}$	$\underset{\pm 1.7}{\textbf{48.39}}$	$\underset{\pm 2.1}{\textbf{34.39}}$	$\underset{\pm 2.6}{\textbf{66.45}}$	$\underset{\pm 1.3}{64.41}$	60.94
UD-Dev	de	en	es	fr	id	it	ja	ko	pt-br	sv	Mean
HMM (re-Impl.)	$\underset{\pm 1.1}{40.86}$	$\underset{\pm1.8}{45.01}$	$\underset{\pm 2.2}{46.99}$	50.24 ±2.1	$\underset{\pm 1.8}{\textbf{37.41}}$	$\underset{\pm 2.3}{47.68}$	$\underset{\pm 1.0}{27.31}$	$\underset{\pm 0.8}{8.32}$	38.33 ±4.2	$\underset{\pm 4.3}{41.77}$	38.39
FHMM (re-Impl.)	58.25 ±1.3	$\underset{\pm 1.7}{62.75}$	$\underset{\pm 2.2}{60.37}$	$\underset{\pm 1.2}{63.68}$	$\underset{\pm 0.7}{49.32}$	$\underset{\pm 1.1}{60.51}$	$\underset{\pm 0.8}{34.09}$	$\underset{\pm 1.7}{21.48}$	$\underset{\pm 2.0}{60.47}$	$\underset{\pm 1.0}{55.41}$	52.63
GHMM (re-Run)	$72.61 \\ {\scriptstyle \pm 1.5}$	$\underset{\pm1.8}{61.34}$	$\underset{\pm 3.2}{\textbf{68.47}}$	$\underset{\scriptstyle\pm 5.6}{62.46}$	$\underset{\scriptscriptstyle\pm4.5}{55.48}$	$\underset{\scriptscriptstyle\pm3.3}{62.90}$	$\underset{\pm 0.9}{\textbf{38.88}}$	$\underset{\pm 2.1}{\textbf{30.00}}$	$\underset{\scriptstyle\pm4.1}{60.77}$	$\underset{\scriptscriptstyle{\pm 2.4}}{62.42}$	57.53
INP-GHMM (re-Run)	$\underset{\pm 0.8}{73.68}$	$\underset{\scriptscriptstyle{\pm 2.1}}{61.72}$	$\underset{\pm 3.5}{\textbf{68.76}}$	$\underset{\pm 5.0}{62.69}$	$\underset{\pm 3.1}{56.91}$	$\underset{\scriptstyle\pm4.0}{64.27}$	$\underset{\pm 1.0}{\textbf{39.09}}$	$\underset{\pm 1.9}{\textbf{30.91}}$	$\underset{\pm 4.0}{61.84}$	$\underset{\pm 2.6}{\textbf{65.16}}$	58.50
Ours	$\underset{\pm 2.2}{69.55}$	67.97 ±2.7	69.05 ±1.9	69.05 ±0.6	62.08 ±0.5	70.42 ±1.6	44.23 ±1.7	39.50 ±2.8	67.65 ±2.3	68.36 ±2.5	62.79
Ours (GHMM Init.)	76.17 ±2.7	_	_	_	_	_	_	_	_	-	_

Table 9: VM results on UD-Split. **Upper Part:** The contribution of different components on UD-Dev by removing one component at a time. * means adopting the language-specific suffix features for this language. "=" means the result is identical to that of Full CRF-AE. **Lower Part:** Performance comparison on UD-Test.

UD-Dev	de*	en*	es*	fr*	id	it*	ja	ko	pt-br*	sv	Mean
Full CRF-AE	$\underset{\pm1.8}{\textbf{66.86}}$	71.36 ±5.4	$\underset{\pm 5.0}{58.69}$	$\underset{\pm 3.0}{63.32}$	60.03 ±1.3	$\underset{\pm 0.6}{\textbf{65.37}}$	$\underset{\pm 2.4}{43.01}$	35.53 ±2.0	64.13 ±5.8	66.81 ±6.2	59.51
w/o Features	57.43 ±4.2	$\underset{\pm 3.5}{61.98}$	$\underset{\scriptstyle\pm4.4}{59.52}$	$\underset{\pm 0.7}{62.20}$	$\underset{\pm 5.3}{52.29}$	$\underset{\scriptstyle{\pm 5.9}}{61.75}$	$\underset{\pm 3.3}{36.50}$	$\underset{\pm 3.9}{29.37}$	$\underset{\pm 6.0}{52.35}$	$\underset{\pm 6.2}{58.34}$	53.17
w/o UD Adjust.	$62.33_{\pm 3.3}$	$\underset{\pm 4.7}{58.76}$	63.07 ±5.9	$\underset{\pm 7.4}{58.29}$	$\underset{\scriptscriptstyle\pm3.8}{52.90}$	59.74 ±4.7	$\underset{\pm 2.7}{35.45}$	$\underset{\pm 2.9}{\textbf{38.84}}$	$\underset{\pm 2.8}{57.38}$	$\underset{\pm 7.8}{57.80}$	54.46
w/o Language Adjust.	$63.24_{\pm 2.6}$	$\underset{\pm 4.5}{69.83}$	$\underset{\pm 6.8}{55.61}$	$\underset{\pm 3.6}{61.90}$	=	$\underset{\pm 8.1}{54.91}$	=	=	$\underset{\pm 6.6}{50.37}$	—	56.12
w/o PLM Repr.	$\underset{\pm 3.2}{64.68}$	$\underset{\pm 3.0}{69.08}$	$\underset{\scriptstyle\pm5.5}{55.64}$	64.21 ±1.5	$\underset{\pm 3.0}{53.96}$	$\underset{\pm 3.7}{63.07}$	$\underset{\pm 1.3}{40.92}$	$\underset{\pm 1.5}{28.62}$	$\underset{\pm 5.2}{61.52}$	$\underset{\pm 4.1}{61.32}$	56.30
w/o 3-stage Training	68.49 ±3.4	$\underset{\pm 5.9}{62.57}$	$\underset{\scriptscriptstyle{\pm 5.8}}{56.15}$	$\underset{\pm 6.9}{61.02}$	61.30 ±6.4	$\underset{\pm 3.5}{56.33}$	$\underset{\pm 2.5}{39.72}$	$\underset{\pm 3.3}{\textbf{38.30}}$	$\underset{\pm 5.5}{60.97}$	$\underset{\pm 4.8}{59.09}$	56.39
$ELMo \rightarrow mBERT$	$\underset{\pm 3.9}{64.31}$	$\underset{\pm 2.7}{\textbf{70.74}}$	$\underset{\pm 4.9}{55.95}$	$\underset{\pm 1.1}{62.59}$	$\underset{\scriptscriptstyle{\pm 2.4}}{56.70}$	$\underset{\pm 1.4}{60.43}$	$\underset{\pm 2.0}{\textbf{43.15}}$	$\underset{\scriptscriptstyle\pm4.4}{\textbf{33.71}}$	$\underset{\pm 5.0}{59.55}$	$\underset{\pm 2.5}{\textbf{66.29}}$	57.34
UD-Dev	de	en	es	fr	id	it	ja	ko	pt-br	sv	Mean
HMM (re-Impl.)	$42.06_{\pm 2.3}$	$\underset{\pm 1.9}{50.10}$	$\underset{\pm 3.5}{50.48}$	$\underset{\pm 2.4}{52.08}$	36.70 ±4.7	$\underset{\pm 2.1}{46.08}$	$\underset{\pm 4.6}{28.98}$	$\underset{\pm 1.3}{23.43}$	$\underset{\pm 3.5}{38.35}$	$\underset{\pm 4.5}{42.95}$	41.12
FHMM (re-Impl.)	$\underset{\pm 3.3}{61.83}$	$\underset{\pm 3.4}{\textbf{68.62}}$	$\underset{\pm 6.4}{57.22}$	$\underset{\pm 1.8}{62.36}$	$\underset{\pm 1.8}{52.92}$	$\underset{\pm 2.0}{62.24}$	$\underset{\pm 1.9}{\textbf{41.16}}$	$\underset{\pm 2.3}{28.49}$	$\underset{\pm 4.2}{62.99}$	$\underset{\pm 2.6}{61.17}$	55.90
GHMM (re-Run)	$\underset{\pm 2.7}{\textbf{67.81}}$	$\underset{\pm 4.0}{53.41}$	61.81 ±4.3	$\underset{\pm 6.2}{55.41}$	$\underset{\pm 5.7}{45.46}$	$\underset{\scriptstyle\pm4.8}{55.16}$	$\underset{\pm 1.7}{\textbf{37.95}}$	$\underset{\pm 4.8}{29.55}$	$\underset{\pm 6.5}{53.52}$	$\underset{\pm 6.0}{\textbf{56.14}}$	51.62
INP-GHMM (re-Run)	$\underset{\pm 1.6}{\textbf{68.33}}$	$\underset{\scriptscriptstyle\pm4.4}{54.03}$	$\underset{\pm 5.0}{61.63}$	$\underset{\pm 5.0}{56.68}$	$\underset{\pm 4.1}{46.39}$	$\underset{\scriptstyle{\pm 5.3}}{57.61}$	$\underset{\pm 2.5}{\textbf{38.38}}$	$\underset{\pm 3.8}{\textbf{30.79}}$	$\underset{\scriptstyle\pm5.9}{53.42}$	$\underset{\pm 6.5}{57.98}$	52.52
Ours	$\underset{\pm 2.5}{67.55}$	70.28 ±5.2	$\underset{\pm 5.0}{59.82}$	64.39 ±2.3	60.01 ±1.3	65.58 ±0.7	$\underset{\pm 2.8}{41.07}$	34.04 ±3.1	63.39 ±5.6	67.63 ±6.2	59.38
Ours (GHMM Init.)	71.94 ± 4.0	-	-	-	-	_	-	-	_	-	_

Table 10: 1-1 accuracy on UD-Split. **Upper Part:** The contribution of different components on UD-Dev by removing one component at a time. * means adopting the language-specific suffix features for this language. "=" means the result is identical to that of Full CRF-AE. Lower Part: Performance comparison on UD-Test.

Model	de	en	es	fr	id	it	ja	ko	pt-br	SV	Mean
Brown (C'10)	60.0	62.9	67.4	66.4	59.3	66.1	60.3	47.5	67.4	61.9	61.9
FHMM (B'10)	$\underset{\pm 1.8}{67.5}$	$\underset{\pm 3.5}{62.4}$	$\underset{\pm 3.1}{\textbf{67.1}}$	$\underset{\pm 4.5}{62.1}$	$\underset{\pm 3.9}{61.3}$	$\underset{\pm 2.9}{52.9}$	$\underset{\pm 2.9}{\textbf{78.2}}$	$\underset{\pm 3.6}{60.5}$	$\underset{\pm 2.2}{63.2}$	$\underset{\pm 2.5}{56.7}$	63.2
AHMM (S'16)	63.4	71.4	74.3	71.9	67.3	60.2	69.4	61.8	65.8	61.0	66.7
MIM (S'19)	$75.4 _{\pm 1.5}$	$\underset{\pm 1.7}{\textbf{73.1}}$	$\underset{\pm 1.0}{73.1}$	$\underset{\pm 2.9}{\textbf{70.4}}$	$\underset{\pm 1.5}{73.6}$	$\underset{\pm 3.3}{\textbf{67.4}}$	$\underset{\pm 0.4}{\textbf{77.9}}$	$\underset{\pm 1.2}{\textbf{65.6}}$	$\underset{\pm 2.3}{\textbf{70.7}}$	$\underset{\pm 1.5}{\textbf{67.1}}$	71.4
SyntDEC (G'20)	$\underset{\pm 1.8}{81.5}$	$\underset{\pm 1.1}{\textbf{76.5}}$	$\underset{\pm 1.9}{\textbf{78.9}}$	$\underset{\pm 3.9}{\textbf{70.7}}$	$\underset{\pm 1.1}{\textbf{76.8}}$	$\underset{\pm 3.3}{\textbf{71.7}}$	$\underset{\pm 1.2}{\textbf{84.7}}$	$\underset{\pm 1.5}{69.7}$	77.7 ±2.1	$\underset{\pm 3.9}{\textbf{68.8}}$	75.7
GHMM (re-Run)	$\underset{\pm 1.9}{\textbf{82.16}}$	$75.31 \\ {}_{\pm 2.1}$	$\underset{\pm 2.2}{\textbf{80.27}}$	$\underset{\pm 3.7}{\textbf{76.59}}$	$76.52 \\ {}_{\pm 4.0}$	$\underset{\pm 5.8}{72.78}$	$\underset{\pm 0.9}{\textbf{79.81}}$	$\underset{\pm 2.0}{67.52}$	$\underset{\pm 4.1}{74.99}$	$73.60 \\ {\scriptstyle \pm 2.9}$	75.96
INP-GHMM (re-Run)	$\underset{\pm1.8}{\textbf{83.48}}$	$\underset{\pm 1.4}{\textbf{76.02}}$	$\underset{\pm 2.7}{81.68}$	$77.40 \\ {\scriptstyle \pm 3.4}$	$\underset{\pm 2.7}{77.72}$	$72.55 \\ {}_{\pm 5.5}$	$\underset{\pm 1.5}{\textbf{79.41}}$	$\underset{\pm 1.8}{68.07}$	$\underset{\pm 4.5}{75.27}$	$\underset{\pm 3.1}{\textbf{74.48}}$	76.61
Ours	$\underset{\pm 2.0}{\textbf{82.41}}$	80.79 ±1.1	82.65 ±2.0	82.67 ±0.6	81.09 ±1.3	$78.13 \\ {\scriptstyle \pm 1.6}$	85.52 ±1.1	74.87 ±2.7	79.67 ±2.4	78.44 ±3.5	80.67
Ours (GHMM Init.)	86.93 ±1.2	_	_	_	_	_	_	_	_	_	_

Table 11: M-1 accuracy on UD-All. C'10 is for Christodoulopoulos et al. (2010), B'10 for Berg-Kirkpatrick et al. (2010), S'16 for Stratos et al. (2016), S'19 for Stratos (2019), and G'20 for Gupta et al. (2020).