STRUCTURAL KNOWLEDGE DISTILLATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Knowledge distillation is a critical technique to transfer knowledge between models, typically from a large model (the teacher) to a smaller one (the student). The objective function of knowledge distillation is typically the cross-entropy between the teacher and the student's output distributions. However, for structured prediction problems, the output space is exponential in size; therefore, the cross-entropy objective becomes intractable to compute and optimize directly. In this paper, we derive a factorized form of the knowledge distillation objective for structured prediction, which is tractable for many typical choices of the teacher and student models. In particular, we show the tractability and empirical effectiveness of structural knowledge distillation between sequence labeling and dependency parsing models under four different scenarios: 1) the teacher and student share the same factorization form of the output structure scoring function; 2) the student factorization produces smaller substructures than the teacher factorization; 3) the teacher factorization produces smaller substructures than the student factorization; 4) the factorization forms from the teacher and the student are incompatible.

1 INTRODUCTION

Deeper and larger neural networks have led to significant improvement in accuracy in various tasks, but they are also more computationally expensive and unfit for resource-constrained scenarios such as online serving. An interesting and viable solution to this problem is knowledge distillation (KD) (Buciluǎ et al., 2006; Ba & Caruana, 2014; Hinton et al., 2015), which can be used to transfer the knowledge of a large model (the teacher) to a smaller model (the student). In the field of natural language processing, for example, KD has been successfully applied to compress massive pretrained language models such as BERT (Devlin et al., 2019) and XLM-R (Conneau et al., 2020) into much smaller and faster models without significant loss in accuracy (Tang et al., 2019; Sanh et al., 2019; Tsai et al., 2019; Mukherjee & Hassan Awadallah, 2020).

A typical approach to KD is letting the student mimic the teacher model's output probability distributions on the training data by using the cross-entropy objective. For structured prediction problems, however, the output space is exponentially large, making the cross-entropy objective intractable to compute and optimize directly. Previous approaches to structural KD either choose to perform KD on local decisions or substructures instead of on the full output structure, or resort to top-K approximation of the objective (Kim & Rush, 2016; Kuncoro et al., 2016; Wang et al., 2020).

This paper shows that the structural KD objective can be factorized by the student's factorization of the output structure scoring function. Consequently, we can tractably compute and optimize the structural KD objective as long as we estimate the teacher's marginal distributions over the substructures produced by the student's factorization. We apply this technique to structural KD between sequence labeling and dependency parsing models under four different scenarios.

1. The teacher and student share the same factorization form of the output structure scoring function.
2. The student factorization produces smaller substructures than the teacher factorization.
3. The teacher factorization produces smaller substructures than the student factorization.
4. The factorization forms from the teacher and the student are incompatible.

We empirically show the effectiveness of our approaches in all the cases. Our approaches outperform strong knowledge distillation baselines in most cases, and with unlabeled data, our approaches can

further improve student models' performance. In a zero-shot cross-lingual transfer case, we also show that student models trained by our approaches can even outperform the teacher models.

2 BACKGROUND

2.1 STRUCTURED PREDICTION

Structured prediction aims to predict a structured output such as a sequence, a tree or a graph. In this paper, we focus on structured prediction problems with a discrete output space, which include most of the structured prediction tasks in natural language processing (e.g., POS tagging, named entity recognition, and dependency parsing) and many structured prediction tasks in computer vision (e.g., image segmentation). We further assume that the scoring function of the output structure can be factorized into scores of a polynomial number of substructures. Consequently, we can calculate the conditional probability of the output structure $\boldsymbol{y}$ given an input $\boldsymbol{x}$ as follows:

$$P(\boldsymbol{y}|\boldsymbol{x}) = \frac{\exp\left(\text{Score}(\boldsymbol{y}, \boldsymbol{x})\right)}{\sum_{\boldsymbol{y}' \in \mathbb{Y}(\boldsymbol{x})} \exp\left(\text{Score}(\boldsymbol{y}', \boldsymbol{x})\right)} = \frac{\exp\left(\text{Score}(\boldsymbol{y}, \boldsymbol{x})\right)}{\mathcal{Z}(\boldsymbol{x})} = \frac{\prod_{\boldsymbol{u} \in \boldsymbol{y}} \exp\left(\text{Score}(\boldsymbol{u}, \boldsymbol{x})\right)}{\mathcal{Z}(\boldsymbol{x})} \quad (1)$$

where $\mathbb{Y}(\boldsymbol{x})$ represents all possible output structures given the input $\boldsymbol{x}$, $\text{Score}(\boldsymbol{y}, \boldsymbol{x})$ is the scoring function that evaluates the quality of the output $\boldsymbol{y}$, $\mathcal{Z}(\boldsymbol{x})$ is the partition function, and $\boldsymbol{u}$ is a substructure of $\boldsymbol{y}$. We define the substructure space $\mathbb{U}(\boldsymbol{x})$ as the set of substructures of all possible output structures given input $\boldsymbol{x}$.

Take sequence labeling for example. Given a sentence $\boldsymbol{x}$, the output space $\mathbb{Y}(\boldsymbol{x})$ contains all possible label sequences of $\boldsymbol{x}$. In linear-chain CRF, a popular model for sequence labeling, the scoring function $\text{Score}(\boldsymbol{y}, \boldsymbol{x})$ is computed by summing up all the transition scores $\text{Score}((y_{i-1}, y_i), \boldsymbol{x})$ and emission scores $\text{Score}(y_i, \boldsymbol{x})$ where i ranges over all the positions in sentence $\boldsymbol{x}$, and the substructure space $\mathbb{U}(\boldsymbol{x})$ contains all possible position-specific labels $\{y_i\}$ and label pairs $\{(y_{i-1}, y_i)\}$.

2.2 KNOWLEDGE DISTILLATION

Knowledge distillation is a technique that trains a small student model by encouraging it to imitate the output probability distribution of a large teacher model. The typical KD objective function is the cross-entropy between the output distributions predicted by the teacher model and the student model:

$$\mathcal{L}_{\text{KD}} = -\sum_{\boldsymbol{y} \in \mathbb{Y}(\boldsymbol{x})} P_t(\boldsymbol{y}|\boldsymbol{x}) \log P_s(\boldsymbol{y}|\boldsymbol{x}) \quad (2)$$

where P_t and P_s are the teacher's and the student's distributions respectively.

During training, the student jointly learns from the gold targets and the distributions predicted by the teacher by optimizing the following objective function:

$$\mathcal{L}_{\text{student}} = \lambda \mathcal{L}_{\text{KD}} + (1 - \lambda)\mathcal{L}_{\text{target}}$$

where λ is an interpolation coefficient between the target loss $\mathcal{L}_{\text{target}}$ and the structural KD loss $\mathcal{L}_{\text{KD}}$. Following Clark et al. (2019); Wang et al. (2020), one may apply teacher annealing in training by decreasing λ linearly from 1 to 0. Because KD does not require gold labels, unlabeled data can also be used in the KD loss.

3 STRUCTURAL KNOWLEDGE DISTILLATION

When performing knowledge distillation on structured prediction, a major challenge is that the structured output space is exponential in size, leading to intractable computation of the KD objective in Eq. 2. However, if the scoring function of the student model can be factorized into scores of sub-

structures (Eq. 1), then we can derive the following factorized form of the structural KD objective.

$$\begin{aligned}
\mathcal{L}_{\text{KD}} &= -\sum_{\boldsymbol{y}\in\mathbb{Y}(\boldsymbol{x})} P_t(\boldsymbol{y}|\boldsymbol{x}) \log P_s(\boldsymbol{y}|\boldsymbol{x}) = -\sum_{\boldsymbol{y}\in\mathbb{Y}(\boldsymbol{x})} P_t(\boldsymbol{y}|\boldsymbol{x}) \sum_{\boldsymbol{u}\in\boldsymbol{y}} \text{Score}_s(\boldsymbol{u},\boldsymbol{x}) + \log \mathcal{Z}_s(\boldsymbol{x}) \\
&= -\sum_{\boldsymbol{y}\in\mathbb{Y}(\boldsymbol{x})} P_t(\boldsymbol{y}|\boldsymbol{x}) \sum_{\boldsymbol{u}\in\mathbb{U}(\boldsymbol{x})} \mathbf{1}_{\boldsymbol{u}\in\boldsymbol{y}} \text{Score}_s(\boldsymbol{u},\boldsymbol{x}) + \log \mathcal{Z}_s(\boldsymbol{x}) \\
&= -\sum_{\boldsymbol{u}\in\mathbb{U}_s(\boldsymbol{x})} \sum_{\boldsymbol{y}\in\mathbb{Y}(\boldsymbol{x})} P_t(\boldsymbol{y}|\boldsymbol{x}) \mathbf{1}_{\boldsymbol{u}\in\boldsymbol{y}} \text{Score}_s(\boldsymbol{u},\boldsymbol{x}) + \log \mathcal{Z}_s(\boldsymbol{x}) \\
&= -\sum_{\boldsymbol{u}\in\mathbb{U}_s(\boldsymbol{x})} P_t(\boldsymbol{u}|\boldsymbol{x}) \text{Score}_s(\boldsymbol{u},\boldsymbol{x}) + \log \mathcal{Z}_s(\boldsymbol{x}) \qquad (3)
\end{aligned}$$

where $\mathbf{1}_{\text{condition}}$ is 1 if the condition is true and 0 otherwise. From Eq. 3, we see that if $\mathbb{U}_s(\boldsymbol{x})$ is polynomial in size and $P_t(\boldsymbol{u}|\boldsymbol{x})$ can be tractably estimated, then the structural KD objective can be tractably computed and optimized. In the rest of this section, we will show that this is indeed the case for some of the most widely used models in sequence labeling and dependency parsing, two representative structured prediction tasks in natural language processing. Based on the difference in score factorization between the teacher and student models, we divide our discussion into four scenarios.

3.1 Teacher and Student Share the Same Factorization Form

Case 1a: Linear-Chain CRF ⇒ Linear-Chain CRF In this case, both the teacher and the student are linear-chain CRF models. An example application is to compress a state-of-the-art CRF model for named entity recognition (NER) that is based on large pretrained contextualized embeddings to a smaller CRF model with static embeddings that is more suitable for fast online serving.

For a CRF student model, if we absorb emission scores into transition scores, then the substructure space $\mathbb{U}_s(\boldsymbol{x})$ contains every two adjacent tokens $\{y_{i-1}, y_i\}$ for $i = 1, \ldots, n$, with n being the sequence length. The substructure marginal $P_t(y_{i-1}, y_i|\boldsymbol{x})$ of the teacher model can be tractably calculated using the forward-backward algorithm.

$$P_t(y_{i-1}, y_i|\boldsymbol{x}) \propto \alpha(y_{i-1}) \times \exp(\text{Score}((y_{i-1}, y_i), \boldsymbol{x})) \times \beta(y_i) \qquad (4)$$

where $\alpha(y_{i-1})$ and $\beta(y_i)$ are forward and backward scores.

Case 1b: Graph-based Dependency Parsing ⇒ Dependency Parsing as Sequence Labeling In this case, we use the biaffine parser proposed by Dozat et al. (2017) as the teacher and the sequence labeling approach proposed by Strzyz et al. (2019) as the student for the dependency parsing task. The biaffine parser is one of the state-of-the-art models, while the sequence labeling parser provides a good speed-accuracy tradeoff. There is a big gap in accuracy between the two models and therefore KD can be used to improve the accuracy of the sequence labeling parser.

Here we follow the head-selection formulation of dependency parsing without the tree constraint. The dependency parse tree $\boldsymbol{y}$ is represented by $\langle y_1, \ldots, y_n \rangle$, where n is the sentence length and $y_i = (h_i, l_i)$ denotes the dependency head of the i-th token of the input sentence, with h_i being the index of the head token and l_i being the dependency label.

The biaffine parser predicts the dependency head for each token independently. It models separately the probability distribution of the head index $P_t(h_i|\boldsymbol{x})$ and the probability distribution of the label $P_t(l_i|h_i, \boldsymbol{x})$. The sequence labeling parser also predicts the head of each token independently. With a MaxEnt decoder, it predicts a joint probability distribution of the index and label of the head token $P_s(h_i, l_i|\boldsymbol{x})$ for each token. Therefore, these two models share the same factorization in which each substructure is a dependency arc specified by y_i. $\mathbb{U}_s(\boldsymbol{x})$ thus contains all possible dependency arcs among tokens of the input sentence $\boldsymbol{x}$. The substructure marginal predicted by the teacher can be easily derived as:

$$P_t(h_i, l_i|\boldsymbol{x}) = P_t(h_i|\boldsymbol{x}) \times P_t(l_i|h_i, \boldsymbol{x}) \qquad (5)$$

Note that in this case, the sequence labeling parser uses a MaxEnt decoder, which is locally normalized for each substructure. Therefore, the structural KD objective in Eq. 3 can be reduced to the

following form without the need for calculating the student partition function $\mathcal{Z}_s(\boldsymbol{x})$.

$$\mathcal{L}_{\text{KD}} = -\sum_{\boldsymbol{u} \in \mathbb{U}_s(\boldsymbol{x})} P_t(\boldsymbol{u}|\boldsymbol{x}) \times \log P_s(\boldsymbol{u}|\boldsymbol{x}) \tag{6}$$

In all the cases below except **Case 3**, the student model is locally normalized and hence we can follow this form of objective.

3.2 Student Factorization Produces Smaller Substructures than Teacher Factorization

Case 2a: Linear-Chain CRF ⇒ MaxEnt In this case, we use a linear-chain CRF model as the teacher and a MaxEnt model as the student. Previous work (Yang et al., 2018; Wang et al., 2020) shows that a linear-chain CRF decoder often leads to better performance than a MaxEnt decoder for many sequence labeling tasks. Still, the simplicity and efficiency of the MaxEnt model is desirable. Therefore, it makes sense to perform KD from a linear-chain CRF to a MaxEnt model.

As mentioned in **Case 1a**, the substructures of a linear-chain CRF model are consecutive labels $\{y_{i-1}, y_i\}$. In contrast, a MaxEnt model predicts the label probability distribution $P_s(y_i|\boldsymbol{x})$ of each token independently and hence the substructures are individual labels $\{y_i\}$. To calculate the substructure marginal of the teacher $P_t(y_i|\boldsymbol{x})$, we can again utilize the forward-backward algorithm:

$$P_t(y_i|\boldsymbol{x}) \propto \alpha(y_i) \times \beta(y_i) \tag{7}$$

where $\alpha(y_i)$ and $\beta(y_i)$ are forward and backward scores.

Case 2b: Second-Order Dependency Parsing ⇒ Dependency Parsing as Sequence Labeling The biaffine parser is a first-order dependency parser, which scores each dependency arc in a parse tree independently. A second-order dependency parser scores pairs of dependency arcs with a shared token. The substructures of second-order parsing are therefore all the dependency arc pairs with a shared token. It has been found that second-order extensions of the biaffine parser often have higher parsing accuracy (Wang et al., 2019; Zhang et al., 2020). Therefore, we may take a second-order dependency parser as the teacher to improve a sequence labeling parser.

Here we consider the second-order dependency parser of Wang et al. (2019) with the head-selection constraint. It employs mean field variational inference to estimate the probabilities of arc existence $P_t(h_i|\boldsymbol{x})$ and uses a first-order biaffine model to estimate the probabilities of arc labels $P_t(l_i|h_i, \boldsymbol{x})$. Therefore, the substructure marginal $P_t(y_i|\boldsymbol{x})$ can be calculated in the same way as Eq. 5.

3.3 Teacher Factorization Produces Smaller Substructures than Student Factorization

Case 3: MaxEnt ⇒ Linear-Chain CRF Here we consider KD in the opposite direction of **Case 2a**. An example application is zero-shot cross-lingual NER. Previous work (Pires et al., 2019; Wu & Dredze, 2019) has shown that multilingual BERT (M-BERT) has strong zero-shot cross-lingual transferability in NER tasks. Many such models employ a MaxEnt decoder. We may distill knowledge from such models to a linear-chain CRF model with static monolingual embeddings for faster speed and potentially better accuracy. As described in **Case 1a**, the substructures of a linear-chain CRF model are consecutive labels $\{y_{i-1}, y_i\}$. Because of the label independence and local normalization in the MaxEnt model, we can compute the substructure marginal of the MaxEnt teacher as follows:

$$P_t(y_{i-1}, y_i|\boldsymbol{x}) = P_t(y_{i-1}|\boldsymbol{x}) P_t(y_i|\boldsymbol{x}) \tag{8}$$

3.4 Factorization Forms From Teacher and Student are Incompatible

Case 4: NER as Parsing ⇒ MaxEnt Very recently, Yu et al. (2020) propose to solve the NER task as graph-based dependency parsing and achieve state-of-the-art performance. They represent each named entity with a dependency arc from the first token to the last token of the named entity, and represent the entity type with the arc label. However, the time complexity of this method is higher than sequence labeling NER methods. In this case, we take a parsing-based NER model as our teacher and a MaxEnt model with the BIOES label scheme as our student.

The two models adopt very different representations of NER output structures. The parsing-based teacher model represents an NER output of a sentence with a set of labeled dependency arcs and defines its score as the sum of arc scores. The MaxEnt model represents an NER output of a sentence with a sequence of BIOES labels and defines its score as the sum of token-wise label scores. Therefore, the factorization forms of these two models are incompatible.

Computing the substructure marginal of the teacher $P_t(y_i|\boldsymbol{x})$, where $y_i \in \{B_l, I_l, E_l, S_l, O|l \in L\}$ and L is the set of entity types, is much more complicated than in the previous cases. Take $y_i = B_l$ for example. $P_t(y_i = B_l|\boldsymbol{x})$ represents the probability of the i-th word being the beginning of a multi-word entity of type 'l'. In the parsing-based teacher model, this probability is proportional to the summation of exponentiated scores of all the output structures that contain a dependency arc of label 'l' with the i-th word as its head and with its length larger than 1. It is intractable to compute such marginal probabilities by enumerating all the output structures, but we can tractably compute them using dynamic programming. See Appendix A for a detailed description of our dynamic programming method.

4 EXPERIMENTS

We evaluate our approaches described in section 3 on monolingual NER (**Case 1a, 2a, 4**), zero-shot cross-lingual NER (**Case 3**), and dependency parsing (**Case 1b, 2b**).

4.1 SETTINGS

Datasets We use CoNLL 2002/2003 datasets (Tjong Kim Sang, 2002; Tjong Kim Sang & De Meulder, 2003) for **Case 1a**, **2a** and **4**, and use WikiAnn datasets (Pan et al., 2017) for **Case 1a**, **2a 3**, and **4**. The CoNLL datasets contain the corpora of four Indo-European languages. We use the same four languages from the WikiAnn datasets. For cross-lingual transfer in **Case 3**, we use the four Indo-European languages as the source for the teacher model and additionally select four languages from different language families as the target for the student models.[1]

We use the standard training/development/test split for the CoNLL datasets. For WikiAnn, we follow the sampling of Wang et al. (2020) with 12000 sentences for English and 5000 sentences for each of the other languages. We split the datasets by 3:1:1 for training/development/test. For **Case 1b** and **2b**, we use Penn Treebank (PTB) 3.0. We follow the same pre-processing pipeline as in Ma et al. (2018). For unlabeled data, we sample sentences that belong to the same domain of the labeled data from the WikiAnn datasets for **Case 1a**, **2a**, **3** and **4**, and we use the BLLIP corpus[2] for **Case 1b** and **2b**.

Models For the student models in all the cases, we use fastText word embeddings and character embeddings as the word representation. For **Case 1a**, **2a** and **4**, we concatenate the multilingual BERT (M-BERT) (Devlin et al., 2019), Flair (Akbik et al., 2018), fastText (Bojanowski et al., 2017) embeddings and character embeddings (Santos & Zadrozny, 2014) as the word representations for the monolingual teacher models. For **Case 3**, we use M-BERT embeddings for the teacher. Also for **Case 3**, we fine-tune the teacher model on the training set of the four Indo-European languages from the WikiAnn dataset and train student models on the four additional languages. For the teacher models in **Case 1b** and **2b**, we simply use the same embeddings as the student.

Baselines We compare our approaches with two baselines in our experiments: training the student model without KD (**Baseline**) and training the student model with a previous KD approach (**Base. KD**, not available in some cases). In **Case 1a**, the KD baseline is posterior KD of Wang et al. (2020) which is a structure-level KD approach for linear-chain CRFs. In **Case 2a**, we train a MaxEnt teacher and run token-level KD in which we minimize the cross-entropy loss between individual label distributions predicted by the teacher and student. In **Case 3**, the KD baseline is also token-level KD. In **Case 1b**, **2b** and **4**, there are no usable KD baselines.

[1]The four languages from the CoNLL datasets are Dutch, English, German and Spanish and the four target languages for **Case 3** are Basque, Hebrew, Persian and Tamil. We use ISO 639-1 language codes (`https://en.wikipedia.org/wiki/List_of_ISO_639-1_codes`) to represent each language.

[2]Brown Laboratory for Linguistic Information Processing (BLLIP) 1987-89 WSJ Corpus Release 1.

Table 1: Averaged F1 scores for NER and labeled attachment scores (LAS) for dependency parsing. CoN. and Wiki. represents CoNLL and WikiAnn respectively. +U denotes experiments with unlabeled data.

Scenario	**Case 1a**			**Case 1b**		**Case 2a**			**Case 2b**		**Case 3**	**Case 4**		
Dataset	CoN.	Wiki.	+U	PTB	+U	CoN.	Wiki.	+U	PTB	+U	+U	CoN.	Wiki.	+U
Teacher	89.15	88.52	88.52	95.96	95.96	89.15	88.52	88.52	96.04	96.04	56.01	88.57	88.38	88.38
Baseline	84.70	83.31	84.19	89.85	90.03	83.87	80.86	82.40	89.85	90.03	41.11	83.87	80.86	82.10
Base. KD	85.27	83.73	84.91	-	-	84.25	82.09	83.07	-	-	38.42	-	-	-
Struct. KD	**85.35**	**84.12**	**85.24**	**91.83**	**91.98**	**84.50**	**82.23**	**83.34**	**91.78**	**91.94**	**45.28**	**84.28**	**81.45**	**82.44**

Training For MaxEnt and linear-chain CRF models, we use the same hyper-parameters as in Akbik et al. (2018). For dependency parsing, we use the same hyper-parameters as in Zhang et al. (2020) for graph-based parsing models and Strzyz et al. (2019) for parsing as sequence labeling models. For M-BERT fine-tuning in **Case 3**, we mix the training data of the four source datasets and train the teacher model with the AdamW optimizer (Loshchilov & Hutter, 2018) with a learning rate of 5e-5 for 10 epochs. We tune the KD temperature (Hinton et al., 2015) in $\{1, 2, 3, 4, 5\}$ and the loss interpolation annealing rate in $\{0.5, 1.0, 1.5\}$. For experiments using unlabeled data, in addition to labeled data, we also use the teacher's prediction on the unlabeled data as pseudo labeled data to train the baselines. This can be seen as an approximate KD method and hence even the **Baseline** benefits from KD to some extent under this setup.

4.2 Results

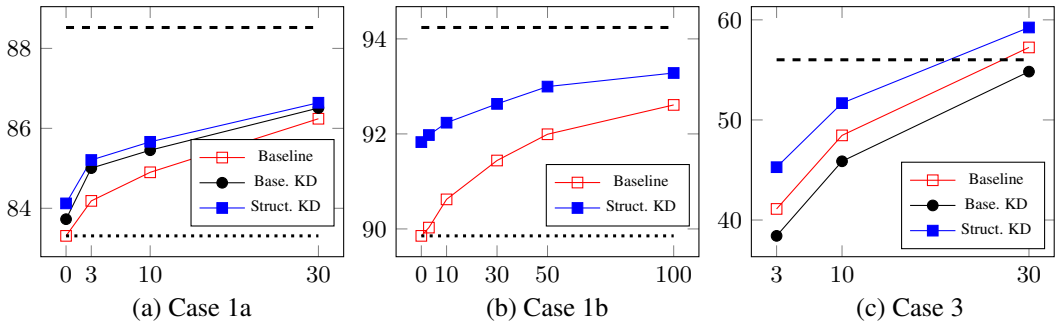
We report results averaged over 5 runs on four languages (except for **Case 1b** and **2b** where the datasets are monolingual) in Table 1. For experiments with unlabeled data, we only report results with 3000 unlabeled sentences. For more detailed results, please refer to the Appendix B.

The results show that our structural KD approaches outperform the **Baseline** and **Base. KD** in all the cases. We also make the following additional observations. The graph-based NER-as-parsing model (**Case 4**) is a less effective teacher than the linear-chain CRF model (**Case 2a**) on all the datasets. It is probably because of the significant difference in the modeling strategies between NER as parsing and NER as sequence labeling. In **Case 3**, the token-level KD baseline even underperforms the **Baseline**, probably because only transferring token-level knowledge is inadequate for the linear-chain CRF student that needs to model label transitions.

5 Analysis

5.1 Amount of Unlabeled Data

We compare our approaches with the baselines with different amounts of unlabeled data for **Case 1a**, **1b** and **3**, which are cases that apply in-domain unlabeled data for NER and dependency parsing, and cross-lingual unlabeled data for NER. We experiment with more unlabeled data for **Case 1b** than for the other two cases because the labeled training data of PTB is more than 10 times larger than the labeled NER training data in **Case 1b** and **3**. Results are shown in Figure 1. Note that we use teacher's prediction on unlabeled data to train the students as explained in Section 4.1, so even the **Baseline** can benefit from unlabeled data. It can be seen from the experimental results that our approaches consistently outperform the baselines, though the performance gaps between them become smaller when the amount of unlabeled data increases. Comparing the performance of the students with the teachers, we can see that in **Case 1a** and **1b**, the gap between the teacher and the student remains large even with the largest amount of unlabeled data. This is unsurprising considering the difference in model capacity between the teacher and the student. In **Case 3**, however, we find that when using 30,000 unlabeled sentences, the CRF student models can even outperform the MaxEnt teacher model, which shows the effectiveness of CRF models on NER.



(a) Case 1a (b) Case 1b (c) Case 3

Figure 1: The accuracy of structural KD and the baselines on different amounts of unlabeled data in three cases. The x-axis represents the amount of unlabeled data in thousand and the y-axis represents the accuracy. The dashed lines are the accuracy of the teacher models. The dotted lines are the accuracy of the baseline models without any knowledge from the teachers.

Table 2: Comparison of the global and local temperature application approaches on the CoNLL NER datasets.

	de	en	es	nl	Avg.
CRF	75.37	91.21	86.55	85.67	84.70
Global	75.67	91.11	86.72	85.92	84.85
Local	**76.61**	**91.41**	**87.20**	**86.19**	**85.35**

Table 3: Running speed and model sizes of the teacher and student models in **Case 2a**.

	Speed (sentences/second)	# Param (M)
Teacher	27.76	233.40
Student	672.2	9.46

Table 4: Averaged F1 score of teachers and it's marginal distributions. Mrg.: Marginal distribution, NER-Par.: NER as parsing (Yu et al., 2020).

	CoNLL	WikiAnn
CRF	**89.15**	**88.52**
CRF-Mrg.	89.08	88.41
NER-Par.	88.57	88.38
NER-Par.-Mrg.	87.40	86.82
MaxEnt	88.65	87.41

5.2 Temperature in Structural Knowledge Distillation.

A frequently used KD technique is dividing the logits of probability distributions of both the teacher and the student by a temperature in the KD objective (Hinton et al., 2015). Using a higher temperature produces softer probability distributions and often results in higher KD accuracy. In structural KD, there are two approaches to applying the temperature to the teacher model, either globally to the logit of $P_t(\boldsymbol{y}|\boldsymbol{x})$ (i.e., $\text{Score}_t(\boldsymbol{y}, \boldsymbol{x})$) of the full structure $\boldsymbol{y}$, or locally to the logit of $P_t(\boldsymbol{u}|\boldsymbol{x})$ of each student substructure $\boldsymbol{u}$. We empirically compare these two approaches in **Case 1a** with the same setting as in Section 4.1. Table 2 shows that the local approach results in better accuracy for all the languages. Therefore, we use the local approach by default in all the experiments.

5.3 Comparison of Teachers

In **Case 2a** and **Case 4**, we use the same MaxEnt student model but different types of teacher models. Our structural KD approaches in both cases compute the marginal distribution $P_t(y_i|\boldsymbol{x})$ of the teacher at each position i following the substructures of the MaxEnt student, which is then used to train the student substructure scores. We can evaluate the quality of the marginal distributions by taking their modes as label predictions and evaluating their accuracy. In Table 4, we compare the accuracy of the CRF teacher and its marginal distributions from **Case 2a**, the NER-as-parsing teacher and its marginal distributions from **Case 4**, and the MaxEnt teacher which is the KD baseline in **Case 2a**. First, we observe that for both CRF and NER-as-parsing, predicting labels from the marginal distributions leads to lower accuracy. This is to be expected because such predictions do not take into account correlations between adjacent labels. While predictions from marginal distributions of the CRF teacher still outperform MaxEnt, those of the NER-as-parsing teacher clearly underperform MaxEnt. This provides an explanation as to why structural KD in **Case 4** has equal or even lower accuracy than the KD baseline in **Case 2a**.

5.4 Comparison of Speed and Model Size

An important goal of KD is to produce faster and smaller models. In Table 3, we show a comparison on the running speed and model size between the teacher and student models on the CoNLL English test set from **Case 2a**. It can be seen that the student model is about 24 times faster and 25 times smaller than the teacher model.

6 Related Work

6.1 Structured Prediction

In this paper, we use sequence labeling and dependency parsing as two example structured prediction tasks. In sequence labeling, a lot of work applied the linear-chain CRF and achieved state-of-the-art performance in various sequence labeling tasks (Ma & Hovy, 2016; Akbik et al., 2018; Liu et al., 2019b; Yu et al., 2020; Wei et al., 2020). Meanwhile, a lot of other work used the MaxEnt layer instead of the CRF for sequence labeling (Devlin et al., 2019; Conneau et al., 2020) because MaxEnt makes it easier to fine-tune pretrained contextual embeddings in training. Another advantage of MaxEnt in comparison with CRF is its speed. Yang et al. (2018) showed that models equipped with the CRF are about two times slower than models with the MaxEnt layer in sequence labeling. In dependency parsing, recent work shows that second-order CRF parsers achieve significantly higher accuracy than first-order parsers (Wang et al., 2019; Zhang et al., 2020). However, the inference speed of second-order parsers is much slower. Zhang et al. (2020) showed that second-order parsing is four times slower than the simple head-selection first-order approach (Dozat & Manning, 2017). Such speed-accuracy tradeoff as seen in sequence labeling and dependency parsing also occurs in many other structured prediction tasks. This makes KD an interesting and very useful technique that can be used to circumvent this tradeoff to some extent.

6.2 Knowledge Distillation in Structured Prediction

KD has been applied in many structured prediction tasks in the fields of NLP, speech recognition and computer vision, with applications such as neural machine translation (Kim & Rush, 2016; Tan et al., 2019), sequence labeling (Wang et al., 2020), speech recognition (Huang et al., 2018), image semantic segmentation (Liu et al., 2019a) and so on. In KD for structured prediction tasks, how to handle the exponential number of structured outputs is a main challenge. To address this difficult problem, recent work resorts to approximation of the KD objective. Kim & Rush (2016) proposed sequence-level distillation through predicting K-best sequences of the teacher in neural machine translation. Kuncoro et al. (2016) proposed to use multiple greedy parsers as teachers and generate the probability distribution at each position through voting. Very recently, Wang et al. (2020) proposed structure-level knowledge distillation for linear-chain CRF models in multilingual sequence labeling. During the distillation process, teacher models predict the top-K label sequences as the global structure information or the posterior label distribution at each position as the local structural information, which is then used to train the student. Besides approximate approaches, an alternative way is using models that make local decisions and performing KD on these local decisions. Anderson & Gómez-Rodríguez (2020) formulated dependency parsing as a head-selection problem and distilled the distribution of the head node at each position. Tsai et al. (2019) proposed MiniBERT through distilling the output distributions of M-BERT models of the MaxEnt classifier. Besides the output distribution, Mukherjee & Hassan Awadallah (2020) further distilled the hidden representations of teachers.

7 Conclusion

In this paper, we propose structural knowledge distillation, which transfers knowledge between structured prediction models. We derive a factorized form of the structural KD objective and make it tractable to compute and optimize for many typical choices of teacher and student models. We apply our approach to four KD scenarios with six cases for sequence labeling and dependency parsing. Empirical results show that our approach outperforms baselines without KD as well as previous KD approaches. With unlabeled data, our approach can even boost the students to outperform the teachers in zero-shot cross-lingual transfer.

REFERENCES

Alan Akbik, Duncan Blythe, and Roland Vollgraf. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pp. 1638–1649, Santa Fe, New Mexico, USA, August 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/C18-1139.

Mark Anderson and Carlos Gómez-Rodríguez. Distilling neural networks for greener and faster dependency parsing. In *Proceedings of the 16th International Conference on Parsing Technologies and the IWPT 2020 Shared Task on Parsing into Enhanced Universal Dependencies*, pp. 2–13, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.iwpt-1.2. URL https://www.aclweb.org/anthology/2020.iwpt-1.2.

Jimmy Ba and Rich Caruana. Do deep nets really need to be deep? In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems 27*, pp. 2654–2662. Curran Associates, Inc., 2014. URL http://papers.nips.cc/paper/5484-do-deep-nets-really-need-to-be-deep.pdf.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017. ISSN 2307-387X.

Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, pp. 535–541, New York, NY, USA, 2006. ACM. ISBN 1-59593-339-5. doi: 10.1145/1150402.1150464. URL http://doi.acm.org/10.1145/1150402.1150464.

Kevin Clark, Minh-Thang Luong, Urvashi Khandelwal, Christopher D. Manning, and Quoc V. Le. BAM! born-again multi-task networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 5931–5937, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1595. URL https://www.aclweb.org/anthology/P19-1595.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 8440–8451, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.747. URL https://www.aclweb.org/anthology/2020.acl-main.747.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://www.aclweb.org/anthology/N19-1423.

Timothy Dozat and Christopher D Manning. Deep biaffine attention for neural dependency parsing. In *International Conference on Learning Representations*, 2017.

Timothy Dozat, Peng Qi, and Christopher D. Manning. Stanford's graph-based neural dependency parser at the CoNLL 2017 shared task. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pp. 20–30, Vancouver, Canada, August 2017. Association for Computational Linguistics. doi: 10.18653/v1/K17-3002. URL https://www.aclweb.org/anthology/K17-3002.

Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*, 2015. URL http://arxiv.org/abs/1503.02531.

Mingkun Huang, Yongbin You, Zhehuai Chen, Yanmin Qian, and Kai Yu. Knowledge distillation for sequence model. In *Proc. Interspeech 2018*, pp. 3703–3707, 2018. doi: 10.21437/Interspeech.2018-1589. URL http://dx.doi.org/10.21437/Interspeech.2018-1589.

Yoon Kim and Alexander M. Rush. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1317–1327, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1139. URL https://www.aclweb.org/anthology/D16-1139.

Adhiguna Kuncoro, Miguel Ballesteros, Lingpeng Kong, Chris Dyer, and Noah A. Smith. Distilling an ensemble of greedy dependency parsers into one MST parser. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1744–1753, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1180. URL https://www.aclweb.org/anthology/D16-1180.

Yifan Liu, Ke Chen, Chris Liu, Zengchang Qin, Zhenbo Luo, and Jingdong Wang. Structured knowledge distillation for semantic segmentation. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2599–2608. IEEE, 2019a.

Yijin Liu, Fandong Meng, Jinchao Zhang, Jinan Xu, Yufeng Chen, and Jie Zhou. GCDT: A global context enhanced deep transition architecture for sequence labeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 2431–2441, Florence, Italy, July 2019b. Association for Computational Linguistics. doi: 10.18653/v1/P19-1233. URL https://www.aclweb.org/anthology/P19-1233.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2018.

Xuezhe Ma and Eduard Hovy. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1064–1074, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1101. URL https://www.aclweb.org/anthology/P16-1101.

Xuezhe Ma, Zecong Hu, Jingzhou Liu, Nanyun Peng, Graham Neubig, and Eduard Hovy. Stack-pointer networks for dependency parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1403–1414, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1130. URL https://www.aclweb.org/anthology/P18-1130.

Subhabrata Mukherjee and Ahmed Hassan Awadallah. XtremeDistil: Multi-stage distillation for massive multilingual models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 2221–2234, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.202. URL https://www.aclweb.org/anthology/2020.acl-main.202.

Xiaoman Pan, Boliang Zhang, Jonathan May, Joel Nothman, Kevin Knight, and Heng Ji. Cross-lingual name tagging and linking for 282 languages. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1946–1958, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1178. URL https://www.aclweb.org/anthology/P17-1178.

Telmo Pires, Eva Schlinger, and Dan Garrette. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4996–5001, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1493. URL https://www.aclweb.org/anthology/P19-1493.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *The 5th Workshop on Energy Efficient Machine Learning and Cognitive Computing - NeurIPS 2019*, 2019.

Cicero D Santos and Bianca Zadrozny. Learning character-level representations for part-of-speech tagging. In *Proceedings of the 31st international conference on machine learning (ICML-14)*, pp. 1818–1826, 2014.

Michalina Strzyz, David Vilares, and Carlos Gómez-Rodríguez. Viable dependency parsing as sequence labeling. 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)*, pp. 717–723, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1077. URL https://www.aclweb.org/anthology/N19-1077.

Xu Tan, Yi Ren, Di He, Tao Qin, and Tie-Yan Liu. Multilingual neural machine translation with knowledge distillation. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=S1gUsoR9YX.

Raphael Tang, Yao Lu, Linqing Liu, Lili Mou, Olga Vechtomova, and Jimmy Lin. Distilling task-specific knowledge from bert into simple neural networks. *arXiv preprint arXiv:1903.12136*, 2019.

Erik F. Tjong Kim Sang. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*, 2002. URL https://www.aclweb.org/anthology/W02-2024.

Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pp. 142–147, 2003. URL https://www.aclweb.org/anthology/W03-0419.

Henry Tsai, Jason Riesa, Melvin Johnson, Naveen Arivazhagan, Xin Li, and Amelia Archer. Small and practical BERT models for sequence labeling. 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)*, pp. 3632–3636, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1374. URL https://www.aclweb.org/anthology/D19-1374.

Xinyu Wang, Jingxian Huang, and Kewei Tu. Second-order semantic dependency parsing with end-to-end neural networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4609–4618, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1454. URL https://www.aclweb.org/anthology/P19-1454.

Xinyu Wang, Yong Jiang, Nguyen Bach, Tao Wang, Fei Huang, and Kewei Tu. Structure-level knowledge distillation for multilingual sequence labeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 3317–3330, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.304. URL https://www.aclweb.org/anthology/2020.acl-main.304.

Zhenkai Wei, Yu Hong, Bowei Zou, Meng Cheng, and Jianmin Yao. Don't eclipse your arts due to small discrepancies: Boundary repositioning with a pointer network for aspect extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 3678–3684, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.339. URL https://www.aclweb.org/anthology/2020.acl-main.339.

Shijie Wu and Mark Dredze. Beto, bentz, becas: The surprising cross-lingual effectiveness of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 833–844, Hong Kong, China, November 2019. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D19-1077.

Jie Yang, Shuailong Liang, and Yue Zhang. Design challenges and misconceptions in neural sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pp. 3879–3889, Santa Fe, New Mexico, USA, August 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/C18-1327.

Juntao Yu, Bernd Bohnet, and Massimo Poesio. Named entity recognition as dependency parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 6470–6476, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.577. URL https://www.aclweb.org/anthology/2020.acl-main.577.

Yu Zhang, Zhenghua Li, and Min Zhang. Efficient second-order TreeCRF for neural dependency parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 3295–3305, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.302. URL https://www.aclweb.org/anthology/2020.acl-main.302.

A Dynamic Programming for Case 4

We describe how the marginal distribution over BIOES labels at each position of the input sentence can be tractably computed based on the NER-as-parsing teacher model using dynamic programming.

Given an input sentence $\boldsymbol{x}$ with n words, we first define the following functions.

- $\overrightarrow{\text{DP}}(i, l)$ represents the summation of scores of all possible labeling sequences of the sub-sentence from the first token to the i-th token while a span ends with the i-th token with a label l.
- $\overrightarrow{\text{DP}}(i, \text{F})$ represents the summation of scores of all possible labeling sequences of the sub-sentence from the first token to the i-th token while there is no arc pointing to the i-th token.
- $\overleftarrow{\text{DP}}(i, l)$ represents the summation of scores of all possible labeling sequences of the sub-sentence from the i-th toke to the last token while a span starts with the i-th token with a label l.
- $\overleftarrow{\text{DP}}(i, \text{F})$ represents the summation of scores of all possible labeling sequences of the sub-sentence from the i-th toke to the last token while there is no arc coming from the i-th token.

We can compute the values of these functions for all values of i and l using dynamic programming. The base cases are:

$$\overrightarrow{\text{DP}}(1, \text{F}) = 1 \qquad\qquad \overleftarrow{\text{DP}}(n, \text{F}) = 1$$

The recursive formulation of these functions are:

$$\overrightarrow{\text{DP}}(i, l) = \sum_{k=1}^{i} \exp(\text{Score}(y_{k,i} = l) * \overrightarrow{\text{DP}}(k, \text{F}) \qquad \overrightarrow{\text{DP}}(i, \text{F}) = \overrightarrow{\text{DP}}(i-1, \text{F}) + \sum_{l \in L} \overrightarrow{\text{DP}}(i-1, l)$$

$$\overleftarrow{\text{DP}}(i, l) = \sum_{j=i}^{n} \exp(\text{Score}(y_{i,j} = l) * \overleftarrow{\text{DP}}(j, \text{F}) \qquad \overleftarrow{\text{DP}}(i, \text{F}) = \overleftarrow{\text{DP}}(i+1, \text{F}) + \sum_{l \in L} \overleftarrow{\text{DP}}(i+1, l)$$

where $\text{Score}(y_{i,j} = l)$ is the score assigned by the teacher model to the dependency arc from i to j with label l.

After dynamic programming, we can compute the substructure marginals of the teacher $P_t(y_i|\boldsymbol{x})$ as follows:

$$P_t(y_i = B_l|\boldsymbol{x}) = \text{DP}(B_l, i)/\mathcal{Z}(\boldsymbol{x}) = \overrightarrow{\text{DP}}(i, \text{F}) * \sum_{j=i+1}^{n} \exp(\text{Score}(y_{i,j} = l)) * \overleftarrow{\text{DP}}(j, \text{F})/\mathcal{Z}(\boldsymbol{x})$$

$$P_t(y_i = I_l|\boldsymbol{x}) = \text{DP}(I_l, i)/\mathcal{Z}(\boldsymbol{x}) = \sum_{k=1}^{i-1} \sum_{j=i+1}^{n} \exp(\text{Score}(y_{k,j} = l)) * \overrightarrow{\text{DP}}(k, \text{F}) * \overleftarrow{\text{DP}}(j, \text{F})/\mathcal{Z}(\boldsymbol{x})$$

$$P_t(y_i = E_l|\boldsymbol{x}) = \text{DP}(E_l, i)/\mathcal{Z}(\boldsymbol{x}) = \overleftarrow{\text{DP}}(i, \text{F}) * \sum_{k=1}^{i-1} \exp(\text{Score}(y_{k,i} = l)) * \overrightarrow{\text{DP}}(k, \text{F})/\mathcal{Z}(\boldsymbol{x})$$

$$P_t(y_i = O|\boldsymbol{x}) = \text{DP}(O, i)/\mathcal{Z}(\boldsymbol{x}) = \overrightarrow{\text{DP}}(i, \text{F}) * \overleftarrow{\text{DP}}(i, \text{F})/\mathcal{Z}(\boldsymbol{x})$$

$$P_t(y_i = S_l|\boldsymbol{x}) = \text{DP}(S_l, i)/\mathcal{Z}(\boldsymbol{x}) = \overrightarrow{\text{DP}}(i, \text{F}) * \exp(\text{Score}(y_{i,i} = l)) * \overleftarrow{\text{DP}}(i, \text{F})/\mathcal{Z}(\boldsymbol{x})$$

where

- $\text{DP}(X, i)$ represents the summation of scores of all possible labeling sequences in which the i-th token is labeled as X. X can be one of 'B_l, I_l, E_l, O, S_l'.
- $\mathcal{Z}(\boldsymbol{x})$ represents the summation of scores of all possible labeling sequences given the input sentence $\boldsymbol{x}$. $y_{i,j} = l$ represents that there is a dependency arc of label 'l' from the i-th word to the j-th word.

The edge cases are:

$$P_t(y_n = B_l|\boldsymbol{x}) = 0 \qquad P_t(y_1 = I_l|\boldsymbol{x}) = P_t(y_n = I_l|\boldsymbol{x}) = 0 \qquad P_t(y_1 = E_l|\boldsymbol{x}) = 0$$

B Detailed Experimental Results

In this section, we present detailed experimental results. Table 5, 6 and 7 show the results of NER task, while table 8 and 9 show the results of Parsing. In tables, we use † to represent our approaches are significantly stronger than the Baselines and ‡ to represent our approaches are significantly stronger than Baseline KD approaches (with a significance level of 0.05).

B.1 Results of NER task

Table 5 and 6 represent the KD results of experiments with labeled and unlabeled datasets and Structural KD outperform baseline on all these cases.

On labeled dataset, structural KD outperform KD baseline in 6 out of 8 cases in scenario of distlling from liner-chain CRF to linear-chain CRF model (**Case 1a**) , 7 out of 8 cases in scenario of distilling from linear-chain CRF to MaxEnt model (**Case 2a**) and 8 out of 8 cases when distilling from graph-based NER model to MaxEnt model (**Case 4**).

On unlabeled dataset, structural KD outperform KD baseline in 10 out of 12 cases in scenario of distlling from liner-chain CRF to linear-chain CRF model (**Case 1a**), 8 out of 12 cases in scenario of distilling from linear-chain CRF to MaxEnt model (**Case 2a**) and 11 out of 12 cases when distilling from graph-based NER model to MaxEnt model (**Case 4**).

For zero shot transfer experiments (**Case 3**), structural KD outperform KD baseline in all 12 cases.

B.2 Results of Parsing task

Tabel 8 and 9 represent the results of experiments of Parsing. Structural KD outperforms KD baselines in all cases. UAS and LAS in these tables were dependency parsing metrics, and they refer to unlabeled attachment score and labeled attachment score respectively.

Table 5: Results of F1 scores for NER task on labeled datasets

Dataset		CoNLL					WikiAnn				
Scenario		de	en	es	nl	Avg.	de	en	es	nl	Avg.
Case 1a	Teacher	83.48	92.25	89.29	91.56	89.15	86.98	83.80	91.85	91.46	88.52
	Baseline	75.37	91.21	86.55	85.67	84.70	80.12	80.09	85.84	87.19	83.31
	Base. KD	76.46	91.38	**87.33**	85.92	85.27	80.02	**81.76**	85.98	87.15	83.73
	Structural KD	**76.61**†	**91.41**†	87.20†	**86.19**†	**85.35**	**80.64**†	81.37	**87.29**†‡	**87.19**	**84.12**
Case 2a	CRF Teacher	83.48	92.25	89.29	91.56	89.15	86.98	83.80	91.85	91.46	88.52
	MaxEnt teacher	82.83	92.03	88.49	91.26	88.65	85.98	82.46	90.81	90.39	87.41
	Baseline	74.44	90.78	85.42	84.83	83.87	77.98	78.52	83.73	83.19	80.86
	Base. KD	75.08	90.95	85.88	85.10	84.25	78.40	**79.52**	84.92	85.50	82.09
	Structural KD	**75.41**†	**91.04**†	**86.25**†‡	**85.28**	**84.50**	**78.49**†	79.48†	**85.28**†	**85.66**†	**82.23**
Case 4	Teacher	82.38	92.41	88.77	90.72	88.57	86.96	83.11	91.41	92.05	88.38
	Baseline	74.44	90.78	85.42	84.83	83.87	77.98	78.52	83.73	83.19	80.86
	Structural KD	**74.90**†	**91.21**†	**85.82**†	**85.20**	**84.28**	**78.66**	**78.97**	**83.83**	**83.34**†	**81.45**

Table 6: Results of F1 scores for NER task on unlabeled datasets

Dataset			WikiAnn with unlabeled sentences				
Scenario		# Unlabeled sent.	de	en	es	nl	avg
Case 1a	Teacher		86.98	83.80	91.85	91.46	88.52
	Baseline	3K	80.66	79.85	87.79	88.44	84.19
	Base. KD	3K	81.56	**81.40**	88.10	88.55	84.91
	Stuctural KD	3K	**81.88**†	81.23†	**88.66**†	**89.20**†‡	**85.24**
	Baseline	10k	82.27	80.32	88.78	88.23	84.90
	Base. KD	10k	82.01	**81.53**	89.28	88.99	85.45
	Stuctural KD	10k	**82.34**	81.27†	**89.85**†‡	**89.19**†	**85.66**
	Baseline	30k	**84.20**	81.19	90.21	89.36	86.24
	Base. KD	30k	84.12	**82.56**	89.82	89.53	86.51
	Stuctural KD	30k	84.17	82.14†	**90.41**‡	**89.84**†	**86.64**
Case 2a	Teacher		86.98	83.80	91.85	91.46	88.52
	Baseline	3K	78.82	78.48	85.54	86.77	82.40
	Base. KD	3K	**79.84**	79.18	85.89	87.36	83.07
	Stuctural KD	3K	79.82†	**79.41**†	**86.36**†	**87.75**†	**83.34**
	Baseline	10k	80.75	78.53	86.93	87.30	83.38
	Base. KD	10k	80.71	79.23	**87.82**	87.80	83.89
	Stuctural KD	10k	**81.07**	**79.41**†	87.77†	**87.99**†	**84.06**
	Baseline	30k	82.49	79.43	88.78	88.74	84.86
	Base. KD	30k	82.35	80.42	**89.32**	**88.84**	85.23
	Stuctural KD	30k	**83.06**‡	**80.43**†	89.02	88.62	**85.28**
Case 4	Teacher		86.96	83.11	91.41	92.05	88.38
	Baseline	3K	78.41	77.22	85.82	86.94	82.10
	Stuctural KD	3K	**78.80**	**78.00**†	**85.75**	**87.22**	**82.44**
	Baseline	10k	79.59	77.53	87.85	**87.51**	83.12
	Stuctural KD	10k	**80.04**	**78.06**†	**88.03**	87.40	**83.38**
	Baseline	30k	81.47	78.59	89.46	88.80	84.58
	Stuctural KD	30k	**81.85**	**79.57**†	**89.55**	**89.13**	**85.03**

Table 7: Result of F1 scores of zero shot transfer experiment on NER task

		WikiAnn				
Case 3	# Unlabeled sent.	eu	fa	he	ta	Avg.
Teacher		67.92	40.30	58.68	57.14	56.01
Baseline	3k	41.77	37.88	41.32	43.46	41.11
Base. KD		52.67	26.32	36.22	38.45	38.42
Structural KD		**53.69**†	**42.02**‡	**42.75**‡	**42.66**‡	**45.28**
Baseline	10k	58.63	34.65	43.37	57.18	48.46
Base. KD		58.87	28.63	41.62	54.35	45.87
Structural KD		**62.50**†‡	**39.72**†‡	**46.22**‡	**58.27**‡	**51.68**
Baseline	30k	74.37	35.70	55.12	63.78	57.24
Base. KD		70.98	29.44	55.50	63.39	54.83
Structural KD		**75.66**‡	**38.08**†‡	**58.52**†‡	**64.69**†‡	**59.24**

Table 8: Result of F1 scores of Parsing task with labeled dataset. Note that all our approaches are significantly stronger than the baseline.

metric		**Case 1b**	**Case 2b**
		PTB	PTB
UAS	Teacher	95.96	96.04
	Baseline	91.78	91.78
	Structural KD	**93.56**	**93.56**
LAS	Teacher	94.24	94.29
	Baseline	89.85	89.85
	Structural KD	**91.83**	**91.78**

Table 9: The accuracy of Parsing task with unlabeled dataset (in thousand). Note that all our approaches are significantly stronger than the baseline.

		Case 1b						**Case 2b**					
		PTB + Bllip						PTB + Bllip					
Metric		3k	10k	30k	50k	100k	Avg.	3k	10k	30k	50k	100k	Avg.
UAS	Baseline	92.00	92.52	93.22	93.69	94.25	93.14	91.99	92.44	93.16	93.69	94.26	93.11
	Struct. KD	**93.71**	**93.93**	**94.26**	**94.58**	**94.84**	**94.26**	**93.67**	**93.90**	**94.30**	**94.64**	**94.89**	**94.28**
LAS	Baseline	90.03	90.62	91.44	91.99	92.61	91.34	90.03	90.59	91.41	91.98	92.66	91.33
	Struct. KD	**91.98**	**92.24**	**92.63**	**93.00**	**93.28**	**92.63**	**91.94**	**92.18**	**92.66**	**93.04**	**93.31**	**92.63**