Coupled POS Tagging on Heterogeneous Annotations

Zhenghua Li, Jiayuan Chao, Min Zhang, Wenliang Chen, Meishan Zhang, and Guohong Fu

***Abstract*—The limited scale and genre coverage of labeled data greatly hinders the effectiveness of supervised models, especially when analyzing spoken languages, such as texts transcribed from speech and informal text including tweets and product comments in Internet. In order to effectively utilize multiple labeled datasets with heterogeneous annotations for the same task, this paper proposes a coupled sequence labeling model that can directly learn and infer two heterogeneous annotations simultaneously, using Chinese part-of-speech (POS) tagging as our case study. The key idea is to bundle two sets of POS tags together (e.g., "$[NN, n]$"), and build a conditional random field (CRF) based tagging model in the enlarged space of bundled tags with the help of *ambiguous labeling*. To train our model on two nonoverlapping datasets that each has only one-side tags, we transform a one-side tag into a set of bundled tags by concatenating the tag with every possible tag at the missing side according to a predefined context-free tag-to-tag mapping function, thus producing ambiguous labeling as weak supervision. We design and investigate four different context-free tag-to-tag mapping functions, and find out that the coupled model achieves its best performance when each one-side tag is mapped to all tags at the other side (namely *complete mapping*), indicating that the model can effectively learn the loose mapping between the two heterogeneous annotations, without the need of manually designed mapping rules. Moreover, we propose a context-aware online pruning strategy that can more accurately capture mapping relationships between annotations based on contextual evidences and thus effectively solve the severe inefficiency problem with our coupled model under complete mapping, making it comparable with the baseline CRF model. Experiments on benchmark datasets show that our coupled model significantly outperforms the state-of-the-art baselines on both one-side POS tagging and annotation conversion tasks. The codes and newly annotated data are released for research usage.[1]**

***Index Terms*—Part-of-speech tagging, coupled sequence labeling, heterogeneous annotations.**

Manuscript received February 18, 2016; revised June 12, 2016, July 24, 2016, and November 14, 2016; accepted December 12, 2016. Date of publication December 22, 2016; date of current version January 26, 2017. This work was supported by the National Natural Science Foundation of China under Grant 61525205, Grant 61572338, and Grant 61432013, and was also sponsored by CCF-Tencent Open Research Fund (AGR20160111). The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Imed Zitouni. *(Corresponding author: Min Zhang.)*

Z. Li, J. Chao, M. Zhang, and W. Chen are with the School of Computer Science and Technology, Soochow University, Suzhou 215006, China (e-mail: zhli13@suda.edu.cn; china_cjy@163.com; minzhang@suda.edu.cn; wlchen@suda.edu.cn).

M. Zhang and G. Fu are with the School of Computer Science and Technology, Heilongjiang University, Harbin 150080, China (e-mail: mason.zms@gmail.com; ghfu@hotmail.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TASLP.2016.2644262

[1]http://hlt.suda.edu.cn/∼zhli

I. Introduction

Given an input sentence of n words, denoted by $\mathbf{x} = w_1 \dots w_n$, POS tagging aims to predict a tag sequence $\mathbf{t} = t_1 \dots t_n$, where $t_i \in \mathcal{T}$ $(1 \leq i \leq n)$ and $\mathcal{T}$ is a predefined tag set. POS tags are designed to represent word classes so that words with the same POS tag play a similar morphological or syntactic role in language usage [1]. As the most fundamental task for text analysis, POS tagging is crucial for many high-level tasks such as named entity recognition (NER) [2], syntactic parsing [3], and information extraction [4]. In particular, studies on dependency parsing demonstrate that POS tag-based features are key for alleviating the data sparseness problem of pure lexical features [5], [6]. However, due to the lack of morphological clues, Chinese POS tagging turns out to be much more challenging than other morphologically-richer languages such as English. The state-of-the-art POS tagging accuracy is about 94% for Chinese, which is much lower than 97% for English.

Another factor that brings down POS tagging accuracy is the data sparseness problem due to limited scale and genre coverage of labeled data. Due to the heavy cost of manual annotation, labeled data is usually limited in both scale and genre, significantly hindering the effectiveness of supervised statistical models in processing real texts, especially when analyzing texts transcribed from speech and informal text such as tweets and product comments in Internet. As a typical sequence labeling task, part-of-speech (POS) tagging is more prone to suffer from the data sparseness issue than binary or multi-class classification problems. Following standard practice, this work builds our approach based on a linear-chain conditional random field (CRF) model [7].

To alleviate the data sparseness problem due to limited scale of labeled data, semi-supervised learning has been extensively studied for POS tagging as a promising research line. [8] show that standard self-training can boost the performance of a simple HMM based POS tagger. [9] apply tri-training to English POS tagging, boosting accuracy from 97.27% to 97.50%. [10] derive word clusters from large-scale unlabeled data as extra features for Chinese POS tagging. Recently, the use of natural annotations becomes a hot topic in sequence labeling problems, especially in Chinese word segmentation [11]–[13]. The idea is to derive segmentation boundaries from implicit information encoded in web texts, such as anchor texts and punctuation marks, and use them as partially labeled training data in sequence labeling models.

This work follows another research line on utilizing multiple labeled data with heterogeneous annotations for alleviating data sparseness. For example, Penn Chinese Treebank (CTB), a widely used benchmark data, contains about 20 thousand

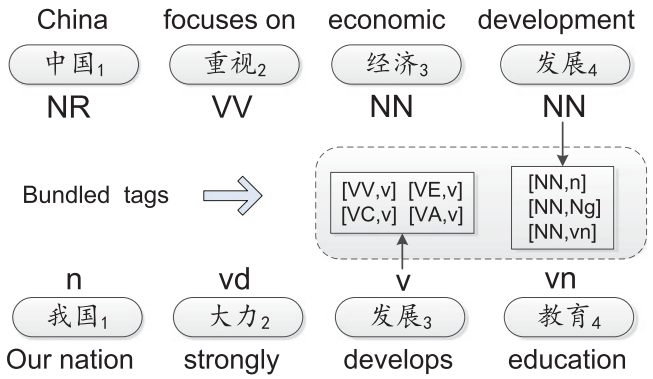


Fig. 1. An example to illustrate the annotation differences between CTB (above) and PD (below), and how to transform a one-side tag into a set of bundled tags.

sentences annotated with word boundaries, POS tags, and syntactic structures [14], [15]. People's Daily corpus $(PD)^2$ is a large-scale corpus annotated with words and POS tags, containing about 300 thousand sentences from the first half of 1998 of People's Daily newspaper (see Table II) [16]. It is a promising idea to learn better models from both resources. However, the key challenge of exploiting the two resources is that they are non-overlapping and adopt different sets of POS tags which are impossible to be precisely converted from one to another based on heuristic rules.[3] Fig. 1 shows two example sentences from CTB and PD. We discuss the differences between the annotation guidelines of the two datasets in Section III.

Previous work on exploiting heterogeneous data (CTB and PD) mainly focuses on indirect guide-feature methods. The basic idea is to use one resource to generate extra guide features on another resource [18], [19], which is similar to stacked learning [20]. First, PD is used as source data to train a source model $Tagger_{PD}$. Then, $Tagger_{PD}$ generates automatic POS tags on the target data CTB, called *source annotations*. Finally, a target model $Tagger_{CTB\text{-}guided}$ is trained on CTB, using source annotations as extra guide features. Although the guide-feature based method is effective in boosting performance of the target model, we argue that it may have two potential drawbacks. First, the target model $Tagger_{CTB\text{-}guided}$ does not directly use PD as training data, and therefore fails to make full use of rich language phenomena in PD. Second, the method is more complicated in real applications since it needs to parse a test sentence twice to get the final results.

This paper proposes a coupled sequence labeling model that directly learns and infers two heterogeneous annotations simultaneously, using Chinese POS tagging as a case study. The key idea is to bundle two sets of POS tags together (e.g. "$[NN, n]$"), and build a CRF based tagging model in the enlarged space of bundled tags. To make use of two non-overlapping datasets that each has only one-side tags, we transform a one-side tag into a set of bundled tags by concatenating the tag with every possible tag at the missing side according to some predefined context-free tag-to-tag mapping rules. After such transformation, each sentence has an exponential-size set of bundled tag sequences as weak supervision, which is known as *ambiguous labeling* [21]–[23]. After a straightforward extension, the CRF-based coupled model can be naturally trained using such ambiguous labeling. In summary, this work makes the following contributions:

1) We propose a simple and effective coupled sequence labeling model for utilizing multiple heterogeneous labeled data. Experiments show that the coupled model is able to learn the implicit loose mappings between heterogeneous annotations.
2) We also propose a context-aware online pruning strategy that can effectively solve the severe inefficiency problem with the coupled model, making it comparable with the baseline CRF model on training and inference speed.
3) Experiments show our approach significantly outperforms the baseline and guide-feature models on one-side POS tagging accuracy of both CTB and *PD*.
4) We have manually annotated CTB tags for 1,000 PD sentences. Experiments on the data show that our coupled model significantly outperforms the baseline and guide-feature models on the annotation conversion task by a large margin.

This paper has substantially extended our preliminary work [24] in the following perspectives:

1) Most importantly, this work proposes a context-aware online pruning strategy that can effectively improve both training and inference efficiency of the coupled model with little accuracy loss (Section IV-C).
2) Based on the newly proposed pruning strategy, we redesign the experimental settings with a more reasonable data split. We rerun all experiments and report totally new results including accuracies on *PD*, with extensive analysis of different models.
3) This paper presents a more comprehensive description of the models and algorithms.
4) Section III-A gives a systematic and detailed discussion on the differences betwen *CTB* and *PD* annotations.

II. Traditional POS Tagging ($Tagger_{CTB}$ and $Tagger_{PD}$)

Typically, POS tagging is treated as a sequence labeling problem and has been previously addressed by machine learning algorithms such as maximum entropy [25], CRF [26], and perceptron [27]. In this work, our proposed coupled model is a direct extension of a traditional linear-chain CRF. To better illustrate our coupled model, this section describes the traditional CRF based POS tagging in detail.

As a log-linear probabilistic model [26], [28], CRF defines the probability of a tag sequence as:

$$p(\mathbf{t}|\mathbf{x};\theta) = \frac{e^{Score(\mathbf{x},\mathbf{t};\theta)}}{Z(\mathbf{x};\theta)} \tag{1}$$

where θ is the model parameters, a.k.a. the feature weights; $Z(\mathbf{x};\theta)$ is the normalization factor which sums over all

[2]http://icl.pku.edu.cn/icl_groups/corpustagging.asp

[3]As commented by a reviewer, we may build a model to learn the mapping rules if heterogeneous annotations are available on the same texts. However, heterogeneous data are usually non-overlapping. As one exception, the "Corpus of Spontaneous Japanese" contains two layers of word segments and POS tags of different granularities [17].

TABLE I
POS TAGGING FEATURE TEMPLATES FOR $\mathbf{f}(\mathbf{x}, i, t', t)$

01: $t \circ t'$	02: $t \circ w_i$
03: $t \circ w_{i-1}$	04: $t \circ w_{i+1}$
05: $t \circ w_i \circ c_{i-1,-1}$	06: $t \circ w_i \circ c_{i+1,0}$
07: $t \circ c_{i,0}$	08: $t \circ c_{i,-1}$
09: $t \circ c_{i,k}, 0 < k < \#c_i - 1$	
10: $t \circ c_{i,0} \circ c_{i,k}, 0 < k < \#c_i - 1$	
11: $t \circ c_{i,-1} \circ c_{i,k}, 0 < k < \#c_i - 1$	
12: **if** $\#c_i = 1$ **then** $t \circ w_i \circ c_{i-1,-1} \circ c_{i+1,0}$	
13: **if** $c_{i,k} = c_{i,k+1}$ **then** $t \circ c_{i,k} \circ "consecutive"$	
14: $t \circ \text{prefix}(w_i, k), 1 \le k \le 4, k \le \#c_i$	
15: $t \circ \text{suffix}(w_i, k), 1 \le k \le 4, k \le \#c_i$	

TABLE II
DATA STATISTICS

		#sentences	#tokens w/ CTB tags	#tokens w/ PD tags
CTB	train	16,091	437,991	–
	dev	803	20,454	–
	test	1,910	50,319	–
PD	train	46,815	–	1,097,839
	train-large	253,722	–	5,936,076
	dev	2,000	–	46,182
	test	5,000	–	118,714
	conversion	1,000	5,769	27,942

possible sequences $\mathbf{t}$ in the search space in order to guarantee that probabilities of all sequences sum to one.

$$Z(\mathbf{x};\theta) = \sum_{\mathbf{t}} e^{Score(\mathbf{x},\mathbf{t};\theta)} \tag{2}$$

$Score(\mathbf{x}, \mathbf{t}; \theta)$ is the score of a specific tag sequence $\mathbf{t}$ given $\mathbf{x}$, which factorizes into bigram scoring parts according to the linear-chain Markovian assumption.[4]

$$\begin{aligned} Score(\mathbf{x},\mathbf{t};\theta) &= \theta \cdot \mathbf{f}(\mathbf{x},\mathbf{t}) \\ &= \sum_{i=1}^{n+1} \theta \cdot \mathbf{f}(\mathbf{x}, i, t_{i-1}, t_i) \end{aligned} \tag{3}$$

where $\mathbf{f}(\mathbf{x}, \mathbf{t})$ is the accumulated feature vector of $\mathbf{t}$ given $\mathbf{x}$. Based on the Markovian assumption, $\mathbf{f}(\mathbf{x}, \mathbf{t})$ is then factorized into bigram features for the sake of efficient inference. $\mathbf{f}(\mathbf{x}, i, t', t)$ is the bigram feature vector for tagging w_{i-1} as t' and w_i as t. $t_0 = START$ and $t_{n+1} = STOP$ are two pseudo tags which mark the start and end position of the sentence. We adopt the state-of-the-art tagging features in [5], [29], as shown in Table I, where $\circ$ means string concatenation; $c_{i,k}$ denotes the k^{th} Chinese character in the word w_i; $c_{i,0}$ is the first Chinese character; $c_{i,-1}$ is the last Chinese character; $\#c_i$ is the total number of Chinese characters contained in w_i; prefix/suffix(w_i, k) denote the k-character prefix/suffix of w_i.

Suppose the training data is $\mathcal{D} = \{(\mathbf{x}_j, \mathbf{t}_j)\}_{j=1}^N$, meaning that $\mathbf{t}_j$ is the gold-standard POS tag sequence of $\mathbf{x}_j$. Then the log likelihood of the training data is defined as follows.

$$\begin{aligned} \mathcal{L}(\mathcal{D};\theta) &= \sum_{j=1}^{N} \log p(\mathbf{t}_j|\mathbf{x}_j;\theta) \\ &= \sum_{j=1}^{N} (Score(\mathbf{x}_j,\mathbf{t}_j;\theta) - \log Z(\mathbf{x}_j;\theta)) \end{aligned} \tag{4}$$

To maximize this log likelihood function, we take its partial derivation with regard to θ.

$$\begin{aligned} \frac{\partial \mathcal{L}(\mathcal{D};\theta)}{\partial \theta} &= \sum_{j=1}^{N} \left(\mathbf{f}(\mathbf{x}_j,\mathbf{t}_j) - \frac{1}{Z(\mathbf{x}_j;\theta)} \frac{\partial Z(\mathbf{x}_j;\theta)}{\partial \theta} \right) \\ &= \sum_{j=1}^{N} \left(\mathbf{f}(\mathbf{x}_j,\mathbf{t}_j) - \sum_{\mathbf{t}} p(\mathbf{t}|\mathbf{x}_j;\theta) \mathbf{f}(\mathbf{x}_j,\mathbf{t}) \right) \\ &= \sum_{j=1}^{N} \left(\mathbf{f}(\mathbf{x}_j,\mathbf{t}_j) - E_{\mathbf{t}|\mathbf{x};\theta}[\mathbf{f}(\mathbf{x}_j,\mathbf{t})] \right) \end{aligned} \tag{5}$$

where $\mathbf{f}(\mathbf{x}_j, \mathbf{t}_j)$ is explained as the empirical counts of the features in the gold-standard reference $\mathbf{t}_j$, whereas $E_{\mathbf{t}|\mathbf{x};\theta}[\mathbf{f}(\mathbf{x}_j, \mathbf{t})]$ is the feature expectations for $\mathbf{x}_j$ under the current model θ. Naively computing the feature expectations requires enumerating an exponentially increased search space ($O(|\mathcal{T}|^n)$). Fortunately, the feature expectations can be further factorized according to the Markovian assumption defined in Eq. (3):

$$\begin{aligned} E_{\mathbf{t}|\mathbf{x};\theta}[\mathbf{f}(\mathbf{x},\mathbf{t})] &= \sum_{\mathbf{t}} p(\mathbf{t}|\mathbf{x};\theta) \left(\sum_{i=1}^{n+1} \mathbf{f}(\mathbf{x}, i, t_{i-1}, t_i) \right) \\ &= \sum_{\mathbf{t}} \left(\sum_{i=1}^{n+1} p(\mathbf{t}|\mathbf{x};\theta) \mathbf{f}(\mathbf{x}, i, t_{i-1}, t_i) \right) \\ &= \sum_{i=1}^{n+1} \left(\sum_{\mathbf{t}} p(\mathbf{t}|\mathbf{x};\theta) \mathbf{f}(\mathbf{x}, i, t_{i-1}, t_i) \right) \\ &= \sum_{i=1}^{n+1} \sum_{(t',t)\in\mathcal{T}^2} \mathbf{f}(\mathbf{x}, i, t', t) \\ &\quad \times \left(\sum_{\mathbf{t}: t_{i-1}=t', t_i=t} p(\mathbf{t}|\mathbf{x};\theta) \right) \\ &= \sum_{i=1}^{n+1} \sum_{(t',t)\in\mathcal{T}^2} \mathbf{f}(\mathbf{x}, i, t', t) p(i, t', t|\mathbf{x};\theta) \end{aligned} \tag{6}$$

where $p(i, t', t|\mathbf{x}; \theta)$ is known as marginal probability of a local bigram clique in Markov random field models, namely tagging w_{i-1} as t' and w_i as t. Using the classic *Forward-Backward* algorithm, we can compute all bigram marginal probabilities in polynomial time of $O(n|\mathcal{T}|^2)$. Given all marginal probabilities, we can then compute feature expectations in Eq. (6) in $O(n|\mathcal{T}|^2)$ time.

[4]Most previous work on POS tagging uses bigram features, since extra usage of trigram is much slower yet brings little accuracy improvement.

III. Mapping Between CTB/PD Annotations

This section describes the differences between the annotation guidelines of CTB and PD, and then proposes four different sets of mapping rules for converting a one-side tag into a set of bundled tags as ambiguous labeling.

A. Annotation Heterogeneity

Based on the Penn Chinese Treebank Project, CTB is built to create a Mandarin Chinese corpus with syntactic bracketing [15]. PD is annotated by Institute of Computational Linguistics at Peking University, with the purpose of building a large-scale corpus with word segmentation, POS tagging, and phonetic notations to facilitate Chinese information processing [16].

The most fundamental difference between CTB and PD is that they adopt different annotation principles. For a focus word, CTB uses the syntactic role that the word takes in the sentence as the main criterion for deciding its POS tags. In other words, a word can have different tags in different contexts. For example, the frequently used auxiliary word "的" is annotated according to its specific syntactic roles into four different tags: "DEG" (translated into "of", as possessive marker), "DEC" (translated into "that", as relative-clause marker), "SP" (as sentence-final particle), and "AS" (as aspect marker). Another example is "发展", which is tagged as "VV" (translated into "develop") or "NN" (translated into "development") according to the context in use. It is natural that CTB adopts such annotation principle, because in this way POS tags can best support higher-level syntactic annotations.

In contrast, based on our observations and some implicit discussions in the guideline document, it seems that PD adopts a quite different annotation principle. First, at the beginning, the annotation of syntactic structures is not part of the plan in *PD*. Therefore, the design of POS tags gives little thought for syntactic annotation. In other words, the POS tags are not intended to support future syntactic annotation. Function words are important elements in syntactic structures. However, *PD* usually assign a single tag to a specific function word, without distinguishing its different functions according to different contexts. Take aforementioned auxiliary word "的" as a concrete example, which is very important for recovering syntactic structures, only receives a single tag "u" (auxiliary word) in PD for all occurrences.[5]

Another important difference is that PD subdivides proper nouns into four different categories: "*nr*" (human names), "*ns*" (geographical names), "*nt*" (names of organization/group/team), and "nz" (other proper nouns). Contrarily, CTB only uses a single tag "NR". This indicates that PD is more suitable to support certain higher-level applications like information extraction which attaches a lot of importance to proper nouns.

[5]An important exception is that PD uses a specific tag "vn" to denote a verb used as a noun in a specific context. For example, "发展" is tagged as "v" (corresponding to "VV" in CTB) in its ordinary usage, and is tagged as "vn" (corresponding to "NN" in CTB) when used as a noun. In this sense, PD uses "n" to tag ordinary nouns and uses "vn" to tag verbs functioning as nouns, whereas CTB makes no such distinction and uses only "NN" for all nouns in the syntactic sense.

[15] summarizes rough correspondences between CTB and PD tags (See [15, Table B.3]). However, we find that it is very difficult to manually build a compact but complete set of mappings between the two sets of POS tags due to the following reasons.

1) CTB and PD sometimes have different criteria in word categorization. For example, "国务院" (translated into "State Council") is annotated as "nt" in PD, but is tagged as an ordinary noun "NN" rather than a proper noun "NR" in CTB. Similarly, PD treats "中文" (translated into "Chinese" as a language) as "nz", whereas CTB tags it as "NN". This turns many-to-one correspondences into many-to-many relationships.
2) There are always exceptions outside the designed tag-to-tag mapping rules, since the mapping relationships are in many cases word-sensitive or even context-sensitive. Moreover, there exist a lot of annotation errors in both CTB and PD. For example, many occurrences of "新华社" (translated into "Xinhua News Agency") are wrongly annotated as "NN" rather than its true tag "NR".
3) In fact, CTB and PD also differ a lot in word segmentation standard, which leads to many irregularities in mapping relationships. In this work, we leave this issue for future research due to space limitation. Generally speaking, PD usually takes coarser-grained word segmentations than CTB. For example, "这个" (translated into "this"), a frequently used string in both data, is treated as a single word with a tag "*r*" (pronoun) in PD, but is segmented into two words "这" and "个" tagged as "DT" (determiner) and "M" (measure word) respectively in CTB. As a result, it seems that no single CTB-tag is perfectly proper for this word "这个/*r*" due to granularity differences in words.

B. Context-Free Tag-to-Tag Mapping Functions

Based on the above analysis and discussion, we can see that building context-free tag-to-tag mapping between CTB and PD annotations can be very difficult. However, due to efficiency considerations, mapping one CTB tag to many PD tags (vice versa) would lead to a huge size of bundled tags and thus make the coupled model prohibitively slow. Therefore, with a lot effort, we have designed and investigated four different mapping functions in this work, which reveals valuable insights into the coupled model, and is considered as one of the main contributions of this work.

A mapping function is a set of symmetric mapping rules. Each mapping rule decides whether one CTB tag (i.e., "NN") can be mapped to one PD tag (i.e., "n"). Formally, we denote the tag set of CTB as $\mathcal{T}^a$, and that of PD as $\mathcal{T}^b$. Then a mapping function $\mathbf{m} : \mathcal{T}^a \times \mathcal{T}^b \rightarrow \{0, 1\}$ is defined as:

$$\mathbf{m}(t^a, t^b) = \begin{cases} 1 & \text{if the two tags can be mapped to each other} \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

where $t^a \in \mathcal{T}^a$ and $t^b \in \mathcal{T}^b$. Alternatively, we can think of a mapping function as a matrix with 0/1 elements.

Take Fig. 1 as an example. The word "发展 4" is manually tagged as "NN" at the CTB side. Suppose that the mapping function $\mathbf{m}$ tells that "NN" can be mapped into three tags at the PD side, i.e., "n", "Ng", and "vn". Then, we create three bundled tags for the word, i.e., "$[NN, n]$", "$[NN, Ng]$", "$[NN, vn]$" as its gold-standard references during training. It is known as *ambiguous labeling* when a training instance has multiple gold-standard labels. Similarly, we can obtain bundled tags for all other words in sentences of CTB and PD based on the predefined mapping function $\mathbf{m}$.

After such transformation from a one-side tag to a set of bundled tags, we can collect from data all possible bundled tags together, and compose a bundled tag space, denoted as $\mathcal{T}^{a\&b}$. Finally, the two non-overlapping heterogeneous datasets are now in the same bundled tag space, and each sentence has an exponential-size set of bundled tag sequences as gold-standard references during training. It is obvious that $\mathcal{T}^{a\&b} \subseteq \mathcal{T}^a \times \mathcal{T}^b$, and a mapping function actually defines a specific bundled tag space. When the mapping function becomes looser, meaning that the matrix $\mathbf{m}$ contains more 1s instead of 0s, the bundled tag space size $|\mathcal{T}^{a\&b}|$ becomes larger.

In the following, we describe the four context-free tag-to-tag mapping functions that we have designed and investigated in this work. The full list of mapping rules for each mapping function is also released for reference.[6]

1) The **tight** mapping function ($\mathbf{m}_{tight}$, 145 bundled tags). We spent about two hours on studying the annotation guidelines of both *CTB* and *PD*, and designed the tight mapping function which on the one hand produces the least number of bundled tags, and on the other hand follows as far as possible the mapping relationships described in the annotation guidelines, as summarized [15, Table B.3].
2) The **automatic** mapping function ($\mathbf{m}_{auto}$, 346 bundled tags). After seeing the disappointing results with the tight mapping function (see Figs. 4 and 5), we then tried the automatic mapping function, and happily found that the coupled model turned out to be much stronger than the baseline model. The automatic mapping function is built as follows. First, we use the baseline $Tagger_{CTB}$ to process PD-train.[7] Then, we collect all the mapping relationships between the gold-standard PD tags and the automatic noisy CTB tags. Finally, the automatically collected relationships are used as the automatic mapping function.
3) The **relaxed** mapping function ($\mathbf{m}_{relaxed}$, 179 bundled tags) is a looser version of the tight mapping function including 34 more weak mapping relationships. After seeing the failure of the tight mapping function and the success of the automatic mapping function, we then designed the relaxed mapping function, since the coupled model with the automatic mapping function is very inefficient (see Table V). We add high-frequency bundled tags from the automatic mapping function into the tight mapping function. In many cases, we need to find evidences from the two real datasets before making our decisions.
4) The **complete** mapping function ($\mathbf{m}_{complete}$, $|\mathcal{T}^a| \times |\mathcal{T}^b| = 33 \times 38 = 1,254$ bundled tags) allows one CTB tag to be mapped to all PD tags and vice versa. The complete mapping function leads to a very huge bundled tag space and makes the coupled model extremely slow. Nevertheless, we decided to experiment with the complete mapping function in order to better understand the capability of the coupled model in learning mappings between heterogeneous annotations. Thanks to this extra try, we further propose an online pruning strategy for the coupled model under complete mapping, which on the one hand dramatically improves the efficiency of the model and on the other hand preserves its superior tagging accuracy.

[6]http://hlt.suda.edu.cn/~zhli/mapping.html

[7]Alternatively, we could try the opposite direction by using $Tagger_{PD}$ to tag CTB-train. We could also merge the mapping rules from both directions by intersection or union. However, we did not try other alternatives since we found the automatic mapping function worked and achieved very close accuracy to the complete mapping function, as shown in Figs. 4 and 5.

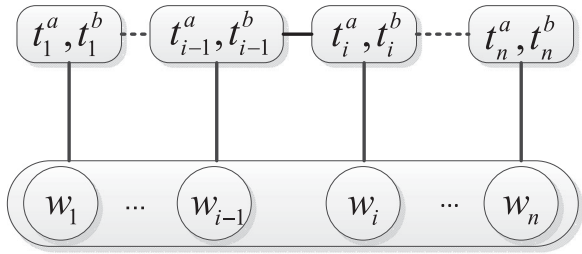


Fig. 2. Graphical structure of our coupled CRF model.

IV. Coupled POS Tagging ($Tagger_{CTB\&PD}$)

In the previous section, we illustrate how to transform the non-overlapping CTB and PD datasets from one-side tags into bundled tags based on predefined mapping functions. In this section, we formally introduce our coupled model, including the model formalization, how to learn from ambiguous labeling, how to train with two datasets, and the online pruning method.

Given a sentence, the goal of the coupled model is to simultaneously predict two POS tag sequences in the form of bundled tags. The basic idea is treating a bundled tag as a single tag and letting the CRF-based model learn and infer in the bundled tag space. Fig. 2 shows the graphical structure of the coupled CRF model. Compared with the traditional CRF, the only change of the coupled CRF is assigning a bundled tag to each word instead of a single-side tag. Then, the score of a bundled tag sequence is defined as follows, which is a straightforward extension of Eq. (3).

$$Score\left(\mathbf{x}, \left[\mathbf{t}^a, \mathbf{t}^b\right]; \theta\right) = \sum_{i=1}^{n+1} \theta \cdot \begin{bmatrix} \mathbf{f}\left(\mathbf{x}, i, \left[t_{i-1}^a, t_{i-1}^b\right], \left[t_i^a, t_i^b\right]\right) \\ \mathbf{f}\left(\mathbf{x}, i, t_{i-1}^a, t_i^a\right) \\ \mathbf{f}\left(\mathbf{x}, i, t_{i-1}^b, t_i^b\right) \end{bmatrix} \tag{8}$$

where the first item of the enlarged feature vector is called *joint features*, which can be obtained by instantiating Table I by

replacing t_i with a bundled tag $[t_i^a, t_i^b]$; the second and third items are called *separate features*, which are based on single-side tags. Take the *CTB* sentence (above) in Fig. 1 as an example. Suppose that the word "发展 4" takes a bundled tag of "$[NN, n]$". Then the 02 feature template in Table I is instantiated into three features, i.e., "发 展/[*NN,n*]", "发 展/*NN*", and "发展/*n*", corresponding to the three terms in Eq. (8) respectively. Experiments below show that the joint features capture the implicit mappings between heterogeneous annotations, and the separate features function as back-off features for alleviating the data sparseness problem of the joint features (see Table VIII).

A. Training With Ambiguous Labeling

After transformation based on a predefined mapping function, as described Section III, the non-overlapping CTB and PD datasets lie in the same bundle tag space. However, the remaining issue is that each word typically has multiple bundled tags as its gold-standard references, instead of a single tag in traditional supervised learning, as shown in Fig. 1. Formally, given an input sentence $\mathbf{x} = w_1 \dots w_n$, the word w_i is annotated with a set of bundled tags, denoted as $\mathcal{T}_i \subseteq \mathcal{T}^{a\&b}$. Then, the sentence $\mathbf{x}$ has an exponential-size set of ambiguous bundled tag sequences as its gold-standard references during training, denoted as $\mathcal{V} = \mathcal{T}_1 \cdots \times \mathcal{T}_i \times \dots \mathcal{T}_n \subseteq \mathcal{T}^n$. This is known as *ambiguous labeling* and can be regarded as a weakly-supervised learning scenario. The next problem is how to make the coupled model effectively learn its parameters θ (i.e., feature weights) from such ambiguous labeling.

Based on the idea of [21]–[23], we derive an ambiguous labeling-oriented training objective function for the CRF-based coupled model, and then apply standard stochastic gradient descent to iteratively learn the model parameters. We introduce the objective function and the derivations of its gradient in the following.

Given an input sentence $\mathbf{x}$ and a set of bundled tag sequence $\mathcal{V}$ as defined above, the probability of $\mathcal{V}$ is naturally the sum of probabilities of all tag sequences contained in $\mathcal{V}$:

$$p(\mathcal{V}|\mathbf{x};\theta) = \sum_{\mathbf{t}\in\mathcal{V}} p(\mathbf{t}|\mathbf{x};\theta) = \frac{Z'(\mathbf{x},\mathcal{V};\theta)}{Z(\mathbf{x};\theta)}$$
$$Z'(\mathbf{x},\mathcal{V};\theta) = \sum_{\mathbf{t}\in\mathcal{V}} e^{Score(\mathbf{x},\mathbf{t};\theta)} \tag{9}$$

where $Z(\mathbf{x};\theta)$ is the same normalization factor as Eq. (2); $Z'(\mathbf{x},\mathcal{V};\theta)$ can be regarded as a constrained normalization factor that only sums over the constrained space $\mathcal{V}$, and will be used several times in the following derivations.

Suppose the training data is $\mathcal{D} = \{(\mathbf{x}_j, \mathcal{V}_j)\}_{j=1}^N$. Then the gradient of the log likelihood is:

$$\frac{\partial \mathcal{L}(\mathcal{D};\theta)}{\partial\theta} = \sum_{j=1}^{N}\left(\frac{\partial \log Z'(\mathbf{x}_j,\mathcal{V}_j;\theta)}{\partial\theta} - \frac{\partial \log Z(\mathbf{x}_j;\theta)}{\partial\theta}\right)$$
$$= \sum_{j=1}^{N}\left(\sum_{\mathbf{t}\in\mathcal{V}_j} p(\mathbf{t}|\mathbf{x}_j,\mathcal{V}_j;\theta)\mathbf{f}(\mathbf{x}_j,\mathbf{t}) - E_{\mathbf{t}|\mathbf{x}_j;\theta}[\mathbf{f}(\mathbf{x}_j,\mathbf{t})]\right)$$
$$= \sum_{j=1}^{N}\left(E_{\mathbf{t}|\mathbf{x}_j,\mathcal{V}_j;\theta}[\mathbf{f}(\mathbf{x}_j,\mathbf{t})] - E_{\mathbf{t}|\mathbf{x}_j;\theta}[\mathbf{f}(\mathbf{x}_j,\mathbf{t})]\right) \tag{10}$$

where $p(\mathbf{t}|\mathbf{x},\mathcal{V};\theta)$ is the constrained conditional probability of $\mathbf{t}$ given $\mathbf{x}$ and $\mathcal{V}$, defined as:

$$p(\mathbf{t}|\mathbf{x},\mathcal{V};\theta) = \frac{e^{Score(\mathbf{x}_j,\mathbf{t};\theta)}}{Z'(\mathbf{x},\mathcal{V};\theta)} \tag{11}$$

which guarantees that the constrained conditional probabilities of all tag sequences in $\mathcal{V}$ sum to one, and all tag sequences outside $\mathcal{V}$ receive zero probabilities. $E_{\mathbf{t}|\mathbf{x},\mathcal{V};\theta}[\mathbf{f}(\mathbf{x},\mathbf{t})]$ is the feature expectations for $\mathbf{x}$ under the constrained search space $\mathcal{V}$. $E_{\mathbf{t}|\mathbf{x};\theta}[\mathbf{f}(\mathbf{x},\mathbf{t})]$ is the same unconstrained feature expectations as defined in Eq. (5) and can be computed according to Eq. (6) with a tiny change that the tag space $\mathcal{T}$ becomes the bundled tag space $\mathcal{T}^{a\&b}$ decided by the predefined mapping function in the scenario of coupled tagging.

Similar to the case of computing $E_{\mathbf{t}|\mathbf{x};\theta}[\mathbf{f}(\mathbf{x},\mathbf{t})]$ in Eq. (6), naively computing the constrained feature expectations $E_{\mathbf{t}|\mathbf{x},\mathcal{V};\theta}[\mathbf{f}(\mathbf{x},\mathbf{t})]$ requires enumerating an exponentially increased search space of $O(|\mathcal{V}| = |\mathcal{T}_1| \times \cdots \times |\mathcal{T}_n|)$. Similarly, we can factorize the constrained feature expectations based on the Markovian assumption defined in Eq. (3):

$$E_{\mathbf{t}|\mathbf{x},\mathcal{V};\theta}[\mathbf{f}(\mathbf{x},\mathbf{t})] = \sum_{\mathbf{t}\in\mathcal{V}} p(\mathbf{t}|\mathbf{x},\mathcal{V};\theta)\left(\sum_{i=1}^{n+1}\mathbf{f}(\mathbf{x},i,t_{i-1},t_i)\right)$$
$$= \sum_{i=1}^{n+1}\sum_{t'\in\mathcal{T}_{i-1},t\in\mathcal{T}_i}\mathbf{f}(\mathbf{x},i,t',t)\left(\sum_{\mathbf{t}\in\mathcal{V}:t_{i-1}=t',t_i=t} p(\mathbf{t}|\mathbf{x},\mathcal{V};\theta)\right)$$
$$= \sum_{i=1}^{n+1}\sum_{t'\in\mathcal{T}_{i-1},t\in\mathcal{T}_i}\mathbf{f}(\mathbf{x},i,t',t)p(i,t',t|\mathbf{x},\mathcal{V};\theta) \tag{12}$$

where $p(i,t',t|\mathbf{x},\mathcal{V};\theta)$ is the constrained marginal probability of tagging w_{i-1} as t' and w_i as t under the search space $\mathcal{V}$, which guarantees for any i,

$$\sum_{t'\in\mathcal{T}_{i-1},t\in\mathcal{T}_i} p(i,t',t|\mathbf{x},\mathcal{V};\theta) = 1 \tag{13}$$

By slightly modifying the classic *Forward-Backward* algorithm, we can compute all bigram marginal probabilities in polynomial time of $O(nm^2)$, where $m = \max_{1\le i\le n}|\mathcal{T}_i|$, meaning the maximum number of tags for a word in the search space $\mathcal{V}$.

B. SGD Training With Two Datasets

After computing the weight gradients of the log-likelihood function, we then apply standard gradient descent procedures to iteratively learn the feature weights θ. In this work, we adopt stochastic gradient descent (SGD) with L2-norm regularization for both the baseline and coupled models. The basic idea is approximating a gradient with a small batch of b training examples for feature weight update, as shown in Algorithm 1. Since

Algorithm 1: SGD training with two labeled datasets.

Input: Hyper-parameters: I, N', M', b
Training data: $\mathcal{D}^{(1)}_{train} = \left\{\left(\mathbf{x}^{(1)}_j, \mathcal{V}^{(1)}_j\right)\right\}_{j=1}^{N}$
$\mathcal{D}^{(2)}_{train} = \left\{\left(\mathbf{x}^{(2)}_j, \mathcal{V}^{(2)}_j\right)\right\}_{j=1}^{M}$
Development data: $\mathcal{D}^{(1)}_{dev}$, $\mathcal{D}^{(2)}_{dev}$
Output: θ
Initialization: $\theta_0 = \mathbf{0}$, $k = 0$
for $i = 1$ **to** I **do** {*iterations*}
  Randomly select N' instances from $\mathcal{D}^{(1)}_{train}$ and M' instances from $\mathcal{D}^{(2)}_{train}$
  Merge the selected instances as a new dataset $\mathcal{D}_i$
  Shuffle $\mathcal{D}_i$
  Traverse $\mathcal{D}_i$, b instances as a batch
    $\mathcal{D}^b_k \subseteq \mathcal{D}_i$ is the current batch, then update as below.
    $\theta_{k+1} = \theta_k + \eta_k \frac{1}{b} \nabla \mathcal{L}(\mathcal{D}^b_k; \theta_k)$
    $k = k + 1$
  Evaluate θ_k on $\mathcal{D}^{(1)}_{dev}$, report first-side accuracy
  Evaluate θ_k on $\mathcal{D}^{(2)}_{dev}$, report second-side accuracy
end for

computations among examples in the same batch are mutually independent, we implement a parallelized version of SGD to accelerate training.

Different from a traditional supervised learning scenario, we have two separate non-overlapping training datasets and each dataset contains one-side tags. Directly merging the two datasets without corpus-balancing may cause CTB to be overwhelmed by PD (see Table IV), since PD contains much more sentences than CTB (see Table II). Moreover, when training statistical models, it is usually beneficial to take smaller steps in monitoring tagging accuracies on the development datasets. Instead of reporting an accuracy after traversing the whole training datasets, we prefer using a random subset of the training data in an iteration and reporting an accuracy after each iteration.

Therefore, we propose a simple corpus-weighting strategy, as shown in Algorithm 1. The algorithm on the one hand has the flexibility of balancing the number of sentences from each training dataset in an iteration, and on the other hand monitors tagging accuracies of the model in smaller intervals. The idea is to randomly sample instances from each training data in a certain proportion before each iteration. The sampled data $\mathcal{D}_i$ is then used for i^{th}-iteration training. $\mathcal{D}^b_k$ is a small batch of training instances used in globally k^{th}-step update; b is the batch size; η_k is the update step size. In this work, we use $b = 30$ for all models based on preliminary experimental results, and follow the implementation in CRFsuite[8] to adjust η_k in a simulated annealing fashion.

After each iteration, we evaluate the current model on two development datasets, as shown at line 1 and 1. Since each dataset has only one-side gold-standard tags, we only report tagging accuracy on the single-side tags by ignoring another-side tags in the output bundled tags. We train each model for at most $I = 1000$ iterations, and stop training if tagging accuracies on both sides do not improve within 30 consecutive iterations. For final evaluation, we also run the coupled model on two test datasets, each having only one-side tags. For each side, we choose the model θ_k that achieves best accuracy on the corresponding development dataset with the same-side gold-standard tags.

Based on findings in our previous work [24], we use $N' = 5K$ and $M' = 5K$ as the default setting for merging two training data in each iteration. We also investigate the use of different N' and M' in Table IV.

[8]http://www.chokkan.org/software/crfsuite/

TABLE III
ONLINE PRUNING WITH DIFFERENT r AND λ

r	λ	Accuracy (%)		Tagging Speed
		CTB-dev	PD-dev	Tokens/Second
2	0.98	94.25	95.03	1278
4	0.98	95.06	95.66	1136
8	0.98	**95.14**	**95.83**	365
16	0.98	95.12	95.81	71
8	10^{-6}	91.85	94.35	1461
8	0.10	94.22	95.19	1461
8	0.30	94.56	95.46	1364
8	0.50	94.76	95.63	1278
8	0.70	94.88	95.70	1136
8	0.80	95.04	95.73	1023
8	0.90	95.15	95.79	758
8	0.95	95.13	**95.82**	465
8	0.99	**95.15**	95.74	244
8	1.00	**95.15**	95.76	127

TABLE IV
EFFECT OF DIFFERENT SETTINGS OF CORPUS WEIGHTING

Training data	N'	M'	Accuracy (%)	
			CTB-dev	PD-dev
CTB-train + *PD*-train	$5K$	$2.5K$	**95.15**	95.68
	$5K$	$5K$	95.14	95.83
	$5K$	$10K$	95.06	**95.90**
CTB-train + *PD*-train-large	$5K$	$5K$	**95.29**	96.55
	$2K$	$25K$	94.78	**96.78**

C. Complete Mapping With Context-Aware Online Pruning

Our study shows that the coupled model with the complete mapping function achieves the best tagging accuracy, but is prohibitively inefficient in training and inference. The reason is that under the complete mapping function, we need to enumerate $|\mathcal{T}^a| \times |\mathcal{T}^b| = 1,254$ bundled tags for each word when computing the feature expectations $E_{\mathbf{t}|\mathbf{x};\theta}[\mathbf{f}(\mathbf{x}, \mathbf{t})]$ in Eq. (10). In contrast, computing the constrained feature expectations $E_{\mathbf{t}|\mathbf{x},\mathcal{V};\theta}[\mathbf{f}(\mathbf{x}, \mathbf{t})]$ according to Eq. (12) is not the bottleneck, since we only need to enumerate either $|\mathcal{T}^a| = 33$ or $|\mathcal{T}^b = 38|$ tags for each word.

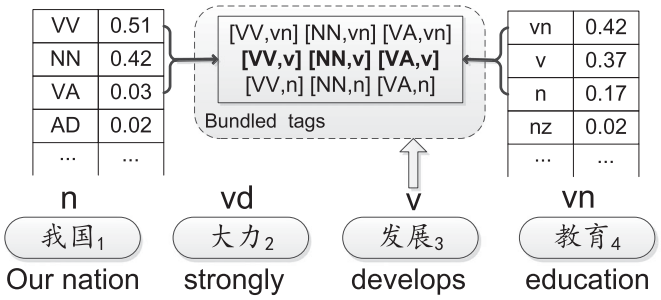


Fig. 3. Illustration of online pruning with $r = 3$ on a PD training sentence.

In order to improve the efficiency of the coupled model, we propose to approximately compute the feature expectations in Eq. (10) based on a context-aware online pruning strategy. The basic idea is considering only a small set of plausible bundled tags instead of all 1,254 bundled tags according to contextual evidences on only one-side separate features. Given an input sentence $\mathbf{x}$, the online pruning strategy works as follows.

1) Apply the Forward-Backward algorithm on only the first-side tag space $\mathcal{T}^a$ using the subset of corresponding separate features $\mathbf{f}(\mathbf{x}, i, t^a_{i-1}, t^a_i)$ defined in Eq. (8) and the current feature weights θ_k.
2) For each word w_i, compute the marginal probability $p(i, t^a|\mathbf{x}; \theta)$ for all $t^a \in \mathcal{T}^a$, which is similar to $p(i, t', t|\mathbf{x}; \theta)$ in Eq. (6).
3) For each word w_i, keep r tags with the highest marginal probabilities as its candidate tags, denoted as $\tilde{\mathcal{T}}^a_i$.
4) Similarly, we can get the second-side candidate tags $\tilde{\mathcal{T}}^b_i$ for w_i by running through the above three steps with the second-side separate features $\mathbf{f}(\mathbf{x}, i, t^b_{i-1}, t^b_i)$.
5) We define the Cartesian product $\tilde{\mathcal{T}}_i = \tilde{\mathcal{T}}^a_i \times \tilde{\mathcal{T}}^b_i$ as the set of possible tags for w_i in computing the feature expectations $E_{\mathbf{t}|\mathbf{x};\theta}[\mathbf{f}(\mathbf{x}, \mathbf{t})]$.
6) Suppose the training sentence $\mathbf{x}$ has the first-side gold-standard tags $\check{\mathbf{t}}^a$, meaning that the manually labeled first-side tag of w_i is $\check{t}^a_i$. Then we define the Cartesian product $\mathcal{T}_i = \{\check{t}^a_i\} \times \mathcal{T}^b_i$ as the gold-standard ambiguous tags for w_i in computing the constrained feature expectations $E_{\mathbf{t}|\mathbf{x},\mathcal{V};\theta}[\mathbf{f}(\mathbf{x}, \mathbf{t})]$. Analogously, we can generate $\mathcal{T}_i$ when $\mathbf{x}$ has the second-side gold-standard tags $\check{\mathbf{t}}^b$.

To illustrate the operations in Step (5) and (6) more clearly, we define $\tilde{\mathcal{V}} = \tilde{\mathcal{T}}_1 \times \cdots \times \tilde{\mathcal{T}}_n$ and $\mathcal{V} = \mathcal{T}_1 \times \cdots \times \mathcal{T}_n$. Then, the gradient of the log-likelihood function in Eq. (10) can be precisely rewritten as follows, which is the feature expectations under the constrained space $\mathcal{V}$ minus those under $\tilde{\mathcal{V}}$.

$$\frac{\partial \mathcal{L}(\mathcal{D};\theta)}{\partial \theta} = \sum_{j=1}^{N} \left(E_{\mathbf{t}|\mathbf{x}_j,\mathcal{V}_j;\theta}[\mathbf{f}(\mathbf{x}_j, \mathbf{t})] - E_{\mathbf{t}|\mathbf{x}_j,\tilde{\mathcal{V}}_j;\theta}[\mathbf{f}(\mathbf{x}_j, \mathbf{t})] \right) \tag{14}$$

Fig. 3 shows an example of online pruning with $r = 3$ on a PD training sentence. The marginal probabilities of single-side tags are computed based on corresponding separate features. We only keep the most likely r tags for each side to compose bundled tags, which is further used as the possible bundled tags for the word when computing the second term in Eq. (14). Since the gold-standard tag of the word is known to be "v", the corresponding three bundled tags, marked in bold font, are used as gold-standard references when computing the first term in Eq. (14).

Regarding time complexity, Step (1–4) requires roughly the same operations as two baseline models; Step (5) needs to run Forward-Backward in the search space of $\tilde{\mathcal{V}}$, thus has a time complexity of $O(nr^4)$, whereas Step (6) runs in the search space of $\mathcal{V}$, thus has a time complexity of $O(nr^2)$. Our experiments show that raising $r = 8$ to 16 leads to no further accuracy gain for our task (see Table III), and online pruning greatly improves both training and inference efficiency with little accuracy loss when $r = 8$.

Beside r, we use another hyper-parameter λ to further reduce the number of one-side tag candidates. The intuition is that in many cases, especially when the model becomes strong after trained for several iterations, we may only need to use r' ($< r$) most likely candidate tags, since the remaining tags have too small probabilities. Therefore, for each word, we define r' as the smallest number of most likely candidate tags whose accumulative probability is larger than λ. Then, we only keep the $\min(r', r)$ most likely candidate tags for this word. Our preliminary experiments show that λ clearly improves efficiency but has little effect on tagging accuracy when $\lambda > 0.9$ and $r = 8$. Based on experimental results, we find that $r = 8$ and $\lambda = 0.98$ are fine for our task in hand.

V. Experiments

We conduct experiments on CTB (version 5.1) and PD, as shown in Table II. We adopt the standard training/dev/test data split for CTB [30]. For PD, we adopt a different setting of data split from that in our previous work [24].[9] We use the sentences in January as the training data, the first 2,000 sentences in February as the development data, and the first 5,000 sentences in June as the test data. Furthermore, to investigate how the scale of PD affects performance of the coupled model, we use a large training data, referred to as *train-large*, by combining all PD sentences except those in February and the first 5,000 test sentences in June. In all our experiments, unless particularly pointed out, we use PD-train instead of PD-train-large for training.

To evaluate different methods on the task of annotation conversion, we have annotated 1,000 ***PD*** sentences with ***CTB*** tags. The sentences are randomly sampled and removed from PD before data split. For each sentence, 20% most ambiguous or

[9]In our previous work [24], we randomly sample 1,000 and 2,500 sentences as *PD*-dev/test. This new data split is actually suggested by a colleague with two purposes. First, under the previous random data split, we find the coupled approach can hardly defeat the baseline single-side tagging model on PD-dev/test, which we believed was because the PD-train and PD-dev/test are too similar to be further improved over the baseline model. After experiments on the new data split, we find the accuracy improvement on PD-dev/test is still marginal (about 0.1%). Furthermore, the guide-feature method is inferior to baseline method by about 0.2%. Now we realize that the accuracy on PD-dev/test is just very difficult to improve. The second purpose is to make the duplication of our experiments and results easier with such data split.

difficult tokens are selected for manual annotation to save annotation effort. The difficulty of a token is measured with its token-wise marginal probability produced by the baseline CRF model trained on CTB. The basic assumption is that a word is more difficult to annotate if its most likely tag candidate gets lower marginal probability. Please refer to our earlier work [24] for the detailed annotation process. Finally, we obtain 5,769 words with both CTB and PD tags. The data is also released with the codes for free research usage.

We use the standard token-wise tagging accuracy as the evaluation metric.[10] For significance test, we adopt Dan Bikel's randomized parsing evaluation comparator [31].[11]

A. Hyper-Parameter Tuning for the Coupled Model

Tuning r and λ: Table III shows the performance of the coupled model with the complete mapping function and online pruning under different thresholds r and λ, where r means the maximum number of candidate single-side tags for each word, and λ means the accumulative probability threshold for further truncating the candidates, as illustrated in Section IV-C.

In the first major row, we set $\lambda = 0.98$ and find that increasing r from 2 to 8 leads to consistently improved accuracy on both CTB and PD sides. However, $r = 16$ does not further improve accuracy, indicating that tags below the top-8 are mostly very unlikely ones and thus insignificant for computing feature expectations. In fact, we find that the two models with $r = 8$ and $r = 16$ have nearly the same accuracy curves on both *CTB*-dev and PD-dev during training. In terms of efficiency, we also report the tagging speed (tokens processed per second) on CTB-dev during evaluation phase. It is obvious that the tagging speed drops when r becomes larger.

Then we set $r = 8$ and try different λ in the second major row. We find that λ clearly improves efficiency but has little effect on tagging accuracy when $\lambda > 0.9$. Although $\lambda = 0.9$ achieves similar accuracy but is faster, we still choose $\lambda = 0.98$ for later experiments so that we can compute the feature expectations more precisely.

Then we gradually decrease λ and find that the accuracies on both sides consistently drops as well. When $\lambda = 10^{-6}$, the online pruning strategy always leaves only one *CTB* tag and one *PD* tag. Then the joint features become useless into model, and the gradients defined in Eq. 14 become unreasonable. This explains why the accuracies become extremely low on both sides.

Effect of N' and M': Table IV shows the results on weighting CTB and PD in different proportions (N' CTB sentences and M' PD sentences) at each iteration, as illustrated in Section IV-B.

[10]To clarify a question asked by a few colleagues, it is necessary to point out that the coupled CRF is not directly evaluated on bundled tags, since CTB and PD dev/test data only have one-side gold-standard tags. For instance, when the coupled model is evaluated on CTB-dev, only the CTB-side tags in the bundled tags are extracted for evaluation and an accuracy on CTB tags is reported. Therefore, the comparison between the baseline model and the coupled model is fair.

[11]http://www.cis.upenn.edu/~dbikel/software.html

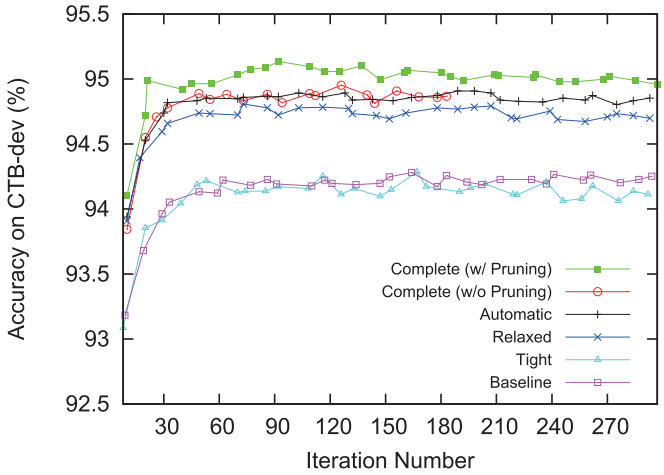


Fig. 4. Accuracy curve on CTB-dev of the baseline model and the coupled model with different mapping functions.

We set $N' = 5$ K (5 thousand sentences), and vary $M' =$ 2.5 K/5 K/10 K. The finding is similar to our previous work [24]. Reducing M' slightly increases accuracy on CTB-dev but decreases accuracy on PD-dev. In contrast, enlarging M' leads to improved accuracy on PD-dev but decreased accuracy on CTB-dev.

We also conduct experiments with *PD*-train-large. We set $N' = 2$ K and $M' = 25$ K (about 10% of the whole training data) to mimic the setting of directly merging the two training data. Comparing with $N' = M' = 5$ K, accuracy on *CTB*-dev drops by 0.51% whereas accuracy on *PD*-dev increases by 0.23%. This is consistent with our intuition that the much larger-scale *PD* would overwhelm *CTB* without corpus-weighting.

Based on the above results, we adopt $N' = 5$ K and $M' = 5$ K in all other experiments.

B. Comparison of Mapping Functions

Fig. 4 shows the accuracy curves on CTB-dev of the baseline model and the coupled model with different mapping functions. For better comparison, the baseline model is also trained using $5K$ CTB-train sentences in each iteration. Please note that for each curve we only draw one peak point in every 10 iterations, where the y-axis is the peak accuracy and the x-axis is the corresponding iteration number. We find it is a nice way to clearly present the results. The coupled model using the complete mapping function without online pruning is very slow and not completed in training. We will update the results when available.

We can see that, contrary to our intuitive assumption, using the tight mapping function leads to slightly worse accuracies than the baseline model. The relaxed mapping function outperforms the tight function by a large margin. The automatic function works slightly better than the relaxed one. The complete mapping function without online pruning achieves similar accuracy to the automatic mapping function. Surprisingly, the complete mapping with online pruning even slightly outperforms the one without online pruning by a small margin. Since

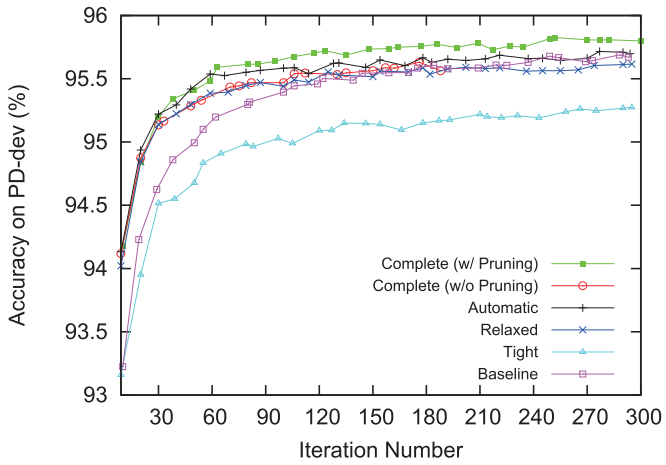


Fig. 5. Accuracy curve on PD-dev of the baseline model and the coupled model with different mapping functions.

TABLE V
COMPARISON OF DIFFERENT MODELS AND MAPPING FUNCTIONS ON DEV DATA

	Accuracy (%)		Speed
	CTB-dev	PD-dev	Toks/Sec
Complete (w/ Pruning)	**95.14** (+0.86)	**95.83** (+0.11)	365
Complete (w/o Pruning)	94.95 (+0.67)	95.63 (-0.09)	3
Automatic	94.91 (+0.63)	95.72 (—)	33
Relaxed	94.81 (+0.53)	95.62 (-0.10)	127
Tight	94.29 (+0.01)	95.33 (-0.39)	232
Guide-feature CRF	94.81 (+0.53)	95.45 (-0.27)	584
Baseline CRF	94.28	95.72	1573

the coupled model under the complete mapping function without online pruning is very slow, we throw away all features with frequency less than 3 before training. For other coupled models, we use all features for training.

Fig. 5 presents the accuracy curves of different models on PD-dev. The overall trend is similar to that in Fig. 4. The main difference is that the baseline model is very strong on PD, and the coupled models achieve very little accuracy gain. The coupled model with the tight mapping function is much worse than the baseline. Using the relaxed mapping function leads to large improvement but is still slightly worse than the baseline. Using the automatic mapping function achieves nearly the same accuracy with the baseline. Finally, the coupled model using the complete mapping with online pruning is slightly better than the baseline.

Table V summarizes different models and mapping functions in terms of both peak accuracy and tagging speed. The coupled model using complete mapping function with online pruning achieves best accuracies on both CTB-dev and PD-dev. It outperforms both the baseline and guide-feature based models. For the guide-feature based CRF, we re-implement the method described in [18]. In terms of efficiency, we can see that online pruning boosts tagging speed by two magnitudes, making the coupled model 120 times faster than the one without online pruning, and comparable with the baseline model.

Our findings are summarized as follows.

1) The coupled model achieves higher accuracy with more relaxed mapping functions. In fact, the results suggest that the complete mapping function works best, indicating the coupled model effectively learns the implicit mappings between heterogeneous annotations without relying on a carefully designed mapping function. Section V-F presents more discussions on this issue.
2) Online pruning greatly boosts the tagging speed of the coupled model with the complete mapping function by two magnitudes without accuracy loss.
3) The coupled model achieves only small accuracy gain over the baseline on PD. We believe the reason is that the scale of PD-train is large enough, making the baseline model too strong to defeat. In fact, the guide-feature based model achieves lower accuracy (-0.27%) on PD-dev than the baseline ($+0.53$ on CTB-dev in contrast). Section V-E presents more analysis on the effect of the scale of PD-train.

TABLE VI
COMPARISON OF TAGGING ACCURACIES ON TEST DATA

	Accuracy (%)	
	CTB-test	PD-test
Coupled CRF	94.74 (+0.67†‡)	95.95 (+0.13†‡)
Guide-feature CRF	94.35 (+0.28†)	95.63 (-0.19†)
Baseline CRF	94.07	95.82
Best reported [30]	94.60	—

C. Final Results on Test Data

Table VI presents results on the test data. The coupled model uses the complete mapping function with online pruning. The best reported result on CTB is from [30], which jointly models Chinese POS tagging and dependency parsing. † means the corresponding approach significantly outperforms the baseline, whereas ‡ means the accuracy difference between the guide-feature based model and the coupled model is significant. The significance test results are gained at a confidence level of $p < 0.005$. We can see that the coupled model significantly outperforms both the baseline and guide-feature CRFs on both test datasets.

D. Results on Annotation Conversion

We evaluate different methods on the annotation conversion task using our newly annotated 1,000 sentences in PD-conversion. Given an input sentence and its source-side gold-standard tags, annotation conversion aims to predict the another-side tags. In our specific case, the gold-standard PD-side tags are provided, and the goal is to obtain the CTB-side tags. The performance is measured on the 5,769 words having manually annotated CTB-side tags.

TABLE VII
PD-TO-CTB CONVERSION ACCURACY ON THE PD-CONVERSION DATA

	Conversion Accuracy (%)
Coupled CRF	**94.02** (+3.76) †‡
Guide-feature CRF	92.96 (+2.70) †
Baseline CRF	90.26

TABLE VIII
FEATURE STUDY ON THE COUPLED MODEL WITH RELAXED MAPPING FUNCTION

	Accuracy (%)	
	CTB-dev	PD-dev
Coupled (w/ all features)	94.81 (+0.53)	95.62 (−0.10)
Coupled (w/ separate features)	94.27 (−0.01)	95.64 (−0.08)
Coupled (w/ joint features)	92.16 (−2.12)	94.42 (−1.30)
Baseline CRF	94.28	95.72

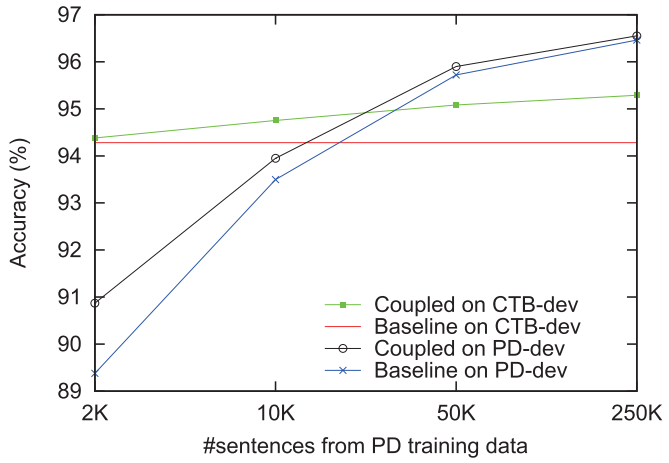


Fig. 6. Effect of the scale of PD training data in sentence number (thousands).

Our coupled model can naturally perform annotation conversion based on **constrained decoding**. The idea is that during decoding, the model only considers the bundled tags that compatible with the gold-standard PD-side tags. Specifically, during online pruning, we directly use the gold-standard PD-side tag sequence as the candidate tags ($r = 1$), whereas only the CTB-side tags are pruned as usual.

For the guide-feature based method, we feed into the model the gold-standard PD-side tags to compose guide features. The baseline model does not use the gold-standard PD-side tags at all. Table VII shows the results. Again, † (vs. baseline) and ‡ (vs. guide-feature model) represent the significance test results at a confidence level of $p < 0.005$. The POS tagging accuracy on this data is much lower than those in Table VI, because the 5,769 words used for evaluation are 20% most ambiguous tokens in each sentence. We can see that the coupled model significantly outperforms both the baseline and guide-feature based models by a large margin on the task of annotation conversion.

E. Analysis

In this section, we analyze the coupled model from different perspectives in order to better understand the model. For the feature study in Table VIII, we adopt the coupled model without online pruning under the relaxed mapping function, which is explained in the "Feature study" part. For all other experiments, we adopt the coupled model with online pruning under the complete mapping function.

On scale of PD: Fig. 6 shows the accuracy changes of the baseline and coupled models corresponding to different scales of PD training data. 250 K means using PD-train-large in Table II; 50 K means using all sentences in PD-train; 10 K/2 K means using the first $10,000/2,000$ sentences in PD-train.

The baseline CTB-model is not affected by the scale of PD training data. It is clear that the coupled model achieves larger improvement on *CTB*-dev over the baseline with larger scalee of PD.

In contrast, as PD becomes larger, the accuracy gap on *PD*-dev becomes smaller between the coupled and baseline models. This support our earlier arguments on why the coupled model only achieves slight improvement on PD-dev/test data. Actually, when using $10K/2K$ PD training sentences, the coupled model can get larger accuracy gains on *PD*-dev (+0.46% and +1.49%). In other words, results in Fig. 6 indicate that the gains on *PD*-dev/test from the coupled approach decline as the base data (i.e., *PD*-train) gets larger, which is understandable. However, we argue that such direct evaluation on *PD*-dev/test are misleading and underestimate the gains from the coupled approach, since *D*-dev/test are from the same genre/domain with *PD*-train, which makes it more challenging to further improve accuracy over the baseline single-side tagging model. We expect more gains can be obtained when the coupled model is applied to open-domain texts.

Feature study: Table VIII investigates individual contributions of the two kinds of features, namely the joint features based on bundled tags and separate features based on single-side tags, as defined in Eq. (8). We adopt the coupled model with the relaxed mapping function for this study due to two-fold reasons. First, the coupled model using complete mapping function is prohibitively inefficient without pruning. Second, under the online pruning framework, the feature ablation study is not possible since the separate features are indispensable to perform pruning.

We can see that when using only separate features, the coupled model achieves nearly the same accuracies on both CTB-dev and PD-dev compared with the baseline model. The reason is that due to the lack of joint features, no connection is built between the two sets of annotations, and hence no help can be expected from each other. When using only joint features, the coupled model becomes largely inferior to the baseline, which is due to the data sparseness problem for the joint features. Furthermore, when the two sets of features are combined, the coupled model achieves large improvements over the baseline on CTB-dev and slight drop on PD-dev.

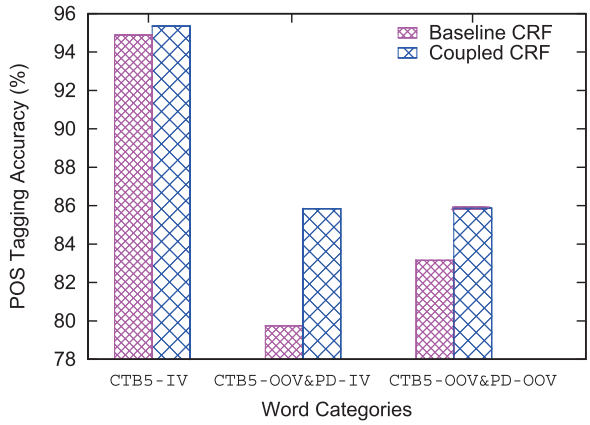


Fig. 7. Accuracy r.w.t. different IV-OOV word categories on CTB-test.

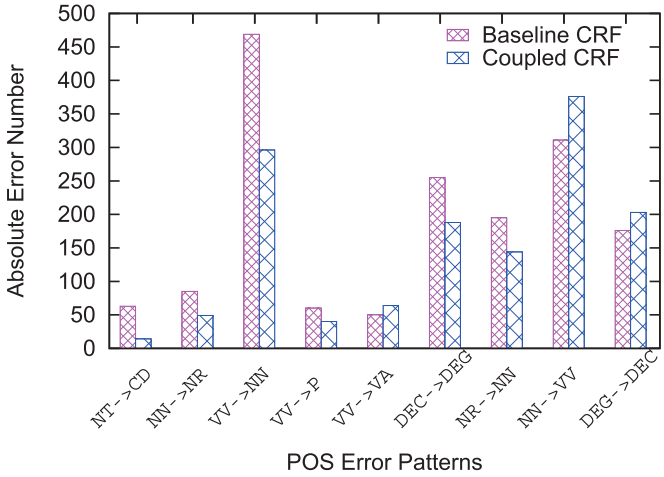


Fig. 8. Statistics for different POS error patterns on CTB-test.

From the results, we can conclude that both joint features and separate features are indispensable components and complementary to each other in the coupled model.

On in/out-of-vocabulary (IV/OOV) words: Fig. 7 investigates how the coupled model improves tagging accuracy from the perspective of IV-OOV word categories. We divide the words in CTB-test into three word categories: words in CTB-train (CTB-IV, proportion of 93.8%), words out of CTB-train but in PD-train (CTB-$OOV\&PD$-IV, 2.0%), and words out of both CTB-train and PD-train (CTB-$OOV\&PD$-OOV, 4.2%).

The accuracy on CTB-IV words is improved only by $+0.46\%$. The accuracy improvement is larger on CTB-$OOV\&PD$-OOV words ($+2.72\%$). The largest gain is from CTB-$OOV\&PD$-IV words ($+6.10\%$), showing that under the coupled model, an important contribution from heterogeneous annotations is that one data (i.e., PD) covers a lot of annotations for words that are missing in another data (i.e., CTB).

On POS tag error patterns: Fig. 8 shows how the coupled model changes the distribution of a number of high-frequency POS tag error patterns compared with the baseline model. An error pattern "*X* → *Y*" means that the focus word, whose true tag is "*X*", is assigned a wrong tag "*Y*". We choose these error patterns with largest reduction/increase ratio in number from the baseline model to the coupled model, and rank them in descending order of absolute change ratio.

The coupled model achieves the largest reduction ratio (77.8% error reduction) on "*NT* → *CD*" (temporal nouns mistagged as cardinal numbers).

We can also see that the coupled model greatly reduces the error numbers of both "$NR \rightarrow NN$" (proper nouns vs. normal nouns) and "$NN \rightarrow NR$". This suggests that PD provides more annotations for resolving "$\{NN, NR\}$" ambiguities. A typical example is "胡适/*NR*" ("Hú Shì", a Chinese person name), which appears 19 times in *CTB*-test, but does not appear in *CTB*-train. As a result, the baseline tagging model trained on *CTB*-train only achieves 8/19 = 42.11% accuracy on this word. In contrast, the coupled model trained on both *CTB*-train and *PD*-train achieves 18/19 = 94.74% accuracy, simply because "胡适/*nr*" appears 2 times in *PD*-train.

The coupled model decreases the number of "$VV \rightarrow NN$" (verb vs. normal noun) by 36.9% percent, but increases the number of "$NN \rightarrow VV$" by 20.9% percent at the same time. This indicates that the annotations in PD may be inclined to assign "VV" when deciding "$\{VV, NN\}$" ambiguities. Take "发展" (translated to "develop" as *VV*, "development" as *NN*) as an example. *CTB*-test contains 63 occurrences of "发展/*NN*" and 42 occurrences of "发展/*VV*". The baseline model makes 14 mistakes of "发展/*VV* → *NN*" and 3 of "发展/*NN* → *VV*", whereas the coupled model makes 10 of "发展/*VV* → *NN*" and 8 of "发展/*NN* → *VV*".

F. Discussion

Regarding this work, many audience and reviewers ask the same question, that is why the complete mapping works best for the coupled model whereas the manually designed relaxed mapping hurts performance. What we find in this work is contrary to the common intuition that statistical models usually become stronger when human knowledge is injected into statistical models.

Our short explanation is that by using the manually designed mapping functions, human knowledge introduces not only stronger supervision with a smaller answer space, but also more noise by ignoring special and irregular mappings. In the case of using the tight mapping function, noise outweighs supervision. As aforementioned, training from ambiguous labeling can be regarded as a kind of weakly-supervised learning. When the mapping function becomes tighter, harder constraints are imposed on tag mappings, thus producing less bundled tags for each word. In other words, supervision is stronger in model training. However, more noise is also introduced at the same time since the tag mapping relationship between CTB and PD is too complicated due to many factors as discussed in Section III-A. As a result, the safe and easy way suggested by this work is to allow all mappings with the complete mapping function, and let the model automatically learn the implicit mapping relationship.

From another perspective, although the two training datasets are non-overlapping at sentence level, they share many common

words, or contexts, or segments (sub-sentences), from which the coupled model can effectively build connections between the two tag sets with the help of the joint and separate features, without referring to any manually designed mapping rules.

The results clearly show that the coupled model can effectively learn the implicit mappings between heterogeneous annotations under the complete mapping function. This makes the coupled model more general and useful, especially with online pruning for efficiency guarantee. After all, in many cases, it is very hard to manually summarize the mapping relationships between heterogeneous annotations for a particular task. Instead, we may directly use the complete mapping function and let the model learn from data based on findings in this work.

VI. Related Work

This work is partially inspired by [32], who propose a model that performs heterogeneous Chinese word segmentation and POS tagging and produces two sets of results following CTB and PD styles respectively. Different from our CRF-based coupled model, their approach adopts a linear model, which directly combines two separate sets of features based on single-side tags, without considering the interacting joint features between the two annotations. They adopt an approximate decoding algorithm which tries to find the best single-side tag sequence with reference to tags at the other side. In contrast, our approach is a direct extension of traditional CRF, and is more theoretically simple from the perspective of modelling. The use of both joint and separate features is proven to be crucial for the success of our coupled model. In addition, their work indicates that their model relies on a hand-crafted loose mapping function between annotations, which is opposite to our findings. The naming of the "coupled" CRF is borrowed from the work of [33], which treats the joint task of Chinese word segmentation and POS tagging as two coupled sequence labeling problems.

[34] propose a shift-reduce dependency parsing model which can simultaneously learn and produce two heterogeneous parse trees. However, their approach assumes the existence of data with annotations at both sides, which is obtained by converting phrase-structure trees into dependency trees with different heuristic rules.

This work is also closely related with multi-task learning, which aims to jointly learn multiple related tasks with the benefit of using interactive features under a share representation [35]–[37]. However, according to our knowledge, multi-task learning typically assumes the existence of data with labels for multiple tasks at the same time, which is unavailable in our situation.

Our model is similar to a factorial CRF [38], in the sense that the bundled tags can be factorized to two connected latent variables. Initially, factorial CRFs are designed to jointly model two related (and typically hierarchical) sequential labeling tasks, such as POS tagging and chunking. In this work, our coupled CRF jointly models two same tasks which have different annotation schemes. Moreover, this work provides a natural way to learn from incomplete annotations where one sentence only contains one-side labels.

Our work is also related with the problem of domain adaptation. [39] proposes an interestingly simple yet effective approach to domain adaptation where we have two labeled datasets from different domains but share the same label set, i.e., a large-scale source data and a small-scale target data. [40] tackle a similar problem as [39] except that the two datasets have disparate label sets. They first derive tag embedding via canonical correlation analysis (CCA), which is then used for computing tag similarities and mapping different tag sets into the same tag space. Then, they propose a new transfer approach that uses source domain data to pre-train their Hidden-Unit CRF to get good initialization of a part of parameters.

Learning with ambiguous labeling are previously explored for classification [21], sequence labeling [22], parsing [23], [41]–[43]. Recently, researchers propose to derive natural annotations from web data, and then transform them into ambiguous labeling to supervise Chinese word segmentation models [11]–[13].

Last but not least, although this works focuses on POS tagging as a case study, our proposed approach in this paper can be easily applied to other sequence labeling problems in NLP, e.g., NER [2], semantic slot filling [44], and mention detection [4], as long as there exist multiple heterogeneously labeled data for the task.

VII. Conclusion

This paper proposes an effective coupled sequence labeling model for exploiting multiple non-overlapping datasets with heterogeneous annotations. To solve the efficiency issue, we also propose a context-aware online pruning approach for approximate gradient computation. We conduct experiments on the task of Chinese POS tagging, using two large-scale labeled data, i.e., *CTB* and *PD*. Results show that our proposed coupled model significantly outperforms the baseline and guide-feature based methods on accuracy of both one-side POS tagging and annotation conversion, and have comparable tagging speed as well. Compared with the baseline CRF model, our coupled model can improve one-side tagging accuracy from 94.07% to 95.74% ($+0.67\%$) on CTB-test, and from 90.26% to 94.02% ($+3.76\%$) on *PD*-to-*CTB* annotation conversion. Especially, we can draw several interesting findings from this work.

1) The coupled model does not rely on any predefined mapping constraints and is able to automatically learn implicit mappings between heterogeneous annotations.
2) The approximate context-aware online pruning approach dramatically improves train and inference efficiency with little accuracy loss.
3) Both the separate features and joint features make indispensable contribution to our coupled model.
4) The coupled model gains most accuracy improvement on words that are absent in *CTB*-train but seen in *PD*-train.

For future, we would like to extend this work in two directions. First, we will look into whether the proposed coupled model is effective for other sequence labeling problems where multiple datasets with heterogeneous annotations exist. Particularly, we will apply the coupled approach to joint word segmentation and POS tagging, so that we tackle the issue that CTB and PD differs in word segmentation standard, which is currently ignored in this work. The challenge is that the tag space would be nearly four times larger due to product of character-in-word

position tags (*B/I/E/S*) and POS tags, and manually designing a mapping function seems more impossible.

Second, we would like to extend the coupled model to learn from more than two heterogeneous annotations, which would be straightforward without worrying the issue of exploded bundled tag space thanks to online pruning.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for helping them elevate this paper from different perspectives. The authors also would like to thank the undergraduate students F. Lu and X. Wang for building the POS tagging annotation system, and L. Lu, D. Hu, Y. Zhang, J. Zhang, Q. Yan, and X. Jiang for data annotation.

REFERENCES

[1] A. Celikyilmaz, R. Sarikaya, M. Jeong, and A. Deoras, "An empirical investigation of word class-based features for natural language understanding," *IEEE/ACM Trans. Audio, Speech Lang. Process.*, vol. 24, no. 6, pp. 994–1005, Jun. 2016.

[2] Y. Benajiba, M. Diab, and P. Rosso, "Arabic named entity recognition: A feature-driven study," *IEEE Trans. Audio, Speech Lang. Process.*, vol. 17, no. 5, pp. 926–934, Jul. 2009.

[3] W. Chen, M. Zhang, and Y. Zhang, "Distributed feature representations for dependency parsing," *IEEE/ACM Trans. Audio, Speech Lang. Process.*, vol. 23, no. 3, pp. 451–460, Mar. 2015.

[4] I. Zitouni and Y. Benajiba, "Aligned-parallel-corpora based semi-supervised learning for arabic mention detection," *IEEE/ACM Trans. Audio, Speech Lang. Process.*, vol. 22, no. 2, pp. 314–324, Feb. 2014.

[5] Z. Li, M. Zhang, W. Che, T. Liu, and W. Chen, "Joint optimization for Chinese pos tagging and dependency parsing," *IEEE/ACM Trans. Audio, Speech Lang. Process.*, vol. 22, no. 1, pp. 274–286, Jan, 2014.

[6] H. Ouchi, K. Duh, H. Shindo, and Y. Matsumoto, "Transition-based dependency parsing exploiting supertags," *IEEE/ACM Trans. Audio, Speech Lang. Process.*, vol. 24, no. 11, pp. 2059–2068, Nov. 2016.

[7] M. Jeong and G. G. Lee, "Triangular-chain conditional random fields," *IEEE Trans. Audio, Speech Lang. Process.*, vol. 16, no. 7, pp. 1287–1302, Sep. 2008.

[8] Z. Huang, V. Eidelman, and M. Harper, "Improving a simple bigram HMM part-of-speech tagger by latent annotation and self-training," in *Proc. 2009 Annu. Conf. North Amer. Chapter Assoc. Comput. Linguistics*, 2009, pp. 213–216.

[9] A. Søgaard, "Semi-supervised condensed nearest neighbor for part-of-speech tagging," in *Proc. 49th Annu. Meeting Assoc. Comput. Linguistics*, 2011, pp. 48–52.

[10] W. Sun and H. Uszkoreit, "Capturing paradigmatic and syntagmatic lexical relations: Towards accurate Chinese part-of-speech tagging," in *Proc. 50th Annu. Meeting Assoc. Comput. Linguistics*, 2012, pp. 242–252.

[11] W. Jiang, M. Sun, Y. Lü, Y. Yang, and Q. Liu, "Discriminative learning with natural annotations: Word segmentation as a case study," in *Proc. 51st Annu. Meeting Assoc. Comput. Linguistics*, 2013, pp. 761–769.

[12] Y. Liu, Y. Zhang, W. Che, T. Liu, and F. Wu, "Domain adaptation for CRF-based Chinese word segmentation using free annotations," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2014, pp. 864–874.

[13] F. Yang and P. Vozila, "Semi-supervised Chinese word segmentation using partial-label learning with conditional random fields," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2014, pp. 90–98.

[14] N. Xue, F. Xia, F.-D. Chiou, and M. Palmer, "The Penn Chinese Treebank: Phrase structure annotation of a large corpus," *Natural Lang. Eng.*, vol. 11, no. 2, 2005, pp. 207–238.

[15] F. Xia, "The part-of-speech tagging guidelines for the penn Chinese treebank 3.0," *Linguistic Data Consortium, Tech. Rep. IRCS-00–07*, 2000.

[16] S. Yu, H. Duan, X. Zhu, B. Swen, and B. Chang, "Specification for corpus processing at Peking University: Word segmentation, POS tagging and phonetic notation (In Chinese)," *J. Chin. Lang. Comput.*, vol. 13, no. 2, pp. 121–158, 2003.

[17] K. Uchimoto, K. Takaoka, C. Nobata, A. Yamada, S. Sekine, and H. Isahara, "Morphological analysis of the corpus of spontaneous japanese," *IEEE Trans. Speech Audio Process.*, vol. 12, no. 4, pp. 382–390, Jul. 2004.

[18] W. Jiang, L. Huang, and Q. Liu, "Automatic adaptation of annotation standards: Chinese word segmentation and POS tagging—A case study," in *Proc. Joint Conf. 47th Annu. Meeting Assoc. Comput. Linguistics/4th Int. Joint Conf. Natural Lang. Process. AFNLP*, 2009, pp. 522–530.

[19] W. Sun and X. Wan, "Reducing approximation and estimation errors for Chinese lexical processing with heterogeneous annotations," in *Proc. 50th Annu. Meeting Assoc. Comput. Linguistics*, 2012, pp. 232–241.

[20] J. Nivre and R. McDonald, "Integrating graph-based and transition-based dependency parsers," in *Proc. 46th Annu. Meeting Assoc. Comput. Linguistics*, 2008, pp. 950–958.

[21] R. Jin and Z. Ghahramani, "Learning with multiple labels," in *Proc. Adv. Neural Inf. Process. System*, 2002, pp. 897–904.

[22] M. Dredze, P. P. Talukdar, and K. Crammer, "Sequence learning from data with multiple labels," in *Proc. ECML/PKDD Workshop Learn. Multi-Label Data*, 2009, pp. 39–48.

[23] O. Täckström, R. McDonald, and J. Nivre, "Target language adaptation of discriminative transfer parsers," in *Proc. North Amer. Chapter Assoc. Comput. Linguistics*, 2013, pp. 1061–1071.

[24] Z. Li, J. Chao, M. Zhang, and W. Chen, "Coupled sequence labeling on heterogeneous annotations: POS tagging as a case study," in *Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics/7th Int. Joint Conf. Natural Language Process.*, 2015, pp. 1783–1792.

[25] A. Ratnaparkhi, "A maximum entropy model for part-of-speech tagging," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 1996, pp. 133–142.

[26] J. Lafferty, A. McCallum, and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proc. 18th Int. Conf. Mach. Learn.*, 2001, pp. 282–289.

[27] M. Collins, "Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2002, pp. 1–8.

[28] G. Heigold, H. Ney, P. Lehnen, T. Gass, and R. Schluter, "Equivalence of generative and log-linear models," *IEEE Trans. Audio, Speech Lang. Process.*, vol. 19, no. 5, pp. 1138–1148, Jul. 2011.

[29] M. Zhang, Y. Zhang, W. Che, and T. Liu, "Character-level Chinese dependency parsing," in *Proc. Annu. Meeting Assoc. Comput. Linguistics*, 2014, pp. 1326–1336.

[30] Z. Li, M. Zhang, W. Che, and T. Liu, "A separately passive-aggressive training algorithm for joint POS tagging and dependency parsing," in *Proc. COLING*, 2012, pp. 1681–1698.

[31] E. W. Noreen, *Computer-intensive Methods for Testing Hypotheses: An Introduction*. New York, NY, USA: Wiley, 1989.

[32] X. Qiu, J. Zhao, and X. Huang, "Joint Chinese word segmentation and POS tagging on heterogeneous annotated corpora with multiple task learning," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2013, pp. 658–668.

[33] X. Qiu, F. Ji, J. Zhao, and X. Huang, "Joint segmentation and tagging with coupled sequences labeling," in *Proc. Int. Conf. Comput. Linguistics*, Mumbai, India, 2012, pp. 951–964.

[34] M. Zhang, W. Che, Y. Shao, and T. Liu, "Jointly or separately: Which is better for parsing heterogeneous dependencies?" in *Proc. Int. Conf. Comput. Linguistics*, 2014, pp. 530–540.

[35] S. Ben-David and R. Schuller, "Exploiting task relatedness for multiple task learning," in *COLT*, pp. 367–580, 2003.

[36] R. K. Ando and T. Zhang, "A framework for learning predictive structures from multiple tasks and unlabeled data," *J. Mach. Learn Res.*, vol. 6, pp. 1817–1853, 2005.

[37] S. Parameswaran and K. Weinberger, "Large margin multi-task metric learning," in *Advances in Neural Information Processing Systems 23*, J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, Eds., 2010, pp. 1867–1875.

[38] C. Sutton, A. McCallum, and K. Rohanimanesh, "Dynamic conditional random fields: Factorized probabilistic models for labeling and segmenting sequence data," *J. Mach. Learn. Res.*, vol. 8, pp. 693–723, 2007.

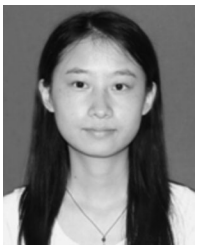
[39] H. Daume III, "Frustratingly easy domain adaptation," in *Proc. Annu. Meeting Assoc. Comput. Linguistics*, 2007, pp. 256–263.

[40] Y.-B. Kim, K. Stratos, R. Sarikaya, and M. Jeong, "New transfer learning techniques for disparate label sets," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, 2015, pp. 473–482.

[41] S. Riezler, T. H. King, R. M. Kaplan, R. Crouch, J. T. I. Maxwell, and M. Johnson, "Parsing the wall street journal using a lexical-functional grammarand discriminative estimation techniques," in *Proc. Annu. Meeting Assoc. Comput. Linguistics*, 2002, pp. 271–278.

[42] Z. Li, M. Zhang, and W. Chen, "Ambiguity-aware ensemble training for semi-supervised dependency parsing," in *Proc. 52nd Annu. Meeting Assoc. Comput. Linguistics*, 2014, pp. 457–467.
[43] Z. Li, M. Zhang, and W. Chen, "Soft cross-lingual syntax projection for dependency parsing," in *Proc. COLING*, 2014, pp. 783–793.
[44] G. Mesnil, "Using recurrent neural networks for slot filling in spoken language understanding," *IEEE/ACM Trans. Audio, Speech Lang. Process.*, vol. 23, no. 3, pp. 530–539, Mar. 2015.

Zhenghua Li received the Bachelor's, Master's, and Ph.D. degrees in computer science from Harbin Institute of Technology, Harbin, China, in 2006, 2008, and 2013, respectively. He joined Soochow University, Suzhou, China afterward, where he is currently an Associate Professor. His current research interests include semisupervised NLP, multilingual NLP, and structural data annotation.



Jiayuan Chao received the Bachelor's degree in computer science in 2013 from Soochow University, Suzhou, China, where she is currently a second-year postgraduate student working toward the Master's degree, and works on natural language processing for web data, including word segmentation, POS tagging, parsing, and so on.

Min Zhang received the Bachelor's and Ph.D. degrees in computer science from Harbin Institute of Technology, Harbin, China, in 1991 and 1997, respectively. In 2012, he joined the Soochow University, Suzhou, China, where he is currently a Distinguished Professor. From 1997 to 1999, he was a Postdoctoral Research Fellow in the Korean Advanced Institute of Science and Technology, Daejeon, South Korea. He began his academic and industrial career as a Researcher at Lernout & Hauspie Asia Pacific, Singapore, in 1999. He joined Infotalk Technology, Singapore, as a Researcher in 2001 and became a Senior Research Manager in 2002. He joined the Institute for Infocomm Research, Singapore, in 2003. His current research interests include machine translation, natural language processing, information extraction, large-scale text processing, intelligent computing, and machine learning.

Wenliang Chen received the Bachelor's degree in mechanical engineering and the Ph.D. degree in computer science from Northeastern University, Shenyang, China, in 1999 and 2005, respectively. Since 2013, he has been with Soochow University, Suzhou, China, where he is currently a Professor. Prior to joining Soochow University, he was a Research Scientist in the Institute for Infocomm Research, Singapore, from 2011 to 2013. From 2005 to 2010, he was an Expert Researcher in National Institute of Information and Communications Technology, Japan. His current research interests include parsing, machine translation, and machine learning.

Meishan Zhang received the Bachelor's degree in physics at the Chinese University of Geosciences, Wuhan, China, during 2000–2004, the Master's degree in computer science from the Institute of Software, Chinese Academy of Sciences, Beijing, China, during 2005–2008, and the Ph.D. degree in computer science from Harbin Institute of Technology, Harbin, China, during 2010–2015. He joined Heilongjiang University, Harbin, China, in 2016, where he is currently an Associate Professor. During 2015–2016, he was a Postdoctoral Research Fellow in Singapore University of Technology and Design.

Guohong Fu received the Bachelor's degree in automobile design, the Master's degree in mechanical design, and the Ph.D. degree in computer science from Harbin Institute of Technology, Harbin, China, in 1990, 1993, and 2001, respectively. In 2007, he joined Heilongjiang University, Harbin, China,, where he is currently a Professor. From 2001 to 2002, he was a Researcher in InfoTalk Technology, Singapore. In 2002, he joined the University of Hong Kong as a Postdoctoral Fellow and became an Honorary Assistant Professor in 2003. From 2006 to 2007, he was a Research Fellow at the City University of Hong Kong. His current research interests include natural language processing, Chinese computing, and text mining.