Learning Tag Dependencies for Sequence Tagging

Yuan Zhang[†,§*], **Hongshen Chen**[‡], **Yihong Zhao**[‡], **Qun Liu**[♢] and **Dawei Yin**[‡]
[†] Key Laboratory of Intelligent Information Processing
Institute of Computing Technology, Chinese Academy of Sciences
[‡] Data Science Lab, JD.com
[♢] ADAPT centre, School of Computing, Dublin City University
[§] University of Chinese Academy of Sciences
zhangyuan02@ict.ac.cn, chenhongshen,ericzhao@jd.com, yindawei@acm.org

Abstract

Sequence tagging is the basis for multiple applications in natural language processing. Despite successes in learning long term token sequence dependencies with neural network, tag dependencies are rarely considered previously. Sequence tagging actually possesses complex dependencies and interactions among the input tokens and the output tags. We propose a novel multi-channel model, which handles different ranges of token-tag dependencies and their interactions simultaneously. A tag LSTM is augmented to manage the output tag dependencies and word-tag interactions, while three mechanisms are presented to efficiently incorporate token context representation and tag dependency. Extensive experiments on part-of-speech tagging and named entity recognition tasks show that the proposed model outperforms the BiLSTM-CRF baseline by effectively incorporating the tag dependency feature.

1 Introduction

Sequence tagging problems have obtained much attention especially in the natural language processing community. Given a sequence of words, sequence tagging aims to predict a linguistic tag for each word such as the part-of-speech tag. Many natural language processing tasks can be cast into this problem, which include part-of-speech (POS) tagging, text chunking, and named entity recognition (NER) etc.

Among various frameworks, neural network based approaches prevail in these years, which take the benefits of representation learning and achieve state-of-the-art performance without massive hand-crafted feature engineering. [Collobert *et al.*, 2011] first introduced neural network for sequence tagging: a context window takes the sequences as input, then followed by a feed-forward neural network. When inferring tags, an independent tag prediction is used, where typically each token in a sentence is considered independently. Afterward, sequence tagging was further improved by recurrent neural network based methods, which is able to successfully capture the correlation and long term dependencies of tokens [Chen *et al.*, 2015; Lample *et al.*, 2016; Zhang *et al.*, 2015; Chen *et al.*, 2016; Vaswani *et al.*, 2016; Liu *et al.*, 2016; Zheng *et al.*, 2017]. Nevertheless, under the isolated tag criterion, tag dependencies are neglected. As [Collobert *et al.*, 2011] discloses, tag dependencies actually exist in a sentence. To overcome tagging dependency problem (the loss of tagging information in isolated tag criterion), a transition matrix between two consecutive tokens is introduced. In particular, it is formalized as a CRF layer [Lafferty *et al.*, 2001]. A global optimal tagging sequence is inferred over the entire tagging space. Significant gains are observed, comparing with the isolated tag criterion [Collobert *et al.*, 2011].

The$_1$	credit$_2$	card$_3$	you$_4$	wo$_5$	n't$_6$	want$_7$	to$_8$	do$_9$	without$_{10}$	.$_{11}$
DT	NN	NN	PRP	MD	RB	VB	TO	VB	IN	.
We$_1$	'll$_2$	have$_3$	to$_4$	do$_5$	without$_6$	.$_7$				
PRP	MD	VB	TO	VB	RB	.				

Figure 1: Illustration of Tag Dependencies.

However, transition matrix in CRF layer is usually position independent and only captures the neighboring tag dependencies, which are typically first order dependencies. Longer ranges of dependencies are not well managed, let along the interactions between the tag sequence and the word sequence. For instance, in figure 1, the part-of-speech tag of the 10^{th} word "without" in the first sentence is dependent on the 3^{rd} tag, where a preposition (IN) is associated with a noun phrase (NN), while it becomes adverb (RB) in the second sentence.

To overcome aforementioned problems, and capture different ranges of tag dependencies and word-tag interactions, in this paper, we propose a multi-channel model with three variants. The multi-channel model learns the word tagging by leveraging both the word context representation and tag dependency feature. In particular, we introduce a tag LSTM to tackle the long range tag dependencies and word-tag in-

*Work done when the first author was an intern at Data Science Lab, JD.com.

teractions. Then three different mechanisms are explored to incorporate the tag dependency feature: **shared tag LSTM** through integrating both features and **isolated tag LSTM** through feature integrated prediction or joint tagging decision.

To sum up, our contributions are as follows:

- We identify the problems of different ranges of tag dependencies and word-tag sequence interactions in sequence tagging tasks.
- A novel multi-channel model is proposed to manage the longer ranges of tag dependencies and the interactions between tag sequence and word sequence.
- Extensive experiments on three datasets verify the effectiveness of the proposed multi-channel model.

2 Background

Given a sentence $x = w_1, w_2, ..., w_N$, the aim of sequence tagging is to figure out the ground truth of tag sequence $y = t_1, t_2, ..., t_N$. For example, in POS tagging, the input tokens are words, and each word is annotated with a POS tag, while in named entity recognition, the allocated tag for each word indicates if a word is part of a named entity.

BiLSTM-CRF model prevails in recent years and achieves state-of-the-art performance on sequence tagging [Chen *et al.*, 2015; Lample *et al.*, 2016; Zhang *et al.*, 2015; Ma and Hovy, 2016; Chen *et al.*, 2016]. We will use BiLSTM-CRF model as our baseline. The model consists of mainly three layers:

- **Embedding layer** maps a word into embeddings, a distributed representation of words;
- **Word LSTM layer**, namely a bi-directional LSTM [Graves and Schmidhuber, 2005], utilizes a left-to-right and a right-to-left LSTM layer to extract word context representations;
- **Inference layer** employs a CRF structure to infer the most likely label for each word.

2.1 Embedding Layer

The embedding layer converts the one-hot encoding of each word w_i into a vector representation $\mathbf{e}_{w_i}$ through an embedding lookup table and a character level LSTM [Lample *et al.*, 2016].

2.2 Word LSTM Layer

Word LSTM layer sequentially extract word context representation $\mathbf{h}_i$ for each word w_i. Long Short-term Memory Networks (LSTM) have been shown to capture long-range dependencies, incorporating with a memory-cell to keep track of information and several gates to control the interaction (input, forget, and output) with the memory cell. We choose the standard implementation[Hochreiter and Schmidhuber, 1997] with both the left-to-right order and the right-to-left order. It generates a left context vector $\overrightarrow{\mathbf{h}_{w_i}}$ and a right context vector $\overleftarrow{\mathbf{h}_{w_i}}$ for word w_i. With both the left and the right context vector, word context representation $\mathbf{h}_i$ is obtained by concatenating them together: $\mathbf{h}_i^{word} = [\overrightarrow{\mathbf{h}_{w_i}}, \overleftarrow{\mathbf{h}_{w_i}}]$.

2.3 Inference Layer

Typically, a Conditional Random Field (CRF) [Lafferty *et al.*, 2001] is employed as inference layer to infer a tag t_i for each word w_i given the word context representation sequence $\mathbf{h}_1^{word}, ..., \mathbf{h}_N^{word}$. It learns the strong dependencies across output labels [Lample *et al.*, 2016] instead of making independent tagging decisions for each output. In particular, a transition scores matrix $\mathbf{A}$ is introduced. $A_{i,j}$ models the transition score from tag i to tag j. Thus, the score of a tag sequence y given an input sentence x is defined as:

$$s(x, y) = \sum_{i=0}^{N} A_{t_i, t_{i+1}} + \sum_{i=1}^{N} P_{i, t_i},$$

where $\mathbf{P}$ is the scores matrix computed by a softmax over all possible tags given word context representation sequence $\mathbf{h}_1^{seq}, ..., \mathbf{h}_N^{seq}$.

Then the probability for the tag sequence y is computed by a softmax over all possible tag sequences:

$$p(y|x) = \frac{\exp\{s(x, y)\}}{\sum_{y' \in Y} \exp\{s(x, y')\}},$$

where Y denotes all possible tag sequences for sentence x. When training, [Collobert *et al.*, 2011] maximize the log-probability of the golden tag sequence:

$$\log(p(y|x)) = s(x, y) - \log(\sum_{y'} \exp\{s(x, y')\}),$$

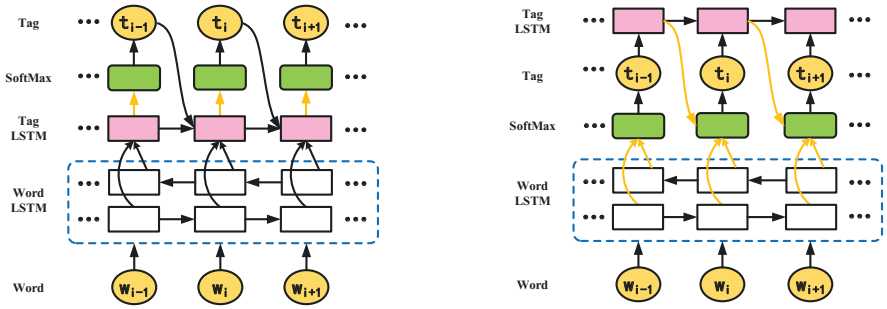
where the latter term can be computed in linear time taking advantage of the associativity and distributivity on the semiring.

Note that the transition score matrix $\mathbf{A}$ is position independent. It only models bigram interactions between two successive tags [Lample *et al.*, 2016; Pei *et al.*, 2014]. Longer ranges of tag dependencies and the interactions between word sequence and tag sequences are still not well managed.

3 Multi-channel Model

In this part, we describe a multi-channel model to address different ranges of tag dependencies and word-tag sequence interactions. BiLSTM-CRF constructs feature representation through a word LSTM. We refer it as single-channel model, as word LSTM is the only feature learning channel for making tagging decisions, whereas the *multi-channel model* leverage both word context representation and tag dependency feature together.

As mentioned previously, transition score matrix in BiLSTM-CRF model partially breaks independent tagging assumption within two consecutive tags. However, tag dependencies also exist in longer ranges as shown in figure 1. Although a simple higher order matrix can handle longer tag dependencies, it not only takes high computation cost, but also suffers from data sparsity. Therefore, in multi-channel model, we employ a tag LSTM to naturally handle different ranges of tag dependencies, add another feature learning channel, and further interact with word context. The model infers each tag based on both the word context representation



(a) Shared Tag LSTM (Model I) (b) Isolated Tag LSTM (Model II & III)

Figure 2: Multi-channel Models. The red blocks are LSTM layers, while the green ones are softmax layers. The yellow circles denote the embedding layer. The yellow information flow is the information channel.

and the tag dependency feature. In particular, for each word w_i, the corresponding tag is chosen based on both the word feature and the previous predicted tags. The tag sequence probability over the word sequence is then formulated as:

$$p(y|x) = \prod_{i}^{N} p(t_i|x, t_{<i}).$$

To incorporate the tag LSTM into the model, two methods—a shared tag LSTM and an isolated LSTM—are introduced in the follow sections.

3.1 Model I: Shared Tag LSTM

In multi-channel model, the tagging decision is made sequentially in a left-to-right order with respect to the word context and tagging history. Motivated by the text generation of neural translation model [Cho *et al.*, 2014; Sutskever *et al.*, 2014], we directly convert the chosen tag t_i of each word into a tag embedding $\mathbf{e}_{t_i}$, and then inject $\mathbf{e}_{t_i}$ into a tag LSTM. As tag dependency feature and word context representation flow together sharing a single LSTM, we name it **shared tag LSTM**. Figure 2a illustrates the multi-channel model using shared tag LSTM. A bi-directional LSTM is also employed to learn the word context representation $\mathbf{h}_i$.

Tagging inference is made independently based on the shared tag LSTM output $\mathbf{h}_i^{shared}$. The probability of tag t_i is computed by a softmax over all tag sets T based on $\mathbf{h}_i^{shared}$:

$$p(t_i|x, t_{<i}) = \frac{\exp\left(\mathbf{h}_i^{shared}\right)_{t_i}}{\sum\limits_{t_i' \in T} \exp\left(\mathbf{h}_i^{shared}\right)_{t_i'}},$$

where $(\cdot)$ is an affine transformation.

3.2 Isolated Tag LSTM

Shared tag LSTM is able to learn long term tag dependencies theoretically. However, in shared tag LSTM, the word context representation and tag dependency feature are jointly conveyed via a single channel. As a result, they might interfere with each other. Intuitively, word context representation plays a major role in tagging decision, and tag dependencies provides an auxiliary effect. With the shared tag LSTM scheme, the minor factor—tag dependencies—might be dominated by the major factor—word context. Consequently, tag dependencies cannot be modeled appropriately.

Hence, we design an **isolated tag LSTM** to separate both feature learning channels as shown in figure 2b. Word context representation and tag dependency feature are learned independently. With an isolated tag LSTM, we have two mechanisms to incorporate tag dependency features: one is feature integrated prediction, where both features are integrated before making tagging decision; the other is joint tagging decision, where each feature outputs tagging probabilities and then votes to make final tagging inference.

Model II: Feature Integrated Prediction

For isolated tag LSTM, it takes in the concatenation of tag embedding and word embedding $[\mathbf{e}_{t_i}, \mathbf{e}_{w_i}]$ for each word w_i, resulting with a tag dependency feature $\mathbf{h}_i^{iso}$, which enables tag LSTM to handle different ranges of tag dependencies and word-tag interactions purely.

One straightforward way to utilizing $\mathbf{h}_i^{iso}$ and word context representation $\mathbf{h}_i^{word}$ is to integrate them before prediction, so called *feature integrated prediction (FIP)* (figure 3):

$$\mathbf{h}_i = \mathbf{W}^{iso}\mathbf{h}_i^{iso} + \mathbf{W}^{word}\mathbf{h}_i^{word}.$$

The tag probability is computed through a softmax over $\mathbf{h}_i$.

Model III: Joint Tagging Decision

Inspired by ensemble averaging, to further enhance the learning of different ranges of tag dependencies, we explore another approach to sufficiently utilize word context representation and tag dependency feature, where both features make independent tagging probability estimations. Then those two probabilities are merged by a weighted average to make the final *joint tagging decision (JTD)*. The tagging probability is defined as:

$$\begin{aligned} & p(t_i|x, t_{<i}) \\ & \stackrel{\text{def}}{=} \lambda \cdot p(t_i|x) + (1-\lambda) \cdot p(t_i|t_{<i}) \\ & = \lambda \frac{\exp\left(\mathbf{h}_i^{word}\right)_{t_i}}{\sum\limits_{t_i' \in T} \exp\left(\mathbf{h}_i^{word}\right)_{t_i'}} + (1-\lambda) \frac{\exp\left(\mathbf{h}_i^{iso}\right)_{t_i}}{\sum\limits_{t_i' \in T} \exp\left(\mathbf{h}_i^{iso}\right)_{t_i'}}, \end{aligned}$$

where λ is the probability controlling the importance of each tagging estimation, and $(\cdot)$ is affine transformation. $p(t_i|x)$ is the tagging probability computed by word LSTM output $\mathbf{h}_i^{word}$ and $p(t_i|t_{<i})$ is tagging probability based on tag dependency feature $\mathbf{h}_i^{iso}$. In this formulation, when the gate λ

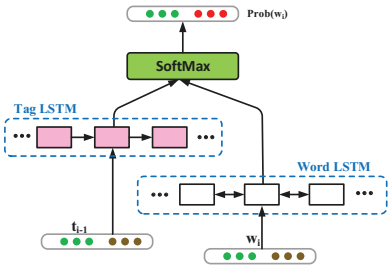


Figure 3: Model II: Feature Integrated Prediction.

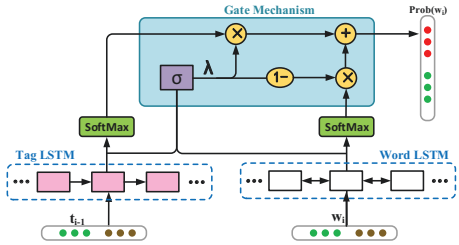


Figure 4: Model III: Joint Tagging Decision (JTD).

Corpus	Data	Sentences	Tokens	NEs
PD	train	96,750	5,958,471	282,851
	dev	1,000	60,147	2,944
	test	20,336	1,240,171	57,298
CoNLL03	train	14,987	204,567	23,376
	dev	3,466	51,578	5,906
	test	3,684	46,666	5,620
WSJ	train	38,219	912,344	-
	dev	5,527	131,768	-
	test	5,462	129,654	-

Table 1: Data statistics.

is close 1, the prediction probability is forced to ignore the tag LSTM and tagging decision is made entirely on word LSTM. While when λ is near to 0, the prediction probability approximates $p(t_i|t_{<i})$, tag LSTM plays a primary role.

The gate λ learns to effectively balance the involvement of tag LSTM and word LSTM when making the final tagging decision according to different scenarios. It is dynamically computed according to $\mathbf{h}_i^{iso}$ and $\mathbf{h}_i^{word}$:

$$\lambda = \sigma\left(\mathbf{h}_i^{iso}, \mathbf{h}_i^{word}\right),$$

where σ is logistic sigmoid function, $(\cdot)$ is affine transformation. When $\mathbf{h}_i^{word}$ fails to capture enough word context information and the gate λ is lower than 0.5, the model can rely more on the probability estimation from tag dependency feature $\mathbf{h}_i^{iso}$. What's more, similar to ensemble averaging method, the model variance can be reduced theoretically. In figure 4, $\mathbf{h}_i^{iso}$ and $\mathbf{h}_i^{word}$ first computes λ as prediction confidence and the tagging probabilities. Then, the final tagging probability is computed through an weighted average between $p(t_i|x)$ and $p(t_i|t_{<i})$.

Although tag LSTM and word LSTM are relatively independent, the two components are actually combined softly by λ, thus they can be trained jointly with respect to $p(t_i|x, t_{<i})$.

Tag LSTM extends in a left-to-right order and lacks the right context information. To enhance its prediction ability, we concatenate the right context vector $\overleftarrow{\mathbf{h}_{w_i}}$ (generated by word LSTM) with the tag LSTM output when computing the tagging probability. In particular, $p(t_i|t_{<i})$ is modified as:

$$p(t_i|t_{<i}) = \frac{\exp\left(\mathbf{h}_i^{iso}, \overleftarrow{\mathbf{h}_{w_i}}\right)_{t_i}}{\sum_{t_i' \in T} \exp\left(\mathbf{h}_i^{iso}, \overleftarrow{\mathbf{h}_{w_i}}\right)_{t_i'}}$$

4 Experiments

4.1 Data Sets

We study the tag dependencies mainly on two tagging tasks. One is part-of-speech (POS) tagging, where each word is allocated with a POS tag. The other is named entity recognition (NER), a classical task in the field of natural language processing, where each word is assigned a tag indicating if it belongs to part of a NE and which kind of NE. Table 1 list the data statistics of different corpus.

Part-of-Speech Tagging

We use the Wall Street Journal (*WSJ*) portion of Penn Treebank [Marcus *et al.*, 1993] for POS tagging task. We adopt the standard data set splits: section 0-18 as training data, 19-21 as development data and section 22-24 as test data.

Named Entity Recognition

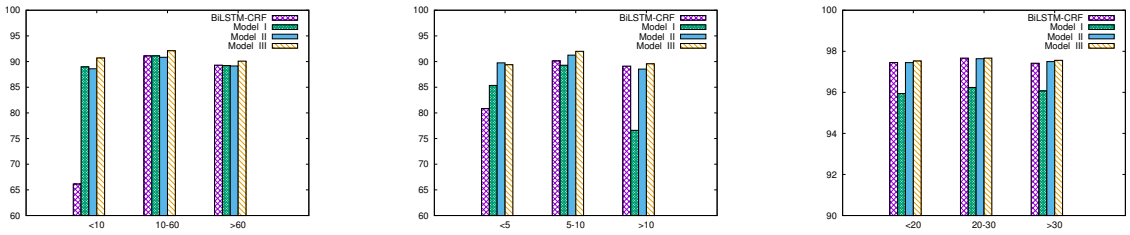
We also conduct NER experiments on two corpora, a Chinese corpus, People's Daily (*PD*)[1] and an English corpus, CoNLL 2003 corpus. For People's Daily, the data set is divided as 96750 sentences for training, 1000 sentences for validation, and 20336 sentences for test. CoNLL 2003[2] is a collection of news wire articles from the Reuters Corpus. We use the standard dataset split. The datasets contain four different types of named entities: locations, persons, organizations, and miscellaneous entities. A significant difference to POS tagging is that the number of NE tags are much less than POS tags. Another difference is that, in a sentence, most words are not belong to any named entities, leading NER a challenge task for learning tag dependencies. Following [Collobert *et al.*, 2011], we use the BIOES (Beginning, Inside, Outside, End, Singleton) tagging scheme. NE *label* represents *PER* (Person), *ORG* (Organization), etc. For a word, B-*label* denotes the beginning of a NE, I-*label* indicates that it is inside a NE but not the beginning, O means that it is not a part of a NE, E-*label* denotes the end of a NE, and S-*label* means singleton.

4.2 Experimental Settings

We use unigram character embedding as input of character level LSTM to learn the word representation, then the learned word representation together with the unigram word embedding are take as input of word LSTM for POS tagging and NER. The dimensionality of word embeddings is 100 for Chinese, and 300 for English. We adopt golden segmentation and random embedding initialization for Chinese. None of other

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

[2] https://www.clips.uantwerpen.be/conll2003/ner/



(a) NE on PD (b) NE on CoNLL 2003 (c) POS Tagging on WSJ

Figure 5: Performances on Different Lengths

Corpus	Model	F-score
PD	BiLSTM-CRF	88.17
	Model I	89.83
	Model II (FIP)	90.47
	Model III (JTD)	**90.79**
CoNLL03	[Collobert *et al.*, 2011]	89.59
	[Lin and Wu, 2009]	83.78
	[Lin and Wu, 2009]*	90.90
	[Passos *et al.*, 2014]	90.05
	[Passos *et al.*, 2014]*	90.90
	[Luo *et al.*, 2015]*+gaz	89.90
	[Luo *et al.*, 2015]*+gaz+linking	91.20
	[Chiu and Nichols, 2015]	90.69
	[Chiu and Nichols, 2015]*	90.70
	[Lample *et al.*, 2016]	90.94
	[Ma and Hovy, 2016]	91.21
	BiLSTM-CRF	88.83
	Model I	89.16
	Model II (FIP)	89.38
	Model III (JTD)	**90.20**
	BiLSTM-CRF+char	90.94
	Model I+char	90.57
	Model II+char (FIP)	90.62
	Model III+char (JTD)	**91.22**

Table 2: F-score for Named Entity Recognition. * indicates the usage of external labeled data.

manual features, like bi-gram or tri-gram word features are included. To avoid over-fitting, we adopt dropout strategy on the embedding layer with a rate of 0.5. The size of LSTM hidden state is set to 100. Training is done on every sample through stochastic gradient descent (SGD) with a learning rate of 0.005. To avoid gradient explosion problem, we set a gradient clipping strategy with a threshold of 1.0. For the proposed multi-channel models, we use the negative log likelihood objective function for training. When testing, we use beam-search to inference the best tag sequence and the beam size is set to 20.

4.3 Experimental Results

Main Performance

For NER, we report standard F-score to measure the performance. Table 2 presents performances for NER on both English and Chinese. Note that we don't leverage any external resources. We achieved a performance better than most previous published works and comparable with the model of [Luo *et al.*, 2015], where NER and entity linking tasks are jointly modeled, and a lot of hand-engineered features are incorporated including spelling features, WordNet clusters, Brown clusters, POS tags, chunks tags, as well as stemming and external knowledge bases like Freebase and Wikipedia.

Both shared tag LSTM and isolated tag LSTM outperform BiLSTM-CRF baseline without leveraging character information. In general, isolated tag LSTM (model II&III) achieves better performances, compared with the shared tag LSTM (model I). It confirms that isolated LSTM is able to capture long term tag dependencies better than shared LSTM. In shared one, the word context representation and tag dependency representation are conveyed through a single channel, where the two factors might interfere with each other.

Model	Accuracy
[Giménez and Marquez, 2004]	97.16
[Toutanova *et al.*, 2003]	97.27
[Manning, 2011]	97.28
[Collobert *et al.*, 2011]	97.29
[Santos and Zadrozny, 2014]	97.32
[Shen *et al.*, 2007]	97.33
[Sun, 2014]	97.36
[Søgaard, 2011]	97.50
[Ma and Hovy, 2016]	97.55
BiLSTM-CRF	97.52
Model I	97.34
Model II (FIP)	97.53
Model III (JTD)	**97.59**

Table 3: POS Tagging Accuracy on WSJ.

For POS tagging, Table 3 lists tagging accuracies of different systems. Both variants of isolated tag LSTM (model II&III) of multi-channel model perform better than BiLSTM-CRF baseline while shared tag LSTM (model I) is a bit lower. The gains on POS tagging is less than on NER. One possible difference lies in the NE tag sparsity, which lead NER more difficult than POS tagging. According to statistics in our corpus, only 7.4% words belong to named entities for Chinese NE and 21.2% for English NE. It is more difficult for the baseline system to capture long range discontinuous NE tag dependencies, but the discontinuous NE tag dependencies can be handled well by our proposed model. However, for POS tagging, the continuous POS tag dependencies are already captured by the transition matrix in CRF layer. The larger improvements on NER just verify the ability of our model in learning discontinuous NE dependencies.

Performance on Different Lengths

We further analyze the performance with respect to the different length of sequences. Figure 5 show the results. For NER, all three variants of multi-channel model consistently surpass BiLSTM-CRF model on short sentences (sentence length less than 10 on PD, length less than 5 on CoNLL2003). It indicates that the improvement of the proposed model over BiLSTM-CRF on short sentences is much larger than those on very long sentences. The NE tag dependencies, with short distances between two discontinuous tags, are captured very well by our proposed models but simply neglected by

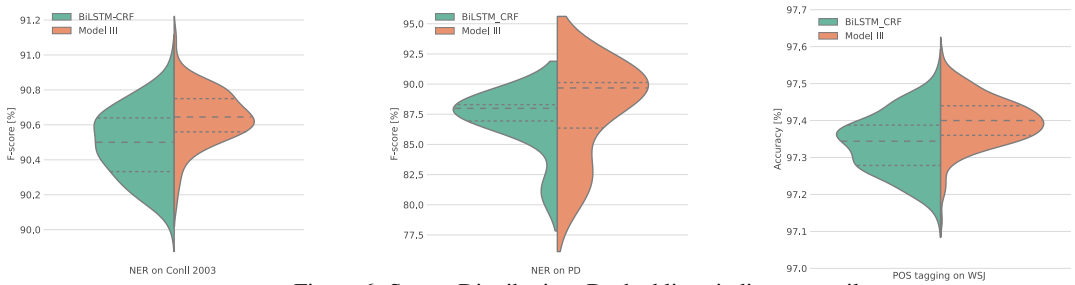


Figure 6: Scores Distribution. Dashed lines indicate quartiles.

Sent	Retailer	Sees	Pitfalls	In	Environmental	Push			
Tag	NN	VBZ	NNS	IN	JJ	NN			
λ	0.97	0.99	0.87	0.98	0.99	0.82			
Sent	More	Elderly	Maintain	Their	Independence				
Tag	JJR	JJ	VBP	PRP$	NN				
λ	0.98	0.96	0.95	0.96	0.86				
Sent	Italy	recalled	Marcello	Cuttitta					
Tag	LOC_S	O	PER_B	PER_E					
λ	1.00	0.91	1.00	0.58					
Sent	one-day	cricket	international	between	Pakistan	and	New	Zealand	
Tag	O	O	O	O	LOC_S	O	LOC_B	LOC_E	
λ	0.71	0.86	1.00	0.99	1.00	0.98	1.00	0.96	

Table 4: Qualitative Analysis.

BiLSTM-CRF, while for those dependencies with very long neighboring NE distances, capturing tag dependencies is also challenging for our models. For POS tagging, performances of different length are relatively comparable, while the isolated tag LSTM performs better than the shared one.

For shared tag LSTM and isolated tag LSTM, the latter one performs more stable and better than the former one on all lengths. We conjecture that under isolated LSTM setting, the feature spaces of two LSTMs are relatively independent and would not interfere with each other. Specifically, model III is relatively better than feature integration.

Qualitative Analysis

The weight λ in joint tagging decision of isolated tag LSTM actually indicates the confidence and balance between word LSTM tagging probability and tag LSTM prediction probability. If $\lambda > 0.5$, the final tagging decision relies more on word LSTM prediction ability, otherwise, the tag LSTM play a more important role. As shown in table 4, λ varies along the sentence, showing the effectiveness of both probabilities.

Score Distribution

To further verify the advantage of multi-channel model in comparison with BiLSTM-CRF model, we draw the scores distribution following [Reimers and Gurevych, 2017]. Figure 6 depicts that the quartiles of our model are above those of the BiLSTM-CRF model on all tasks, which suggests that the performance benefits from modeling tag dependencies.

5 Related Work

Traditional sequence tagging models are mostly linear statistical models, including Hidden Markov Models, Conditional Random Fields[Passos *et al.*, 2014; Cuong *et al.*, 2014; Luo *et al.*, 2015] and perceptron [Collins, 2002], which rely heavily on manual features and task specific resources. For instance, orthographic features and external resources such as gazetteers are widely used in NER [Ma and Hovy, 2016]. Due to the high cost of task-specific knowledge [Ma and Xia, 2014], these models are hardly to transfer to other tasks.

Recent years, non-linear neural networks are widely and successfully applied to sequence tagging, with the benefit of distributed word representations and rich feature representation learning. [Collobert *et al.*, 2011] used a simple feed-forward neural network with a sliding context window over the input sequence embeddings, and infers the tag sequence with a CRF layer. A tag transition matrix is used for inference, which makes the model effective. Most subsequent work on neural segmentation followed this method, improving the extraction of emission features by using more complex neural network structures. [Zheng *et al.*, 2013] applied this structure in word segmentation and POS tagging to get rid of the hand-crafted features. [Chen *et al.*, 2015] used a LSTM-CRF model to capture long-range sequence dependencies for word segmentation. [Chiu and Nichols, 2015] proposed to use BiLSTM to model word information in NER.[Lample *et al.*, 2016] used BiLSTM-CRF in NER. [Ma and Hovy, 2016] applied a similar structure. Most of the previous works utilize a CRF layer to restrict neighboring tag dependencies, leaving behind longer ranges of tag dependencies. Whereas we use tag LSTM to handle different ranges of tag dependencies. [Pei *et al.*, 2014] used a tensor neural network to model complicated interactions between context and tags for word segmentation, which is in spirit similar to us. They added the single previous one tag in the model without considering a longer history of all previous tags, while we handle all the predicted tag sequence and model the tag and context interactions along the tag prediction.

6 Conclusion

We study the tag dependencies in sequence tagging and identify the problems of the long range tag dependencies and tag-word sequence interactions. Novel multi-channel models are introduced to exploit different ranges of word and tag dependencies and their interactions. Experiments show that our specifically designed variants effectively handle the tag dependency, especially the discontinuous NE tag dependencies, and achieve significant improvements over the baselines.

Acknowledgements

The corresponding author is Hongshen Chen. We thank the anonymous reviewers for their constructive comments.

References

[Chen *et al.*, 2015] Xinchi Chen, Xipeng Qiu, Chenxi Zhu, Pengfei Liu, and Xuanjing Huang. Long short-term memory neural networks for chinese word segmentation. In *EMNLP*, pages 1197–1206, 2015.

[Chen *et al.*, 2016] Hongshen Chen, Yue Zhang, and Qun Liu. Neural network for heterogeneous annotations. In *EMNLP*, pages 731–741, 2016.

[Chiu and Nichols, 2015] Jason PC Chiu and Eric Nichols. Named entity recognition with bidirectional lstm-cnns. *arXiv preprint arXiv:1511.08308*, 2015.

[Cho *et al.*, 2014] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *EMNLP*, pages 1724–1734, 2014.

[Collins, 2002] Michael Collins. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *EMNLP*, 2002.

[Collobert *et al.*, 2011] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *JMLR*, 12(Aug):2493–2537, 2011.

[Cuong *et al.*, 2014] Nguyen Viet Cuong, Nan Ye, Wee Sun Lee, and Hai Leong Chieu. Conditional random field with high-order dependencies for sequence labeling and segmentation. *JMLR*, 15(1):981–1009, 2014.

[Giménez and Marquez, 2004] Jesús Giménez and Lluis Marquez. Svmtool: A general pos tagger generator based on support vector machines. In *ICLR*, 2004.

[Graves and Schmidhuber, 2005] Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional lstm networks. In *IJCNN*, 2005.

[Hochreiter and Schmidhuber, 1997] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[Lafferty *et al.*, 2001] John Lafferty, Andrew McCallum, and Fernando CN Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. 2001.

[Lample *et al.*, 2016] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural architectures for named entity recognition. In *NAACL*, pages 260–270, 2016.

[Lin and Wu, 2009] Dekang Lin and Xiaoyun Wu. Phrase clustering for discriminative learning. In *ACL*, 2009.

[Liu *et al.*, 2016] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. Recurrent neural network for text classification with multi-task learning. In *IJCAI*, 2016.

[Luo *et al.*, 2015] Gang Luo, Xiaojiang Huang, Chin-Yew Lin, and Zaiqing Nie. Joint named entity recognition and disambiguation. In *EMNLP*, 2015.

[Ma and Hovy, 2016] Xuezhe Ma and Eduard Hovy. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *ACL*, pages 1064–1074, 2016.

[Ma and Xia, 2014] Xuezhe Ma and Fei Xia. Unsupervised dependency parsing with transferring distribution via parallel guidance and entropy regularization. In *ACL*, 2014.

[Manning, 2011] Christopher D Manning. Part-of-speech tagging from 97% to 100%: is it time for some linguistics? In *Cicling*, pages 171–189, 2011.

[Marcus *et al.*, 1993] Mitchell P Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330, 1993.

[Passos *et al.*, 2014] Alexandre Passos, Vineet Kumar, and Andrew McCallum. Lexicon infused phrase embeddings for named entity resolution. *arXiv:1404.5367*, 2014.

[Pei *et al.*, 2014] Wenzhe Pei, Tao Ge, and Baobao Chang. Max-margin tensor neural network for chinese word segmentation. In *ACL*, pages 293–303, 2014.

[Reimers and Gurevych, 2017] Nils Reimers and Iryna Gurevych. Reporting score distributions makes a difference: Performance study of lstm-networks for sequence tagging. In *EMNLP*, pages 338–348, 2017.

[Santos and Zadrozny, 2014] Cicero D Santos and Bianca Zadrozny. Learning character-level representations for part-of-speech tagging. In *ICML*, pages 1818–1826, 2014.

[Shen *et al.*, 2007] Libin Shen, Giorgio Satta, and Aravind Joshi. Guided learning for bidirectional sequence classification. In *ACL*, volume 7, pages 760–767, 2007.

[Søgaard, 2011] Anders Søgaard. Semisupervised condensed nearest neighbor for part-of-speech tagging. In *ACL*, pages 48–52, 2011.

[Sun, 2014] Xu Sun. Structure regularization for structured prediction. In *NIPS*, pages 2402–2410, 2014.

[Sutskever *et al.*, 2014] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *NIPS*, pages 3104–3112, 2014.

[Toutanova *et al.*, 2003] Kristina Toutanova, Dan Klein, Christopher D Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *NAACL*, pages 173–180, 2003.

[Vaswani *et al.*, 2016] Ashish Vaswani, Yonatan Bisk, Kenji Sagae, and Ryan Musa. Supertagging with lstms. In *NAACL*, 2016.

[Zhang *et al.*, 2015] Meishan Zhang, Yue Zhang, and Duy-Tin Vo. Neural networks for open domain targeted sentiment. In *EMNLP*, pages 612–621, 2015.

[Zheng *et al.*, 2013] Xiaoqing Zheng, Hanyang Chen, and Tianyu Xu. Deep learning for chinese word segmentation and pos tagging. In *EMNLP*, pages 647–657, 2013.

[Zheng *et al.*, 2017] Suncong Zheng, Feng Wang, Hongyun Bao, Yuexing Hao, Peng Zhou, and Bo Xu. Joint extraction of entities and relations based on a novel tagging scheme. In *ACL*, 2017.