# Transferring from Formal Newswire Domain with Hypernet for Twitter POS Tagging

**Tao Gui**[1], **Qi Zhang**[1], **Jingjing Gong**[1],
**Minlong Peng**[1], **Di Liang**[1], **Keyu Ding**[2], **Xuanjing Huang**[1]
[1]Shanghai Key Laboratory of Intelligent Information Processing, Fudan University
[2]iFlytek Co., Ltd.
[1]{tgui16,qz,jjgong15,mlpeng16,xjhuang}@fudan.edu.cn
[2]kyding@iflytek.com

## Abstract

Part-of-Speech (POS) tagging for Twitter has received considerable attention in recent years. Because most POS tagging methods are based on supervised models, they usually require a large amount of labeled data for training. However, the existing labeled datasets for Twitter are much smaller than those for newswire text. Hence, to help POS tagging for Twitter, most domain adaptation methods try to leverage newswire datasets by learning the shared features between the two domains. However, from a linguistic perspective, Twitter users not only tend to mimic the formal expressions of traditional media, like news, but they also appear to be developing linguistically informal styles. Therefore, POS tagging for the formal Twitter context can be learned together with the newswire dataset, while POS tagging for the informal Twitter context should be learned separately. To achieve this task, in this work, we propose a hypernetwork-based method to generate different parameters to separately model contexts with different expression styles. Experimental results on three different datasets show that our approach achieves better performance than state-of-the-art methods in most cases.

## 1 Introduction

With the continuous growth of online communication, hundreds of millions of online conversational messages have become important resources for various applications such as real-time event detection (Sakaki et al., 2010), stock prediction (Bollen et al., 2011) and public health analysis (Wilson and Brownstein, 2009). Because these applications need to process natural language text, POS tagging, which is one of the fundamental natural language processing tasks, has become one of the basic pre-processing components of such applications. The performance of POS tagging may highly impact the results of these applications.

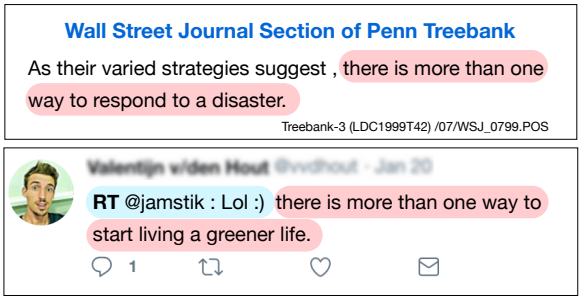



Figure 1: Examples of WSJ and tweets. Segments with red highlights can be regarded as the similar expressions. Segments with blue highlights correspond to expressions that cannot be learned from WSJ.

Most of the POS tagging methods that can achieve state-of-the-art performance are based on supervised learning algorithms (Gimpel et al., 2011). Although these methods can achieve good performance for in-domain data, their performance usually drops quickly when processing data from a domain that is different from that of the training data (Caruana and Niculescu-Mizil, 2006). To achieve better performance, we usually need to manually label a large amount of in-domain data. However, the task of constructing labeled data is time-consuming and tedious. Currently, various methods have been proposed to solve this problem using out-of-domain data, including domain adaptation (Daumé III, 2009; Gui et al., 2017), multi-task learning (Ben-David et al., 2007), and dual learning (Chandrasekaran et al., 2014).

Most existing methods aim to learn the shared representations or parameters, which can reduce the classification or regression model errors of each task/domain. However, these methods usually ignore the fact that each domain has domain-specific features that should not be shared. From

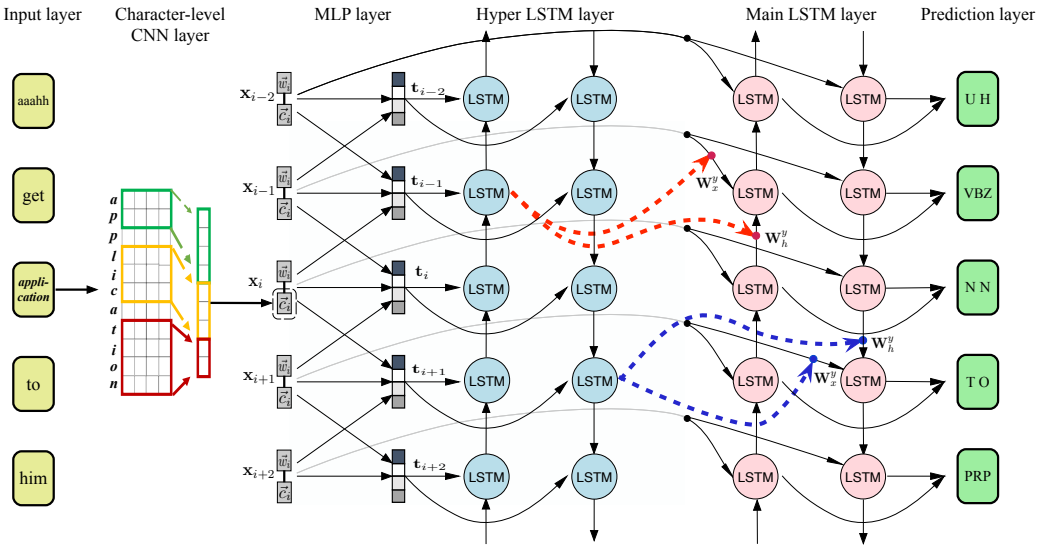



Figure 2: Model architecture of the proposed dynamic conversion neural networks.

the language characteristics, in spite of the expressions that are similar to the newswire text, Twitter has some informal expressions that cannot be shared, as shown in Figure 1. Hu et al. (2013) investigated the characteristics of language on Twitter and found that Twitter users not only tended to mimic the linguistic practices of traditional media, like news, they also appeared to be developing linguistically unique styles. We also believe that the tweets simply follow the standard language rules, and at some point eventually deviate from those. Thus, tweets are a combination of formal expressions and informal expressions with conversions between different expression styles.

Based on the above observations, we believe that annotated sentences of newswire text can be selectively used to help tag contextual segments of tweets, especially for formal expressions. To achieve this goal, in this work we adopt bidirectional long short-term memory (bi-LSTM) networks (Schuster and Paliwal, 1997), which have been successfully used for various sequence tagging problems. However, different from previous methods, the formal expressions and informal expressions in a sentence should be separately modeled. Inspired by recent work on dynamic parameter prediction (Ha et al., 2016), we propose a method to generate the context-specific parameters of the bi-LSTM based on different styles of context for POS tagging. We evaluated our models on three different corpora. The results demonstrated that the proposed method can benefit from annotated newswire text data and achieve competitive performance. In addition, we visualized the context distributions and the change of parameters. The visualization results verified the fact that different contexts produce different parameters for POS tagging.

The main contributions of the paper can be summarized as follows.

1. We study problems in the segment modeling method to apply domain adaptation to the POS tagging task. Based on the observations from a linguistic perspective, we found that there are many shared expressions between the newswire and tweets, and some expressions cannot be shared.

2. A novel neural network architecture based on the bi-LSTM was proposed to perform the task. Different parameters were applied in different contexts.

3. Experimental results demonstrated that the proposed method can benefit from different domains. We also conducted qualitative and quantitative analyses to show why our model can achieve better performance.

## 2 Approach

Twitter is responsible for colorful linguistic expressions, and full of the conversions between formal expressions and informal expressions. To address this problem, we propose the Dynamic Conversion Neural Networks (DCNN), which can dynamically generate different parameters for POS

tagging based on different contexts as shown in Figure 2. Our model mainly consists of four parts: (1) a CNN layer for extracting word representations $\mathbf{x}_i$, (2) an MLP layer for producing low-dimensional context representations $\mathbf{t}_i$, (3) a hyper LSTM layer for generating the weights $\mathbf{W}$ of a main LSTM, and (4) the main LSTM layer with dynamic parameters for POS tagging.

The architecture of the main network is the same with any sequence labeling model, which learns to map the word representations to the corresponding labels. However, the parameters of the main network can be modified according to our purpose. we use a low-dimensional context distribution vector $\mathbf{t}_i$ as the input of the hyper LSTM to generate the weights of the main LSTM. Thus, the weights of the main LSTM will be subject to a change in context vector. Therefore, the main LSTM can predict the POS tag based on the different parameters.

Ha et al. (2016) also proposed a HyperRNN network, in which the hyper net is influenced by the main net. This is inconsistent with our motivation. Different from (Ha et al., 2016), to make the parameters of the main LSTM totally controlled by the context distribution, we changed the architecture by cutting off the data path from the main LSTM to the hyper LSTM. This method can prevent the hidden states of the main LSTM from influencing the hyper LSTM. In addition, we add an extra layer of learning the context representations based on features returned by a CNN.

### 2.1 Word and Context Representations

Out-of-vocabulary words are frequently used in Twitter. Moreover, new symbols, abbreviations, and words are constantly being created. These make word representations difficult to address. Thus, robust methods should be used to extract the morphological and shape information from words.

Inspired by (Santos and Zadrozny, 2014), we adopted character-level convolutional neural networks (CNN layer) to tackle this problem, which can take all of the characters of the word into consideration and output important orthographic features (Santos and Zadrozny, 2014). Suppose that we are given the sentence $X = \{w_1, w_2, \dots\}$ with vocabulary $\mathcal{V}$ of words. We use multiple filters with varying widths to obtain the orthographic feature vector $\vec{c}_i$ for word $w_i$. Then, the orthographic feature vector $\vec{c}_i$ is concatenated to the word embedding $\vec{w}_i$ to form word representation $\mathbf{x}_i$ as the input of the main LSTM. Utilizing a bi-LSTM to model sentences, the model can extract the sequential relations and contextual information.

The context is a fixed window of words around target word. The context representations are learned by the MLP later. To do this, we apply a fully connected neural network, which takes sequential word representations in a fixed window as input to generate a low-dimensional vector $\mathbf{t}_i$ as follows:

$$\mathbf{t}_i = softmax(MLP[\mathbf{x}_{i-r}...\mathbf{x}_{i-1} \oplus \mathbf{x}_i \oplus \mathbf{x}_{i+1}...\mathbf{x}_{i+r}]), \quad (1)$$

where $[\cdot \oplus \cdot]$ represents concatenation operation. $r$ represents the length from central word $x_i$ to the edge of the window. $MLP$ is the multilayer perceptrons function, which transfers the context matrix to a low-dimensional vector. We apply $MLP$ to every window of contexts. $softmax$ denotes the softmax function that converts the context vector into a probability distribution. The goal is to learn an MLP layer that, given sequential word representations, estimates a distribution over the contexts.

### 2.2 Adaptive Weight Generation

The identical weights at each time step will limit the expressiveness of recurrent neural network (RNN) (Ha et al., 2016). To overcome the limitation, our model uses a small network (hyper network) taking low-dimensional context representations as inputs to dynamically generate the parameters of a large network (main network) for POS tagging. Different with (Ha et al., 2016), at every time step, the hyper LSTM only takes the context representation $\mathbf{t}_i$ as an input and generates the hidden state $\hat{\mathbf{h}}_i$ as an output. This hidden state $\hat{\mathbf{h}}_i$ is used to generate the weights for the main LSTM at the same time step. The hyper LSTM and main LSTM are jointly trained with backpropagation and gradient descent. Next, we will give a more formal description of the weight generation.

The hyper LSTM is a standard LSTM (Hochreiter and Schmidhuber, 1997), which takes context vectors as inputs and outputs hidden states. The

hyper LSTM is defined as follows:

$$\begin{bmatrix}\hat{\mathbf{g}}_i\\ \hat{\mathbf{i}}_i\\ \hat{\mathbf{f}}_i\\ \hat{\mathbf{o}}_i\end{bmatrix} = \left(\begin{bmatrix}\mathbf{W}_t^{\hat{g}}, \mathbf{W}_{\hat{h}}^{\hat{g}}\\ \mathbf{W}_t^{\hat{i}}, \mathbf{W}_{\hat{h}}^{\hat{i}}\\ \mathbf{W}_t^{\hat{f}}, \mathbf{W}_{\hat{h}}^{\hat{f}}\\ \mathbf{W}_t^{\hat{o}}, \mathbf{W}_{\hat{h}}^{\hat{o}}\end{bmatrix} \bullet \begin{bmatrix}\mathbf{t}_i\\ \hat{\mathbf{h}}_{i-1}\end{bmatrix} + \begin{bmatrix}\hat{\mathbf{b}}^{\hat{g}}\\ \hat{\mathbf{b}}^{\hat{i}}\\ \hat{\mathbf{b}}^{\hat{f}}\\ \hat{\mathbf{b}}^{\hat{o}}\end{bmatrix}\right) \quad (2)$$

$$\begin{aligned}\hat{\mathbf{c}}_i &= \phi(\hat{\mathbf{g}}_i) \odot \sigma(\hat{\mathbf{i}}_i) + \hat{\mathbf{c}}_{i-1} \odot \sigma(\hat{\mathbf{f}}_i)\\ \hat{\mathbf{h}}_i &= \sigma(\hat{\mathbf{o}}_i) \odot \phi(\hat{\mathbf{c}}_i),\end{aligned}$$

where $\phi$ denotes the $tanh$ function, and $\sigma$ is the $sigmoid$ function. $\odot$ and $\bullet$ represent the Hadamard product and matrix product, respectively. We assume that $\hat{y}$ is one of $\{\hat{g}, \hat{i}, \hat{f}, \hat{o}\}$. The hyper LSTM has $N_{\hat{h}}$ hidden units, and $N_t$ is the dimensionality of $\mathbf{t}_i$. Then, $\mathbf{W}_t^{\hat{y}} \in \mathbb{R}^{N_{\hat{h}} \times N_t}$, $\mathbf{W}_{\hat{h}}^{\hat{y}} \in \mathbb{R}^{N_{\hat{h}} \times N_{\hat{h}}}$, $\hat{\mathbf{b}}^{\hat{y}} \in \mathbb{R}^{N_{\hat{h}}}$ are the parameters of the hyper LSTM and stay invariable during one sentential sequence.

Inspired by (Ha et al., 2016), we adopted a *weight scaling vector* $\mathbf{d}$ which is a linear projection of $\hat{\mathbf{h}}_i$. $\mathbf{d}$ is used to linearly scale each row of the weight matrix in the standard LSTM. Because the context vector $\mathbf{t}$ is produced by the different contexts at each time step, the hidden state $\mathbf{h}_i$ and $\mathbf{d}_i$ will change corresponding to $\mathbf{t}_i$. Thus, the main LSTM can be modified as follows:

$$\begin{bmatrix}\mathbf{d}_{i,x}^y\\ \mathbf{d}_{i,h}^y\\ \mathbf{d}_{i,b}^y\end{bmatrix} = \begin{bmatrix}MLP_x^y(\hat{\mathbf{h}}_i)\\ MLP_h^y(\hat{\mathbf{h}}_i)\\ MLP_b^y(\hat{\mathbf{h}}_i)\end{bmatrix}$$

$$\begin{aligned}\mathbf{y}_i = &\left[\mathbf{d}_{i,x}^y \otimes \mathbf{W}_x^y, \mathbf{d}_{i,h}^y \otimes \mathbf{W}_h^y\right] \bullet \begin{bmatrix}\mathbf{x}_i\\ \mathbf{h}_{i-1}\end{bmatrix} \quad (3)\\ &+ \mathbf{d}_{i,b}^y \otimes \mathbf{b}^y\\ \mathbf{c}_i = &\phi(\mathbf{g}_i) \odot \sigma(\mathbf{i}_i) + \mathbf{c}_{i-1} \odot \sigma(\mathbf{f}_i)\\ \mathbf{h}_i = &\sigma(\mathbf{o}_i) \odot \phi(\mathbf{c}_i),\end{aligned}$$

where $\otimes$ represents the element-wise product with broadcasting. $\mathbf{y}$ is one of $\{\mathbf{g}, \mathbf{i}, \mathbf{f}, \mathbf{o}\}$. Generally, $N_{\hat{h}}$ and $N_t$ are much smaller than $N_h$ and $N_x$, respectively. Thus, the size of the parameters in the hyper LSTM is hundreds of times less than that of the standard LSTM.

According to the above functions, if the model is given contexts with different styles, it will generate different parameters for all kinds of gates in the main LSTM. Then, the outputs of the main LSTM are used to predict the POS tags of the central words with the cross entropy loss:

$$\mathcal{L}_{POS} = -\sum_i z_i * \log \hat{z}_i, \quad (4)$$

| Dataset | #Train | #Dev | #Test |
|---|---|---|---|
| RIT-Twitter | 10652 | 2242 | 2291 |
| NPSChat | 40497 | - | 4500 |
| ARK-Twitter | 26594 | - | 7707 |

Table 1: The statistics of the datasets used in our experiments, where # represents the number of tokens in datasets.

where $z_i$ is the one-hot vector of the POS tagging label corresponding to $\mathbf{x}_i$. $\hat{z}_i$ is the output of the top softmax layer: $\hat{z}_i = softmax(MLP(\mathbf{h}_i))$.

## 3 Experimental Setup

In this section, we will first detail the datasets we used. Then, we will describe several baseline methods, including a number of classic taggers and a series of deep learning sequence labeling methods.

### 3.1 Datasets

Following (Derczynski et al., 2013), we use RIT-Twitter (Ritter et al., 2011) as our main dataset. The RIT-Twitter was split into training, development and evaluation sets (RIT-Train, RIT-Dev, RIT-Test). The splitting method was shown in (Derczynski et al., 2013). In order to verify the validity of our model, we also tested it on two more datasets, NPSChat (Forsythand and Martell, 2007), and ARK-Twitter (Gimpel et al., 2011) using standard splits. The tag-sets of the RIT-Twitter and NPSChat are PTB-like, while that of the ARK-Twitter is specific. In order to use WSJ labeled data in experiments on ARK-Twitter, we performed the mapping from PTB tag-sets to ARK tag-sets, according to the PTB POS Tagging Guidelines (Santorini, 1990) and ARK Guidelines[1]. The mapping proceeded from fine to coarse.

For pretraining the word embedding, we constructed a dataset containing 30 million tweets, from Twitter using its API. We introduced a newswire dataset containing 1173K tokens as the written language dataset, namely the Wall Street Journal (WSJ) from the Penn TreeBank v3 (Marcus et al., 1993). During training, we mixed each of RIT-Twitter, NPSChat and ARK-Twitter with WSJ into three kinds of training data.

The detailed data statistics of the above datasets used in this work are listed in Table 1.

[1] http://www.ark.cs.cmu.edu/TweetNLP/

### 3.2 Competitor Methods

We applied several classic and state-of-the-art methods for comparison. In addition, we used a series of deep learning sequence labeling methods as baselines for comparison, as follows:

**Stanford POS Tagger** is a widely used part-of-speech taggers described in (Toutanova et al., 2003). It demonstrates the broad use of features and appropriate model regularization, which produces a superior level of performance (97.24%). In this work, we trained it using two different sets: sections 0-18 of the WSJ (**Stanford-WSJ**) and a mixed corpus of WSJ, IRC, and Twitter (**Stanford-MIX**).

**T-POS** (Ritter et al., 2011) adopts hierarchical clustering and Brown clustering methods to address the issue of OOV words and lexical variations. It also uses conditional random fields and other standard sets of features to perform the task. In this work, we trained it using three different sets: the WSJ (**T-POS-WSJ**), RIT-Train (**T-POS-RIT**) and a mixed corpus of WSJ, IRC, and RIT-Train (**T-POS-MIX**).

**GATE Tagger** (Derczynski et al., 2013) uses an approach that combines the available taggers for different tagsets. The tagger adopts a vote-constrained bootstrapping method with unlabeled data and assigns prior probabilities to handle of unknown words and slang.

**ARK Tagger** (Owoputi et al., 2013) is a system that reports the best accuracy on ARK-Twitter. It uses unsupervised word clustering and a variety of lexical features.

**TPANN** (Gui et al., 2017) applies adversarial networks and autoencoder to model labeled out-of-domain data, unlabeled in-domain data and labeled in-domain data and achieved the best performance on RIT-Twitter.

**Bidirectional LSTM (Bi-LSTM)** (Wang et al., 2015) has been widely used in a variety of sequence labeling tasks. In this work, we also evaluated it as a baseline.

**Bi-HyperLSTM** (Ha et al., 2016) was used as a substitute for the standard Bi-LSTM. What makes the Bi-HyperLSTM model different from the proposed model is that we used context distribution to generate the parameters of main LSTM.

### 3.3 Initialization and Hyperparameter

The word embeddings for all the models were initialized with the word2vec tool (Mikolov et al., 2013) on 30 million tweets. The other parameters excluding the word embeddings, such as the parameters in LSTM and MLP, were initialized by randomly sampling from a uniform distribution in [-0.05, 0.05].

The dimensionality of the word embedding was set at 200. The dimensionality for the randomly initialized character embedding was set at 25. We adopted a hyper LSTM with 160 hidden neurons to produce the weights of each gates of the main LSTM with 250 hidden neurons. The dimensionality of the context vector was set at 10.

Our DCNN could be trained end-to-end with backpropagation and gradient-based optimization was performed using the Adam update rule (Kingma and Ba, 2014) with learning rate 0.0001.

## 4 Results and Analysis

In this section, we will report the experimental results and a detailed analysis of the results for the three different datasets.

### 4.1 Evaluation on RIT-Twitter

The RIT-Twitter was introduced in (Ritter et al., 2011). This dataset uses a tagset based on the Penn Treebank tagset with several Twitter-specific tags: *retweets, @usernames, hashtags, and urls.*

Table 2 lists the results of our method compared with other methods on this dataset. The first part shows the results of the classic methods. From the result of Stanford-WSJ, we can see that although it can achieve a superior level of performance (97.24%) on the WSJ dataset, the accuracy drops significantly to 73.37% when applied to the Twitter dataset. If we add some in-domain data to the training set, the Stanford-MIX can improve by 10% compared to the Stanford-WSJ. The same phenomenon can be observed from T-POS tagger. If we apply more features, like clustering, bootstrapping and lexical features, the T-POS, GATE tagger and ARK tagger can achieve better performances. Although TPANN achieve accuracy of 90.92%, it incorporates additional a large amount of in-domain unlabeled data. Our method is more competitive because of the use of much fewer data sets.

| Methods | Training Set | RIT-Test | RIT-Dev |
|---|---|---|---|
| Stanford-WSJ (Toutanova et al., 2003) | | - | 73.37% |
| Stanford-MIX | | - | 83.14% |
| T-POS-WSJ (Ritter et al., 2011) | | 81.30% | |
| T-POS-RIT | | 84.55% | 84.83% |
| T-POS-MIX | | 88.30% | - |
| GATE Tagger (Derczynski et al., 2013) | | 88.69% | 89.37% |
| ARK Tagger (Owoputi et al., 2013) | | 90.40% | - |
| TPANN (Gui et al., 2017) | | 90.92% | 91.08% |
| Bi-LSTM | RIT-Train | 89.48% | 89.30% |
| Bi-LSTM* | | 89.31% | 90.37% |
| Bi-HyperLSTM | | 88.65% | 89.16% |
| Bi-HyperLSTM* | | 88.30% | 89.56% |
| DCNN | | 89.87% | 90.50% |
| Bi-LSTM | WSJ + RIT-Train | 90.09% | 90.37% |
| Bi-LSTM* | | 90.31% | 90.81% |
| Bi-HyperLSTM | | 90.57% | 90.41% |
| Bi-HyperLSTM* | | 90.44% | 90.54% |
| DCNN | | **91.18%** | **91.17%** |

Table 2: Token level accuracies of different methods on RIT-Test and RIT-Dev. The first part demonstrates the results of classic methods. The second part demonstrates a series of deep learning methods trained on RIT-train. The third part demonstrates the same deep learning methods train on the mixed dataset of RIT-train and WSJ. DCNN refers to our dynamic conversion neural network. Other models are described in the Section 3.2. The symbol * represents the model concatenates the context vectors with the word representations as inputs.

The second part shows the results of the deep learning methods trained on the RIT-Train dataset. We can see that if the sequence labeling methods are just trained on the RIT-Train dataset, their accuracies can exceed those of most conventional taggers. Thus, the deep learning methods are competitive and avoid feature engineering. Compared with other models, the DCNN achieved best performance among the models just trained on RIT-Train dataset.

The third part shows the results of the deep learning methods trained on the mixed dataset of the RIT-Train and WSJ. As observed, when we added the WSJ data to train the models, all of them could obtain different degrees of improvement. Moreover, our model could make better use of the out-of-domain data and obtained the best result. Compared with the ARK tagger, which achieved the previous best result in conventional methods, our model was almost 0.78% better. The error reduction rate was more than 8%. Our model also outperformed the TPANN, which incorporated additional unlabeled in-domain data.

From the perspective of utilizing a low-dimensional context vector, we provided the same information (word information and context information) for all of the deep learning models as shown in Table 2. However, except for the DCNN, the other models were incapable of utilizing the context information. Most of the models could not obtain obvious improvement. In contrast, our DCNN could make better use of the context information to generate more appropriate parameters for POS tagging. Next, we will analyze the behavior how the DCNN changes parameters when encountering different context vectors.

Intuitively, contexts with different language expression styles should be transformed into different vectors. Figure 3 visualizes the context distribution. Subfigure (a) shows the context vector extracted from WSJ. We can see that the formal expressions are mainly concentrated in the middle of the four dimensions. This phenomenon can be observed in the subfigure (b), where the formal expressions in the Twitter are concentrated in the middle of the same dimensions and the informal expressions are concentrated in another three dimensions. Notice that in our experimental setup, the dimensionality of the context vector is

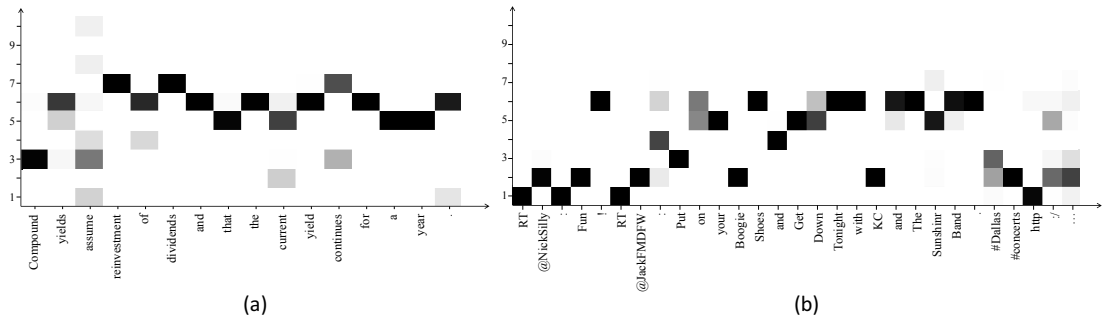
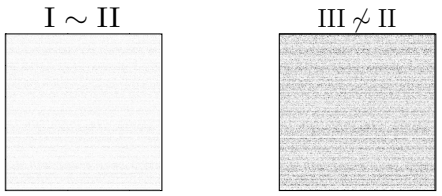



Figure 3: Visualization of context distribution. The left sentence comes from WSJ, and the right one comes from Twitter.




I ➤ *Compound yields assume reinvestment of dividends and that the current yield continues for a year.* (WSJ Formal)
II ➤ *Very grateful for another Monday and a new week full of endless possibilities ! Go World!* (Twitter Formal)
III ➤ *RT @NickSilly : Fun ! RT @JackFMDFW : Put on your Boogie Shoes and Get Down Tonight with KC and The Sunshine Band . #Dallas #concerts http :/ ...* (Twitter Informal)

Figure 4: Visualization of the comparison of weight matrices. We denote the weight matrix of output gate as $W^o_h$. Each subgraph represents the result of element-wise subtraction $|W^o_{h_i} - W^o_{h_j}|$, where $|\cdot|$ means absolute value, $i$ and $j$ are the sequence numbers of sentences. If the two sentences have a similar expression style, then the absolute value would be close to zero represented by white color. We use $i \sim j$ to represent it. On the contrary, We use $i \not\sim j$ to represent the different expression styles. We only visualize the weights on the last time step.

set to 10. However, the values of the last three dimensions are close to zero. Consequently, we set this hyperparameter to 7 and we achieve a higher accuracy of 91.27%.

Figure 4 shows how the weight matrix $W^o_h$ in output gate gets changed when the model inputs different kinds of contexts. Through making a comparison among the sentences I, II and III. we can find that although the sentences II and III are both from Twitter, whereas the sentence I is from WSJ, If the style of sentence II is close to that of sentence I, then the model will produce similar weight values to achieve the task. If the style of sentence I is different from that of sentence III, then the model will produce different parameters more suitable for Twitter-specific sentences. The similar phenomena can be found in other gates.

| **Methods** | **Acc.** |
|---|---|
| Forsyth (2007) | 90.8% |
| ARK Tagger | 93.4% ± 0.3% |
| Gui et al. (2017) | **94.1%** |
| Bi-LSTM(IRC) | 90.3% |
| Bi-LSTM(WSJ + IRC) | 93.2% |
| DCNN | **94.0%** |

Table 3: Accuracy comparison of different methods on NPSChat Corpus.

### 4.2 Evaluation on NPSChat

The NPSChat Corpus (Forsythand and Martell, 2007) is a PTB-POS annotated dataset of Internet Relay Chat (IRC) room messages from 2006. The corpus consists of 10,567 posts out of approximately 500,000 posts gathered from various online chat services in accordance with their terms of service. The authors of the corpus made several decisions during the process that were unique to the chat domain regarding some abbreviations, emotions and misspelled words. For example, LOL and :-) were frequently encountered in the chat messages. Because these expressions conveyed emotion, they were treated as individual tokens and tagged as interjections (UH).

Table 3 lists the results of different taggers evaluated on NPSChat. Our method was tested using the same setup as the experiments in (Forsyth, 2007). The training part contained 90% of the data. The testing part contained the remaining 10%. Based on the results, we can see that our method could achieve the best accuracy (94.0%), which was significantly better than 90.8% (Forsyth, 2007). They trained the tagger on a mix of several corpora tagged with the Penn Treebank tag set. Our method also outperformed

| Methods | Acc. |
|---|---|
| Gimpel et al. (2011) | 89.17% |
| Gui et al. (2017) | 92.8% |
| ARK Tagger | **93.2%** |
| ARK Tagger† | 92.38% |
| Bi-LSTM(OCT27) | 90.59% |
| Bi-LSTM(WSJ + IRC + OCT27) | 91.57% |
| DCNN(WSJ + IRC + OCT27) | **92.42%** |

Table 4: Accuracy comparison of different methods on ARK-Twitter Corpus. The symbol † represents ARK Tagger trained without tagdicts and namelists.

the ARK tagger, which applies various external corpus and features, e.g., Brown clustering, PTB, Freebase lists of celebrities, and video games.

### 4.3 Evaluation on ARK-Twitter

The ARK-Twitter that contains 34K tokens uses a novel tagset. The training set (OCT27) is provided in (Gimpel et al., 2011). It is a dataset of POS-tagged tweets consisting almost entirely of tweets sampled from one particular day (October 27, 2010). However, the test set was introduced in (Owoputi et al., 2013), and contains 574 tweets (DAILY547). The DAILY547 consists of one random English tweet from every day between January 1, 2011 and June 30, 2012. Thus, the distribution between the training set and test set may be slightly different. For example, a substantial fraction of the messages in the training data are about a basketball game that occurred on that day.

The results of the ARK tagger and TweetNLP Tagger in Table 4 are reported in (Owoputi et al., 2013). We can see that our method could significantly outperform the TweetNLP Tagger. However, our method was worse than the ARK tagger. By analyzing the incorrect results, we found that 20.3% of the errors occurred between nouns and proper nouns. Because our model does not incorporate any knowledge of proper nouns, it is difficult for it to recognize proper nouns from datasets. As reported in (Owoputi et al., 2013), if ARK-tagger does not add tag dictionary features and name list features, its performance will drop to 92.38%, which is lower than that of the DCNN. Thus, our model is also competitive when lacking of knowledge.

## 5 Related Work

At a very early time, Schmidhuber (1992) began to explore the concept of fast weights, in which one network can produce context-dependent weight changes for a second network (Schmidhuber, 1992, 1993). Moreover, they provided the theoretical possibility of a recurrent network version. Recently, numerous studies have been conducted in this field (Moczulski et al., 2015; Fernando et al., 2016). De Brabandere et al. (2016) introduced a new framework called the dynamic filter network where the filters in the CNN are generated dynamically. Ha et al. (2016) explored the use of this approach in recurrent networks. Our work uses a different mechanism to generate parameters, which can make the parameters subject to a change in context representations. we cut off the data path from the main LSTM to the hyper LSTM. This method can prevent the hidden states of the main LSTM from influencing the hyper LSTM.

Recently, deep learning has achieved promising results on POS tagging. Santos and Zadrozny (2014) used a CNN to construct a character-based model for English (PTB) and Portuguese. Wang et al. (2015) used the bi-LSTM on WSJ and reported a state-of-the-art performance. However, because of a lack of training data and an unconstrained writing style, these models encountered resistance in the implementation process on Twitter. In this work, we focused on the linguistic correlation between Twitter and newswire and took the linguistic characteristics into consideration. To selectively utilize out-of-domain data, we used a low-dimensional context vector to generate different parameters for text with different expression styles and obtained better results.

## 6 Conclusion

In this work, we study the problem of incorporating labeled newswire texts for Twitter POS tagging tasks. From a linguistic perspective, we find that Twitter users not only tend to mimic the formal expressions of traditional media, like news, but they also appear to be developing linguistically informal styles. Hence, we predict that labeled data from the newswire should selectively be used to help tag contextual segments of tweets. To achieve this task, we introduce a novel deep neural network architecture that can dynamically

generate different parameters based on different expression styles for POS tagging. To evaluate the performance of the proposed method, we compare the method with previous state-of-the-art methods on three different datasets. Experimental results demonstrate that the proposed method can achieve better performance in most cases. We also visualize some parameters learned for the proposed method to demonstrate the motivation for this work.

## Acknowledgments

The authors wish to thank the anonymous reviewers for their helpful comments. This work was partially funded by China National Key R&D Program (No.2017YFB1002104), National Natural Science Foundation of China (No.61751201, 61532011, 61473092, and 61472088), and STCSM (No.16JC1420401, 17JC1420200).

## References

Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. 2007. Analysis of representations for domain adaptation. In *NIPS*, pages 137–144.

Johan Bollen, Huina Mao, and Xiaojun Zeng. 2011. Twitter mood predicts the stock market. *Journal of Computational Science*, 2(1):1–8.

Rich Caruana and Alexandru Niculescu-Mizil. 2006. An empirical comparison of supervised learning algorithms. In *Proceedings of the 23rd ICML*, pages 161–168. ACM.

Bharath Chandrasekaran, Han-Gyol Yi, and W Todd Maddox. 2014. Dual-learning systems during speech category learning. *Psychonomic bulletin & review*, 21(2):488.

Hal Daumé III. 2009. Frustratingly easy domain adaptation. *arXiv preprint arXiv:0907.1815*.

Bert De Brabandere, Xu Jia, Tinne Tuytelaars, and Luc Van Gool. 2016. Dynamic filter networks. In *NIPS*.

Leon Derczynski, Alan Ritter, Sam Clark, and Kalina Bontcheva. 2013. Twitter part-of-speech tagging for all: Overcoming sparse and noisy data. In *RANLP*, pages 198–206.

Chrisantha Fernando, Dylan Banarse, Malcolm Reynolds, Frederic Besse, David Pfau, Max Jaderberg, Marc Lanctot, and Daan Wierstra. 2016. Convolution by evolution: Differentiable pattern producing networks. In *GECCO*, pages 109–116. ACM.

Eric N Forsyth. 2007. Improving automated lexical and discourse analysis of online chat dialog.

Eric N Forsythand and Craig H Martell. 2007. Lexical and discourse analysis of online chat dialog. In *ICSC 2007.*, pages 19–26. IEEE.

Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A Smith. 2011. Part-of-speech tagging for twitter: Annotation, features, and experiments. In *ACL: Human Language Technologies: short papers-Volume 2*, pages 42–47. Association for Computational Linguistics.

Tao Gui, Qi Zhang, Haoran Huang, Minlong Peng, and Xuanjing Huang. 2017. Part-of-speech tagging for twitter with adversarial neural networks. In *EMNLP*.

David Ha, Andrew Dai, and Quoc Le. 2016. Hypernetworks. In *ICLR 2016*.

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

Yuheng Hu, Kartik Talamadupula, Subbarao Kambhampati, et al. 2013. Dude, srsly?: The surprisingly formal nature of twitter's language. In *ICWSM*.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

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

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*, pages 3111–3119.

Marcin Moczulski, Misha Denil, Jeremy Appleyard, and Nando de Freitas. 2015. Acdc: A structured efficient linear layer. *arXiv preprint arXiv:1511.05946*.

Olutobi Owoputi, Brendan O'Connor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A Smith. 2013. Improved part-of-speech tagging for online conversational text with word clusters. ACL.

Alan Ritter, Sam Clark, Oren Etzioni, et al. 2011. Named entity recognition in tweets: an experimental study. In *EMNLP*.

Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. 2010. Earthquake shakes twitter users: real-time event detection by social sensors. In *Proceedings of WWW 2010*, pages 851–860. ACM.

Beatrice Santorini. 1990. Part-of-speech tagging guidelines for the penn treebank project (3rd revision). *Technical Reports (CIS)*, page 570.

Cicero D Santos and Bianca Zadrozny. 2014. Learning character-level representations for part-of-speech tagging. In *ICML-14*, pages 1818–1826.

Jürgen Schmidhuber. 1992. Learning to control fast-weight memories: An alternative to dynamic recurrent networks. *Neural Computation*, 4(1):131–139.

Jürgen Schmidhuber. 1993. A self-referential weight matrix. In *Proceedings of the International Conference on Artificial Neural Networks, Amsterdam*, pages 446–451.

Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.

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

Peilu Wang, Yao Qian, Frank K Soong, Lei He, and Hai Zhao. 2015. Part-of-speech tagging with bidirectional long short-term memory recurrent neural network. *arXiv preprint arXiv:1510.06168*.

Kumanan Wilson and John S Brownstein. 2009. Early detection of disease outbreaks using the internet. *CMAJ*, 180(8):829–831.