information

MDPI

Article

Bidirectional Long Short-Term Memory Network with a Conditional Random Field Layer for Uyghur Part-Of-Speech Tagging

Maihemuti Maimaiti [1,2], Aishan Wumaier [1,2,*], Kahaerjiang Abiderexiti [1,2] and Tuergen Yibulayin [1,2]

[1] School of Information Science and Engineering, Xinjiang University, Urumqi 830046, China; mahmutjan@xju.edu.cn (M.M.); kaharjan@xju.edu.cn (K.A.); turgun@xju.edu.cn (T.Y.)
[2] Xinjiang Laboratory of Multi-Language Information Technology, Xinjiang University, Urumqi 830046, China
* Correspondence: hasan1479@163.com or hasan1479@xju.edu.cn; Tel.: +86-136-599-13514

Received: 30 October 2017; Accepted: 27 November 2017; Published: 30 November 2017

Abstract: Uyghur is an agglutinative and a morphologically rich language; natural language processing tasks in Uyghur can be a challenge. Word morphology is important in Uyghur part-of-speech (POS) tagging. However, POS tagging performance suffers from error propagation of morphological analyzers. To address this problem, we propose a few models for POS tagging: conditional random fields (CRF), long short-term memory (LSTM), bidirectional LSTM networks (BI-LSTM), LSTM networks with a CRF layer, and BI-LSTM networks with a CRF layer. These models do not depend on stemming and word disambiguation for Uyghur and combine hand-crafted features with neural network models. State-of-the-art performance on Uyghur POS tagging is achieved on test data sets using the proposed approach: 98.41% accuracy on 15 labels and 95.74% accuracy on 64 labels, which are 2.71% and 4% improvements, respectively, over the CRF model results. Using engineered features, our model achieves further improvements of 0.2% (15 labels) and 0.48% (64 labels). The results indicate that the proposed method could be an effective approach for POS tagging in other morphologically rich languages.

Keywords: Uyghur; part-of-speech tagging; conditional random field; long short-term memory; bidirectional long short-term memory

1. Introduction

Part-of-speech (POS) tagging, which is a fundamental task in natural language understanding, has attracted considerable attention from researchers for various languages. In computational linguistics, this task involves labeling words in sentences with a unique POS tag according to their syntactic function in context. It plays an important role in natural language processing (NLP) and has been widely applied to a few high-level NLP tasks such as syntactic analysis, named entity recognition, and machine translation [1]. With the creation of social media and the development of electronic communication in Xinjiang, China, a large quantity of digital text in Uyghur is produced currently. The information extracted from these texts can be used for different NLP tasks such as POS tagging for Uyghur. Uyghur is an agglutinative and morphologically rich language. Therefore, this is an extremely challenging and interesting task. At present, state-of-the-art POS tagging accuracy is approximately 97% for English [2–5], approximately 96% for Chinese [6–9] on news text, and approximately 96.85% for Uyghur [10].

In Uyghur, words can be broadly divided into independent words, function words, and exclamatory words. Independent words include verbs and substantive words. Nouns, adjectives, numerals, quantifiers, pronouns, adverbs, and mimetic words belong to the class of substantive

words [11]. Function words include three kinds of words: conjunctions, prepositions, and particles. Uyghur is an agglutinative language, meaning that potentially several affixes (e.g., denoting person, number, case, or mood) are frequently attached to one word stem. Independent word affixes are divided into two main types: verbal affixes and substantive affixes. There are 150 verbal affixes and 65 different substantive affixes, which includes 49 noun affixes, 57 numeral affixes, and 55 adjective affixes. In theory, the number of various combinations of nominal affixes is 1502. However, according to recent statistical analysis [10], only 368 combinations appear in practice. For instance, there are 21 different affix variants of the word *weqe* ("accident," "event," or "incident") in the corpus used in this paper (as shown in Table 1).

Table 1. Different variants of *weqe* ("accident," "event," or "incident").

Variant	Translation	Suffixes
weqesini	the accident	*si*, *ni*
weqesimu	this accident also	*si*, *mu*
weqesige	on this accident	*si*, *ge*
weqesidin	from this accident	*si*, *din*
weqesi	an accident	*si*
weqede	in an accident	*de*
weqesining	of the accident	*si*, *ning*
weqelerde	in all accidents	*ler*, *de*
weqelerdin	from all these accidents	*ler*, *din*
weqelerni	all these accidents	*ler*, *ni*
weqelerning	of all these accidents	*ler*, *ning*

It is necessary to perform morphological analysis of Uyghur words before POS tagging. If POS tagging is performed without stemming, different variants of the same word will be identified as different words, and a large number of unknown words will appear, instead of different morphosyntactic variants (as shown in Table 1). To fully understand this issue, consider the following sentence (in Latin script):

alimjan ulugh alimimiz mehmud qeshqeri tughulghinining 1000-yilliqini xatirilesh ilmiy muhakime yighinida söz qildi.

Translation: Speech delivered by Alimjan at the 1000th anniversary conference of great scholar and lexicographer Mahmud al-Kashgari.

In this example, *alim* appears twice, as *alimjan* ("a person name") and *alimimiz* ("our scholar," or "our scientist"), and both instances are nouns. If *alim*, which is the more frequently used form, is the only form that appears in the training corpus, POS tagging would identify *alimimiz* as out of vocabulary (OOV). Unfortunately, (i) there is no open source stemming tool, (ii) the development cost of such a tool is high because the Uyghur language is a low-resource language and it has agglutinative and rich morphological features, and iii) the performance of stemming affects the performance of POS tagging.

To address this problem, we propose embedding words and characters and using syllable features in a bidirectional long short-term memory network with a conditional random field layer (BI-LSTM-CRF). This method combines handcrafted features with a neural network model and is described in Section 3.5. The experimental results are described in Section 4.

However, when morphological analysis is performed, the number of ambiguous phenomena may increase. For example, *at* means "name", "horse", or "shoot" and is either a verb or a noun, *atqin* means "shoot it" and is a verb, and *atlar* means "horses" and is a noun. After stemming, these words become the ambiguous word *at*, which is quite difficult to distinguish. To our knowledge, there is no study that addresses this problem. As mentioned earlier, in the Uyghur language, different word classes take different affixes (common affixes are also present), e.g., Uyghur nouns are inflected for number (singular and plural), case (nominative, accusative, dative, locative, ablative, genitive, similitude,

locative-qualitative, limitative, equivalence) [11], and person (first, second, third), and verbs are conjugated for tense: present and past; person; voice: causative and passive; aspect: continuous; mood. In addition, these affixes are typically attached to the stem in a relatively fixed order, e.g., the general order of attachment for nouns is number, person, and case. For example, the word *atlirimning* means "my horses'" and can split into affixes such as *at* (stem) + *lir* (plural) + *im* (first person) + *ning* (genitive case). Therefore, the affixes and their order in a word may refer to the class that the words belong to. It is better to use intra-word information to capture syntactic and semantic information on Uyghur POS tagging. We obtain word shape information in our proposed model using character embedding, which is described in Section 3.6.

Several studies on Uyghur POS tagging employ a small POS tag set; however, only a few studies consider a large POS tag set that can support high-level NLP tasks with richer information. Moreover, most existing Uyghur POS tagging models are linear statistical models, such as hidden Markov models (HMMs), maximum entropy models (MEMs), and n-gram models, all of which are limited to using only past and future features.

Our main contributions in this paper are as follows: (i) We apply long short-term memory (LSTM) networks, bidirectional LSTM (BI-LSTM) networks, an LSTM network with a conditional random field layer (LSTM-CRF), and the BI-LSTM-CRF model to Uyghur POS tagging. We experimentally compare the performance of the models on Uyghur POS tagging data sets and show that this task can be effectively performed by neural networks and that competitive tagging accuracy can be obtained without handcrafted features. Moreover, we show that because the BI-LSTM-CRF model considers word- and sentence-level information and can fully use past and future input features, it is an effective method of performing the POS tagging task in morphologically rich languages. (ii) For the first time, we examine the performance of easily applied engineered features, such as syllable- and suffix-based features, with character embedding and word embedding in Uyghur POS tagging and further improve the performance. (iii) We demonstrate that our approach can achieve state-of-the-art performance on small and large tag sets.

2. Related Works

In recent years, several POS tagging approaches have been developed. Collobert et al. [12] proposed a learning algorithm that can be applied to POS tagging; their system learns internal representations on a large unlabeled training data set instead of exploiting man-made features. Its results on common data sets indicate that such an approach performs well. Ptaszynski and Momouchi [13] applied a handcrafted dictionary to Ainu POS tagging. Evaluation on a training set provided positive results. Zheng, Chen, and Xu [14] explored the feasibility of performing Chinese POS tagging using a deep learning method, in which a multilayer neural network [15] is used to discover relevant features in input sentences. In addition, dos Santos et al. [2] proposed a convolutional neural network that learns the character-level representation of words and then associates them with a word-level representation to perform POS tagging. The evaluation of the system on the Wall Street Journal and Mac-Morpho corpora obtained accuracies of 97.32% and 97.47%, respectively. Labeau, Löser, and Allauzen [16] introduced a POS tagging application that can infer word representations from a character stream without using any man-made features. Pan, Yan, Zhou, Yu, and Guo [17] presented a Khmer automatic POS tagging method based on a cascaded CRF model that achieved an accuracy of 95.44% on an open corpus. Abdulkareem and Tiun [18] designed and implemented several POS tagging models (such as k-nearest neighbor, naïve Bayes, and decision tree models) for Arabic tweets and achieved an accuracy of 87.97%.

POS tagging for Uyghur has drawn attention in recent years. For instance, Tahir, Tursun, and Rozi [19] attempted to label POS automatically by adopting a bigram model based on an HMM model. Their tag set was designed for a speech synthesis system; data smoothing and unknown words were not considered. Najmidin, Mamat, and Ibrahim [20] presented n-gram-based POS tagging for Uyghur texts. The parameters and data smoothing of the n-gram model were analyzed, and the efficiencies

of bigram and trigram models were compared. Wang, Zu, and Litifu [21] investigated functional suffix strings and discussed the feasibility of POS tagging. Their results indicate that such a method is useful for Uyghur and other Turkic languages. To capitalize on the context features, Imam, Maimaiti, Ibrayim, and Abdurixit [22] employed perceptron training and Viterbi algorithms for POS tagging. Palidan and Fang [10] presented a maximum-entropy-based POS tagging model that combines the morphological features for multi-category word POS tagging and the data sparsity problem caused by inflection. Their results show that the suffix feature significantly improves the form type and unknown word tagging accuracy compared with other feature-based tagging models. Our model differs from the above models in that we use a powerful BI-LSTM-CRF network, which performs better than conventional statistical models, and that we use word- and character-level BLSTMs to collect longer context information and to extract more useful character-level features for the Uyghur language.

3. Methods

3.1. CRF Model

A CRF [23] is an undirected graphical model that has been successfully applied in several sequence labeling tasks including word segmentation, POS tagging, and named entity recognition. The CRF model can prevent the limited feature selection in HMMs and MEMs by considering the correlations between labels in neighborhoods [4]. Furthermore, it can acquire a global optimum via a process of global feature normalization.

Let an observation sequence that must be labeled be $S = \{s_1, s_2, \cdots, s_n\}$, where s_i is the vector of the ith word, and let $L = \{l_1, l_2, \cdots, l_n\}$ be a sequence of labels for S, where l_i is the label of the ith word. The linear-chain CRF model can then be written as

$$\mathrm{p}(L|S;W,b) = \frac{\prod_{i=1}^{n} \psi_i(l_{i-1}, l_i, S)}{\sum_{l' \in \varphi(S)} \prod_{i=1}^{n} \psi_i\left(l'_{i-1}, l'_i, S\right)} \tag{1}$$

where $\psi_i(l', l, S) = exp\left(W_{l'l}^T S_i + b_{l',l}\right)$ is the potential function corresponding to a label pair (l', l), W_T is the weight vector, b is the bias, and $\varphi(S)$ denotes the set of possible label sequences for S.

3.2. LSTM Model

An LSTM network is a special kind of recurrent neural network that is capable of learning long-term dependencies and can retrieve rich global information. An LSTM unit uses a series of multiplicative gates, such as input, output, and forget gates, and a memory cell to control the information flows in and out of the internal states of the network [24]. In addition, it determines the information that should be discarded or sent to the next time step. There are several slightly different versions of LSTM; here, we present a vanilla LSTM with the structure given in Figure 1.

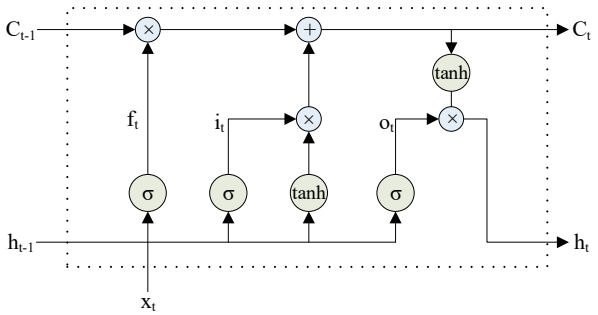

Figure 1. Simple LSTM model.

For time t, the multiplicative gates and memory are defined as follows:

$$\begin{cases} f_t = \sigma\left(W_f[h_{t-1}, x_t] + b_f\right) \\ i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \\ o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \\ C_t = f_t * C_{t-1} + i_t * \tanh(W_c[h_{t-1}, x_t] + b_c) \\ h_t = o_t * \tanh(C_t) \end{cases} \tag{2}$$

where $\sigma(\cdot)$ is the non-linear sigmoid function and f, i, o, C, and h are the vectors of the forget gate, input gate, output gate, memory cell, and hidden state, respectively. These vectors have the same size. Moreover, W_f, W_i, W_o, and W_C denote the weight matrices and b_f, b_i, b_o, and b_C represent the bias vectors.

3.3. Bidirectional LSTM Model

In sequence labeling tasks, it is beneficial to employ the previous and future input features over a given duration. However, the hidden state in a single forward LSTM captures previous features only and does not consider the future. Therefore, an elegant solution is BI-LSTM [25], which can be regarded as a stack of two LSTM layers. The previous features are extracted by a forward LSTM layer, and the future features are captured by a backward LSTM layer. In this way, we can effectively utilize the previous and future features; this alleviates the disambiguation problem mentioned in Section 1.

3.4. LSTM-CRF Model

For practical applications, the combination of a linear statistical model with a neural network has been proposed to prevent the problem that the performance of a neural network is largely determined by data. We implemented an LSTM-CRF [4] model consisting of an LSTM network and a CRF model. The basic idea is to use the LSTM layer to consider the previous input features and obtain sentence level tag information from the CRF layer. Therefore, the output is an optimal tag sequence instead of mutually independent tags.

Formally, $X = \{x_1, x_2, \cdots, x_n\}$ represents a generic input sequence, $y = \{y_1, y_2, \cdots, y_n\}$ represents the tag sequence for X, and P_{n*k} denotes a probability matrix, where k is the number of tag types. The optimal tag sequence can be obtained by maximizing the target function.

$$s(X, y) = \sum_{i=0}^{n} A_{y_i, y_{i+1}} + \sum_{i=0}^{n} P_{i, y_i} \tag{3}$$

where $P_{i,j}$ is the probability that the ith word is tagged as the ith tag and A is the state-transition matrix, where element $A_{i,j}$ is the probability of transferring from the ith tag to the jth tag.

3.5. BI-LSTM-CRF Model

Similar to the LSTM-CRF model, the BI-LSTM-CRF model is constructed from a BI-LSTM network and a CRF model. The output vectors of BI-LSTM are fed into CRF using the structure given in Figure 2.

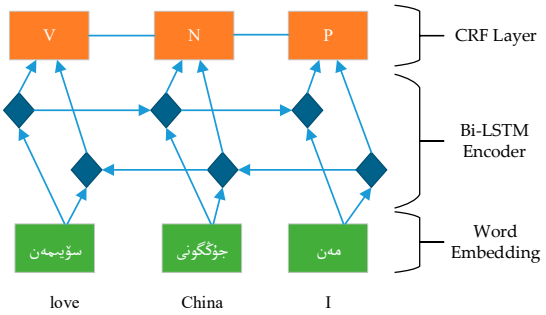

Figure 2. BI-LSTM-CRF model.

3.6. Features

3.6.1. Word Embeddings

In NLP, word embeddings [26], also known as distributed word representations, can capture the semantic and syntactic features of a word and reduce the requirement for handcrafted features [2]. We use randomly initialized word embeddings with 256 dimensions.

3.6.2. Character Embeddings

According to dos Santos et al. [2] and Lample et al. [27], character-level representations can extract morphological features from words and are extremely useful, particularly for morphologically rich languages. Our proposed method is similar to that of Lample et al. [27]; we randomly initialize a character lookup table with every character and feed every character embedding of the words from the character lookup table to the bidirectional LSTMs. Then, we concatenate the result of forward and backward representations to derive character-level word representations, which have 50 dimensions. Finally, we concatenate it with the word-level representation from a word lookup table to form the final word embeddings.

3.6.3. Engineered Features

We built a candidate feature set of useful features to determine which feature has the strongest influence on POS tagging. The candidate feature set consists of 11 features, as described below:

1. F_{wc}: The word feature, which represents the word itself.
2. F_{suffix}: The suffix of the word. We collected 153 unique suffixes.
3. F_s: The word without its suffix. The performance of the morphological analysis affects the accuracy of this stemming feature. However, we discovered that a word without a suffix typically corresponds to its stem. Hence, we substituted words without suffixes for stems.
4. F_{st}: The classification of the suffix. The set of suffixes is divided into eight types according to the POS type.
5. F_{bt}: The first-level POS label. We use this feature only when the tag set consists of 64 tags, and it is predicted using a 15-label tagging model with shared parameters.

In POS tagging, the syllable features of the words are crucial. It is better to extract syllables than stems in morphologically rich languages such as Uyghur. More accurate morphological information can be obtained through appropriate selection of syllables as features. Uyghur words can be composed of multiple syllables, which we express as follows:

$$Word = \{Syllable_1, Syllable_2, \cdots, Syllable_n\} \tag{4}$$

where n is the number of syllables in a word.

For most words, the initial syllables mainly describe the semantic information, which can be used to reduce the OOV. The ensuing syllables mainly describe the suffix information, which can be used to distinguish different contexts. Given such considerations, we performed a statistical analysis of syllables to select the best syllable features. The statistics of the number of syllables per word that appear in our corpus are summarized in Table 2.

According to Table 2, the majority of Uyghur words have a syllable length of seven or less. Moreover, the length of the syllables that contain grammatical information is between one and four. Therefore, depending on the syllable length of the current token, we extract features with different syllable lengths. These syllable features and their formulas are given below.

Table 2. Syllable [1] statistics.

Syllable Length	Frequency	Token Percentage (%)
1	2284	2.328
2	10,591	10.797
3	28,274	28.824
4	28,945	29.508
5	17,371	17.709
6	7014	7.151
7	2544	2.594
8	780	0.795
9	219	0.223
10	53	0.054
11	10	0.010
12	6	0.006
13	2	0.002
14	1	0.001

[1] A rule-based Uyghur syllabification tool (we also have a web service interface) developed by the natural language processing group of Xinjiang Laboratory of Multi-Language Information Technology is used. The accuracy is over 99%. We can provide it to anyone for research purposes.

6. F_{f2}: The first two syllables of a word. This feature can be calculated using the following formula:

$$F_{f2}(n) = \begin{cases} word, & n \leq 2 \\ \{Syllable_1, Syllable_2\}, & n > 2 \end{cases} \tag{5}$$

7. F_{la}: All syllables except the first one. This feature can be calculated using the following formula:

$$F_{la}(n) = \begin{cases} word, \ n = 1 \\ \{Syllable_2, \cdots, Syllable_n\}, \ n > 2 \end{cases} \tag{6}$$

8. F_{l4}: The last one to four syllables in a word. This feature can be calculated using the following formula:

$$F_{l4}(n) = \begin{cases} word, \ n = 1 \\ \{Syllable_n\}, \ n = 2 \\ \{Syllable_{n-1}, Syllable_n\}, \ 2 < n \leq 4 \\ \{Syllable_{n-2}, \cdots, Syllable_n\}, \ 4 < n \leq 8 \\ \{Syllable_{n-3}, \cdots, Syllable_n\}, \ n > 8 \end{cases} \tag{7}$$

9. F_{l3}: The last one to three syllables of a word. This feature can be calculated using the following formula:

$$F_{l3}(n) = \begin{cases} word, \ n = 1 \\ \{Syllable_n\}, \ n = 2 \\ \{Syllable_{n-1}, Syllable_n\}, \ 2 < n \leq 5 \\ \{Syllable_{n-2}, \cdots, Syllable_n\}, \ n > 5 \end{cases} \tag{8}$$

10. F_{l2}: The last one or two syllables of a word. This feature can be calculated using the following formula:

$$F_{l2}(n) = \begin{cases} word, & n = 1 \\ \{Syllable_n\}, & n = 2 \\ \{Syllable_{n-1}, Syllable_n\}, & n > 2 \end{cases} \quad (9)$$

11. F_{l1}: The last syllable in a word. This feature can be calculated using the following formula:

$$F_{l1}(n) = \begin{cases} word, & n = 1 \\ \{Syllable_n\}, & n \geq 2 \end{cases} \quad (10)$$

4. Experiments and Results

4.1. Data Sets

At present, there is no widely known uniform specification for Uyghur POS tagging sets; however, there have been several attempts to establish a tagging standard. For example, Xinjiang Laboratory of Multi-Language Information Technology and Xinjiang Normal University have independently created their tagging standards.

Xinjiang Laboratory of Multi-Language Information Technology created a manually annotated Uyghur POS tagging corpus that contains over 1.2 million tokens. Its tag set uses 15 first-level POS labels (as shown in Table 3), 71 second-level POS labels, and 51 third-level POS labels.

Table 3. First-level POS tagging set for Uyghur.

No.	Name	Tag	No.	Name	Tag
1	Noun	N	9	Interjection	E
2	Adjective	A	10	Verb	V
3	Numeral	M	11	Punctuation	Y
4	Quantifier	Q	12	Modal Particle	T
5	Adverb	D	13	Postposition	R
6	Pronoun	P	14	Affix	X
7	Mimetic Word	I	15	Latin word	LW
8	Conjunction	C			

Here, we use the corpus of Xinjiang Laboratory of Multi-Language Information Technology and its first- and second-level POS tag sets for modeling and conducting experiments (only 64 labels are used for the second-level POS tag set in this work; all punctuations are classified as one tag). The corpus statistics are summarized in Table 4.

Table 4. Corpora [2] statistics.

Datasets	Sentences	Tokens	Distinct Tokens
Training	40,000	743,955	78,477
Development	9641	180,931	34,451
Test	10,000	185,158	34,762

[2] This corpus is constructed by the natural language processing group of Xinjiang Laboratory of Multi-Language Information Technology. It is unpublished; please feel free to contact the author if you want to obtain the corpus.

4.2. Results and Discussion

This section presents the results of training CRF (http://github.com/zhongkaifu/CRFSharp), LSTM, LSTM-CRF, BI-LSTM, and BI-LSTM-CRF with identical feature sets. For CRF training,

the window size is 5 for F_{wc} and 3 for other features; all other parameters maintained at their default values. We used stochastic gradient descent with a fixed learning rate of 0.01 and a dropout rate of 0.5. Therefore, the differences in the results are entirely due to the different models.

4.2.1. Selection of Engineered Features

To determine which features are distinctive and more effective in POS tagging, we studied the effects of different engineered features on the CRF model.

Table 5 shows the accuracy of POS tagging for the CRF model with different feature combinations. It is quite clear that not all features are valid: a few features have a larger contribution to the accuracy than others. For instance, the accuracies achieved using the combinations $< F_{wc} + F_s + F_{suffix} >$ and $< F_{wc} + F_s + F_{suffix} + F_{st} >$ are not significantly different. Hence, considering the utilization of resources and for increasing the speed of training, not all features are employed in the follow-up experiments; the combination $< F_{wc} + F_s + F_{suffix} + F_{f2} + F_{l2} >$ is utilized.

Table 5. Performance of different engineered features (%).

F_{wc}	F_s	F_{suffix}	F_{st}	F_{f2}	F_{la}	F_{l4}	F_{l3}	F_{l2}	F_{l1}	F_{bt}	15 Labels		64 Labels	
											Test	Dev	Test	Dev
√	√										96.91	96.98	93.39	93.44
√	√	√									97.48	97.51	94.63	94.72
√	√	√	√								97.51	97.54	94.66	94.74
√	√	√	√	√							97.97	97.98	95.21	95.31
√	√	√	√	√	√						98.05	98.03	95.31	95.35
√	√	√	√	√	√	√					98.16	98.17	95.52	95.56
√	√	√	√	√	√	√	√				98.20	98.23	95.54	95.62
√	√	√	√	√	√	√	√	√			98.20	98.24	95.55	95.63
√	√	√	√	√	√	√	√	√	√		**98.23**	**98.29**	**95.61**	**95.66**
√	√	√	√	√	√	√	√	√	√	√	-	-	95.53	95.62

4.2.2. Comparison of Different Models

To assess the influence of models with respect to word and character features, we trained each model with the word and character features from the same data set. In addition, in Section 3.6.3, we presented several engineered features that were shown to be effective in Section 4.2.1. We then used these features jointly with the word embedding feature; each feature corresponds to a 30-dimensional embedding vector. For instance, F_{wn} is the word feature, F_i is the engineered feature, and i is the ith engineered feature. The sample can be defined as follows:

$$\begin{cases} sample = \{F_{wn} \oplus F_s\} \\ F_s = \{F_1 \oplus F_2 \oplus \cdots \oplus F_i\} \end{cases} \tag{11}$$

where $\oplus$ is the direct connection operation. After the concatenation operation, the sample contains word features, e.g., semantic and syntactic information, and additional morphological information. We also trained CRF, LSTM-CRF and BI-LSTM-CRF models with engineered features.

To verify the availability of the proposed method, we compare the results of different models in Table 6.

Table 6 provides the accuracies of the proposed models. It shows that the BI-LSTM-CRF model achieves accuracies of 98.41% and 95.74% on the 15- and 64-label test data sets, respectively, which is clearly higher than the accuracies of CRF and the other models. The reason for this phenomenon is that the CRF model typically requires several features, such as spelling and morphological features, to achieve good performance. In contrast, the LSTM based models, i.e., BI-LSTM, LSTM-CRF, and BI-LSTM-CRF, are more robust and less reliant on non-word features.

Table 6. Performance of the different models. "Word only" refers to the BI-LSTM-CRF model that uses word embeddings only, "Char only" refers to the BI-LSTM-CRF model that uses character-level embeddings only, "+ Feature" refers to the model that uses the engineered features which discussed in Section 4.2.1 (%).

Model	15 Labels		64 Labels	
	Test	Dev	Test	Dev
LSTM	97.03	97.06	92.87	93.00
BI-LSTM	97.87	97.90	93.78	93.86
CRF	95.72	95.90	91.79	91.88
LSTM-CRF	98.28	98.29	95.65	95.73
BI-LSTM-CRF	98.41	98.43	95.74	95.81
CRF + Feature	98.20	98.21	95.55	95.62
LSTM-CRF + Feature	98.58	98.58	96.16	96.19
BI-LSTM-CRF + Feature	**98.61**	**98.66**	**96.22**	**96.28**
BI-LSTM-CRF (Char only)	97.07	97.08	92.94	92.92
BI-LSTM-CRF (Word only)	97.33	97.42	94.33	94.40

When additional features are used, BI-LSTM-CRF outperforms CRF and LSTM-CRF and obtains the highest accuracy for every data set. For 15 labels, the accuracies of BI-LSTM-CRF are 0.41% and 0.45% higher than that of CRF, and for 64 labels, the accuracies are 0.67% and 0.66% higher than that of CRF. There is no significant difference between LSTM-CRF and BI-LSTM-CRF. System performance of all models are further improved after the engineered features are added. The improvement for 64 labels is significant, that is, an improvement of 0.48% on the test set that uses the BI-LSTM-CRF model. This is because the engineered features effectively reduce data sparseness and provide rich morphological information. This phenomenon demonstrates that it is useful to add a few handcrafted features to the BI-LSTM-CRF and LSTM-CRF models when the training data set is limited and the tag set is extremely large, and it could be more effective for morphologically rich languages such as Uyghur to jointly consider syllable- or morpheme-based representations that are larger than a character and require rich morphological information.

4.2.3. Comparison with Different Configurations

In this experiment, in order to understand the behavior of BI-LSTM-CRF in different conditions, we performed an error analysis on the testing set. Specifically, we partition each data set into in-vocabulary words (IV), out-of-vocabulary words (OOV), multi-category words (MC) and in-vocabulary-and-single-category words (IVASC). A word is considered IV if it appears in both the training and testing (or development) set, and OOV words are the ones do not appear in training set but in the testing (or development) set. MC words are the ones that can represent more than one part-of-speech in whole data set, while IVASC are the IV words that have only one part-of-speech. The statistics of the partition on each corpus are shown in Table 7.

Table 7. Statistics of the partition on each corpus.

Datasets	IV	OOV	15 Labels		64 Labels	
			MC	IVASC	MC	IVASC
All	-	-	1075	-	4241	-
Training	-	-	1062	-	4139	-
Development	24,161	10,290	776	23,395	3001	21,232
Test	24,383	20,379	816	23,579	3081	21,389

Table 8 illustrates the performances of BI-LSTM-CRF models on different subsets of words. The results of CRF model are provided as a baseline.

Table 8. POS results on test data set with BI-LSTM-CRF using different configurations, "Word" refers to word embeddings, "Char" refers to character-level embeddings and "Word-Char" refers to a combination of previous two architectures, "+ Feature" refers to the engineered features which discussed in Section 4.2.1 (%).

Models	15 Labels				64 Labels			
	IV	**OOV**	**MC**	**IVASC**	**IV**	**OOV**	**MC**	**IVASC**
Baseline	97.07	74.55	91.09	97.77	94.03	56.78	84.64	96.41
Baseline + Feature	98.82	88.37	**92.20**	99.60	96.68	77.84	**86.52**	99.26
Word	98.93	72.26	90.88	99.88	96.95	53.42	85.37	99.89
Char	97.52	90.15	89.84	98.42	93.81	79.33	83.50	96.45
Word-Char	99.01	89.04	91.03	99.94	96.92	77.35	85.23	99.90
Word-Char + Feature	**99.03**	**92.05**	91.02	**99.97**	**97.11**	**82.32**	85.96	**99.95**

We can see in Table 8 that, for both tag set, the Word-Char + Feature model performs best followed by Word-Char model. For the OVV words, the accuracy of the Char-based architectures, i.e., Char, Word-Char, and Word-Char + Feature, reaches large improvements over the baseline. This demonstrates that by adding character-based embeddings, BI-LSTM-CRF model more powerful on OOV words. For the IV words, the Char architecture reaches only small improvements over baseline on 15 labels, while on 64 labels the Char architecture is worse than the baseline. Interestingly, the character-level embeddings seem to have opposite effects on MC words. The Baseline + Feature model is competitive to the BI-LSTM-CRF models on MC words. We can also see in the results that engineered features largely improve the system performance, especially when dealing with OOV words. We think that the features address quite different information and add up well. This result suggests that, for the Uyghur POS tagging, the selected engineered features are very effective for both CRF and BI-LSTM-CRF models.

5. Conclusions

We studied the POS tagging problem as a sequence labeling problem. We applied LSTM network-based models to Uyghur POS tagging and reported the state-of-the-art tagging accuracy on small and large tag sets. Instead of using engineered features, the proposed method uses word- and character-based representations that capture morphological and orthographic information and achieves better accuracy than the CRF model, which relies heavily on handcrafted features and domain-specific knowledge. Furthermore, carefully selected engineered features were used to further improve the results for the CRF and BI-LSTM-CRF models.

Acknowledgments: This work was supported by the National Natural Science Foundation of China (grant numbers 61462083, 61262060, 61463048, 61662077, 61331011), 973 Program (grant number 2014cb340506), National Social Science Foundation of China (grant number 10AYY006), and Young Doctor Program of the Training Project of Young Scientific and Technological Innovation Talents [grant number QN2015BS004].

Author Contributions: Maihemuti Maimaiti and Aishan Wumaier conceived and designed the experiments; Maihemuti Maimaiti performed the experiments; Maihemuti Maimaiti and Kahaerjiang Abiderexiti analyzed the data; Tuergen Yibulayin contributed materials and data; Maihemuti Maimaiti wrote the paper; Kahaerjiang Abiderexiti helped write the paper; Aishan Wumaier and Tuergen Yibulayin revised the manuscript. All authors have read and approved the final manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Song, H.J.; Son, J.W.; Noh, T.G.; Park, S.B.; Lee, S.J. A cost sensitive part-of-speech tagging: Differentiating serious errors from minor errors. In Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, Jeju Island, Korea, 8–14 July 2012; pp. 1025–1034.

2. Santos, C.D.; Zadrozny, B. Learning character-level representations for part-of-speech tagging. In Proceedings of the 31th International Conference on Machine Learning, Beijing, China, 21–26 June 2014; pp. 1818–1826.
3. Garrette, D.; Baldridge, J. Type-supervised hidden Markov models for part-of-speech tagging with incomplete tag dictionaries. In Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Jeju Island, Korea, 12–14 July 2012; pp. 821–831.
4. Huang, Z.; Xu, W.; Yu, K. Bidirectional LSTM-CRF models for sequence tagging. *arXiv* **2015**, arXiv:1508.01991.
5. Wang, P.; Qian, Y.; Soong, F.K.; He, L.; Zhao, H. A unified tagging solution: Bidirectional LSTM recurrent neural network with word embedding. *arXiv* **2015**, arXiv:1511.00215.
6. Han, X.; Huang, D. Research on Chinese part-of-speech tagging based on semi hidden Markov model. *Chin. Comput. Syst.* **2015**, *36*, 2813–2816.
7. Hong, M.-C.; Zhang, K.; Tang, J.; Li, J.-Z. A Chinese part-of-speech tagging approach using conditional random fields. *Comput. Sci.* **2006**, *10*, 148–155.
8. Wei, O.; Wu, J.; Su, Y. Analysis and improvement of statistics-based Chinese part-of-speech tagging. *J. Softw.* **2000**, *11*, 473–480.
9. Zhang, H.P.; Yu, H.K.; Xiong, D.Y.; Liu, Q. HHMM-based Chinese lexical analyzer ICTCLAS. In Proceedings of the Second SIGHAN workshop affiliated with 41th ACL, Sapporo, Japan, 11–12 July 2003; pp. 184–187.
10. Palidan, T.; Fang, D. Fusion of morphological features for Uyghur part-of-speech tagging based on maximum entropy model. *J. Northwest Univ. (Nat. Sci. Ed.)* **2015**, *45*, 721–726.
11. Hämit, T. *Modern Uyghur Grammar (Morphology)*; Yıldız: Istanbul, Turkey, 2003; pp. 32–36.
12. Collobert, R.; Weston, J.; Bottou, L.; Karlen, M.; Kavukcuoglu, K.; Kuksa, P. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.* **2011**, *12*, 2493–2537.
13. Ptaszynski, M.; Momouchi, Y. Part-of-speech tagger for Ainu language based on higher order hidden Markov model. *Expert Syst. Appl.* **2012**, *39*, 11576–11582. [CrossRef]
14. Zheng, X.; Chen, H.; Xu, T. Deep learning for Chinese word segmentation and POS tagging. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, Seattle, WA, USA, 18–21 October 2013; pp. 647–657.
15. Collobert, R. Deep learning for efficient discriminative parsing. In Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, Fort Lauderdale, FL, USA, 11–13 April 2011; pp. 224–232.
16. Labeau, M.; Löser, K.; Allauzen, A.; von Neumann, R.J. Non-lexical neural architecture for fine-grained POS tagging. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, 17–21 September 2015; pp. 232–237.
17. Pan, H.; Yan, X.; Zhou, F.; Yu, Z.; Guo, J. A Khmer word segmentation and part-of-speech tagging method based on cascaded conditional random fields. *J. Chin. Inf. Process.* **2016**, *30*, 110–116.
18. Abdulkareem, M.; Tiun, S. Comparative analysis of ML POS on Arabic tweets. *J. Theor. Appl. Inf. Technol.* **2017**, *95*, 403–411.
19. Tahir, N.; Tursun, D.; Rozi, A. On technology of automatically tagging POS of Uyghur sentences oriented toward auto-division of prosodic layer boundary. *Comput. Appl. Softw.* **2011**, *28*, 165–168.
20. Najmidin, N.; Mamat, M.; Ibrahim, T. Experimental study of n-gram based Uyghur part of speech tagging. *Comput. Eng. Appl.* **2012**, *48*, 137–140.
21. Wang, H.; Zu, Y.; Litifu, T. The Uyghur POS-tagging method based on functional suffix strings. *J. Chin. Inf. Process.* **2013**, *27*, 179–183.
22. Imam, P.; Maimaiti, M.; Ibrayim, T.; Abdurixit, K. A perceptron approach to Uyghur POS tagging. *J. Chin. Inf. Process.* **2014**, *28*, 187–191.
23. Lafferty, J.; McCallum, A.; Pereira, F.C. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In Proceedings of the Eighteenth International Conference on Machine Learning, Williamstown, MA, USA, 28 June–1 July 2001; pp. 282–289.
24. Krause, B.; Lu, L.; Murray, I.; Renals, S. Multiplicative LSTM for sequence modelling. *arXiv* **2016**, arXiv:1609.07959.
25. Ma, X.; Hovy, E. End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF. *arXiv* **2016**, arXiv:1603.01354.

26. Turian, J.; Ratinov, L.; Bengio, Y. Word representations: A simple and general method for semi-supervised learning. In Proceedings of the Meeting of the Association for Computational Linguistics, Uppsala, Sweden, 11–16 July 2010; pp. 384–394.
27. Lample, G.; Ballesteros, M.; Subramanian, S.; Kawakami, K.; Dyer, C. Neural architectures for named entity recognition. *arXiv* **2016**, arXiv:1603.01360.