# A Gap-Based Framework for Chinese Word Segmentation via Very Deep Convolutional Networks

**Zhiqing Sun**
Peking University

**Gehui Shen**
Peking University

**Zhihong Deng**
Peking University

{1500012783, jueliangguke, zhdeng}@pku.edu.cn

## Abstract

Most previous approaches to Chinese word segmentation can be roughly classified into character-based and word-based methods. The former regards this task as a sequence-labeling problem, while the latter directly segments character sequence into words. However, if we consider segmenting a given sentence, the most intuitive idea is to predict whether to segment for each gap between two consecutive characters, which in comparison makes previous approaches seem too complex. Therefore, in this paper, we propose a gap-based framework to implement this intuitive idea. Moreover, very deep convolutional neural networks, namely, ResNets and DenseNets, are exploited in our experiments. Results show that our approach outperforms the best character-based and word-based methods on 5 benchmarks, without any further post-processing module (e.g. Conditional Random Fields) nor beam search.

## 1 Introduction

Unlike English, Chinese are written without explicit word delimiters, which makes word segmentation a fundamental and preliminary task in Chinese natural language processing. Recently, neural approaches for Chinese Word Segmentation (CWS) are attracting huge interest and a great deal of neural models have given competitive results to the best statistical models.

Previous neural approaches to CWS can be roughly classified into character-based and word-based. The former regards this task as a sequence-labeling problem, while the latter directly segments character sequence into words.
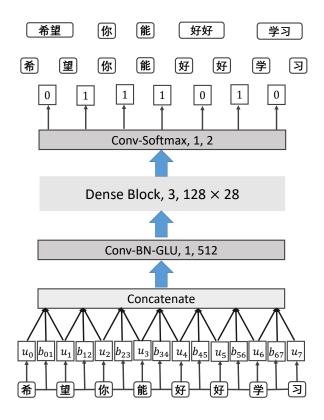


Figure 1: A Gap-Based Convolutional Network with a dense block that directly segment "希望你能好好学习" into "希望 (hope)　你 (you)　能 (can)　好好 (happily)　学习 (study) ".

Since Xue (2003), most character-based methods use {B, I, E, S} labels to denote the Beginning, Internal, End of a word and a Single-character word, respectively. To get the label scores for each character, tensor neural network (Pei et al., 2014), recursive neural network (Chen et al., 2015a), long-short-term-memory (RNN-LSTM) (Chen et al., 2015b) and convolutional neural network (CNN) (Wang and Xu, 2017) have been proposed. A transition score $[A]_{i,j}$ for jumping from $i$ to $j$ labels in successive characters is

then introduced to handle the label-label transition and give a structured output. With the help of transition scores, a $|s| \times 4$ label score sequence can be decoded into a $|s| \times 1$ label sequence for inference, where $|s|$ is the number of the characters in a sentence. A post-processing module such as Conditional Random Fields (CRF) (Lafferty et al., 2001) or maximum margin criterion (Taskar et al., 2004) can be used to enforce structure consistency and provide an objective function.

In word-based framework, Zhang et al. (2016) proposed a transition-based model which decodes a sentence from left-to-right incrementally. Cai and Zhao (2016) and Cai et al. (2017) proposed to score candidate segmented outputs directly. Yang et al. (2017) introduced partial-word into word-based models. For these word-based models, LSTMs (Hochreiter and Schmidhuber, 1997) or their variants are used for feature extraction, maximum margin criterion is used for training and beam search is used for inference.

Despite of the great success these methods achieved, there are still problems in both character-based and word-based frameworks. The main problem of the character-based framework is the use of post-processing modules, e.g., CRF or maximum margin criterion. In computer vision literature, current state-of-the-art models tend to be in an end-to-end scheme and directly get output from the neural networks (Ren et al., 2015; He et al., 2017), which in comparison make these post-processing modules seem overdesigned. Moreover, the use of {B, I, E, S} labels may produce redundant information. For example, "B" may be more similar to "S" or "I" than "E". These redundancies are not considered in all previous character-based models. An explanation we offer for the extensive using of these post-processing modules in the current state-of-the-art character-based models is that they are not good at capturing character combination features. Besides, the word-based models suffer from the problem of non-parallel, and they can only use the word segmentation information from the previous time steps.

In this paper, we propose a concise and efficient approach that overcomes the problems of character-based and word-based models: To improve the feature combination, we introduce deep convolutional neural networks to extract features for segmentation. Moreover, we directly predict segmentation for each gap between two consecutive characters. Thus, we need not structure our scores and avoid inference decoding. Because technically speaking, our framework are segmenting based on the gaps, we refer to our approach as Gap-Based Convolutional Networks (Gap-Based ConvNets). Figure 1 illustrates how our model works.

We evaluate Gap-Based ConvNets on 5 different benchmark datasets, namely CTB6, PKU, MSR, AS and CityU. As a pure supervised model, Our approach outperforms the current state-of-the-art pure supervised results on all of these benchmarks by a large margin, while are also competitive with the best semi-supervised results. We hope that our simple framework will open up a new way and serve as a solid baseline for CWS research. We also hope that this paper can help ease future research in other sequence-labeling tasks.

The contributions of this paper could be summarized as follows.

- We propose an end-to-end framework for CWS that directly classify the gaps between two consecutive characters and our results outperform the state-of-the-art character-based and word-based methods.

- Very deep neural networks are first introduced for CWS, in which we propose residual blocks and dense blocks to integrate multiple level character features. We also show that deeper neural networks can achieve better performance in CWS.

## 2 Gap-Based Framework

Our gap-based framework is described in detail in this section. First of all, if we consider segmenting a given sentence $s$ (character sequence) into chunks (words), the most intuitive idea is to predict whether to segment for each gap between two consecutive characters. That is to say, $|s| - 1$ predictions can determine the segmentation of the sentence $s$.

### 2.1 Gap Feature Representation

We follow previous works and consider both uni-character embedding and bi-character embedding when transforming the one-hot sparse discrete sequence $s$ into real-valued representation.

Two separated look-up tables are used to generate a 2-dimension representation $u$ of size $|s| \times e_u$

for uni-character embedding and a 2-dimension representation $b$ of size $(|s| - 1) \times e_b$ for bi-character embedding, respectively, where $e_u$ is the dimension of uni-character embedding and $e_b$ is the dimension of bi-character embedding.

The representation of a gap is then a concatenation of the uni-character embedding of its two consecutive characters and their bi-character embedding. Temporal convolutional layers (Conv) with kernel size 1 is applied to the concatenation, in order to integrate the uni-character and bi-character information. The output of the convolutional layer is regarded as the input to the feature extraction blocks.

We follow Gehring et al. (2017) and activate the convolutions with gated linear units (GLU) (Dauphin et al., 2016), which is a non-linearity operation that implement a gating mechanism over the convolution $Y = [AB]$:

$$v([AB]) = A \otimes \sigma(B) \qquad (1)$$

Moreover, Batch normalization (BN) (Ioffe and Szegedy, 2015) follows each convolutional layer $A$ before it is scaled by the gate $\sigma(B)$. Biases are not used in convolutions.

In Figure 1, $u_i$ and $b_j$ denote the uni-character representation and the bi-character representation, respectively, and "Concatenate" denotes a concatenating operation. "Conv-BN-GLU, k, d" denotes a temporal convolutional operation of kernel size $k$ and kernel number $d$, which is then batch normalized and activated by GLU.

## 2.2 Feature Extraction

Most of the previous applications of neural network to CWS use an architecture which is rather shallow (up to 5 layers). Wang and Xu (2017) proposed a 5-layer convolutional neural network. Chen et al. (2015b) compared their LSTM models in different layers and found their 1-layer LSTM model works best. Chen et al. (2017), Cai et al. (2017), Yang et al. (2017) and Zhou et al. (2017) also use 1-layer LSTM or bi-LSTM to extract features. These architectures are rather shallow in comparison to the deep convolutional networks which have pushed the state-of-the-art in computer vision. Besides, the use of 5-character context window (Chen et al., 2017; Zhou et al., 2017; Yang et al., 2017) in these models shows that their models are not good at capturing character combination features.
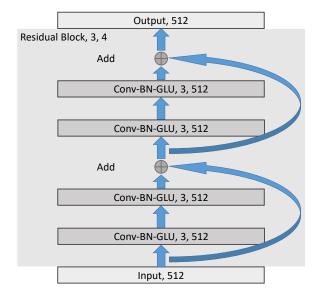


Figure 2: A 4-layer residual block with convolutional kernel size 3, input depth and output depth 1024, denoted by "Residual Block, 3, 4".

In this section, we propose deep feature extraction blocks, namely, residual blocks and dense blocks, to capture character combination features. To the best of our knowledge, this is the first time that very deep convolutional networks have been applied to sequence-labeling.

**Residual Block**  Traditional convolutional feed-forward networks connect the output of the $\ell$-th layer as input to the $(\ell + 1)$-th layer. We can represent this scheme as a layer transition: $x_\ell = H_\ell(x_{\ell-1})$. Observing that deep feed-forward networks are hard to train, ResNets (He et al., 2016) add a skip-connection that bypasses the non-linear transformations with an identity function:

$$x_\ell = H_\ell(x_{\ell-1}) + x_{\ell-1} \qquad (2)$$

Our $L$-layer wide residual block design follows Zagoruyko and Komodakis (2016). We define $H_\ell(\cdot)$ as a composition of two gated linear unit (GLU), where each of them split the kernel-size-3 convolutions (Conv) into two parts, namely, $A$ and $B$, and control the convolutional layer $A$ with the gate $\sigma(B)$. Each $A$ is batch normalized (BN) before the gate.

The Gap-Based ConvNets with a residual block is referred as Gap-Based ResNets in the rest of the paper. Figure 2 illustrates the layout of an example residual block schematically.
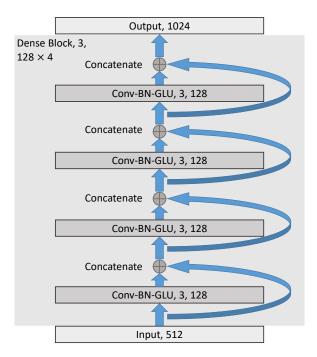
Figure 3: A non-bottleneck 4-layer dense block with a growth rate of 128, convolutional kernel size 3, input depth 512 and output depth $512 + 128 \times 4 = 1024$, denoted by "Dense Block, 3, 128 $\times$ 4". Each layer takes the concatenation of all preceding feature-maps as input.

**Dense Block** The densely connected architecture (Huang et al., 2017) is an extension of the residual architecture. Consider the gap representation $x_0$ are going to pass through the dense block and our dense block has $L$ layers. Then each layer implements a non-linear transformation $H_\ell(\cdot)$ to the concatenation of all preceding layers' feature maps:

$$x_\ell = H_\ell([x_0, x_1, \ldots, x_{\ell-1}]) \quad (3)$$

where $\ell$ indexes the layer and $[x_0, x_1, \ldots, x_{\ell-1}]$ refers to the concatenation of the feature-maps produced in layers $0, \ldots, \ell - 1$.

Similar to the residual blocks, we define $H_\ell(\cdot)$ as a gated linear unit (GLU) with a temporal convolutional layer (Conv) with kernel size 3 and Batch normalization (BN).

The number of the units in the temporal convolutional layer is referred as the growth rate $k$, as the concatenated layer depth grows by adding more layers. The Gap-Based ConvNets with such a dense block is referred as Gap-Based DenseNets.

A temporal convolutional layer with kernel size 1 is introduced as the bottleneck layer before each

convolutional layer with kernel size 3 to reduce the number of input feature maps. We set the number of kernel units in the bottleneck layers to $k$ as the same as the growth rate. Thus, our $H_\ell$ will be Conv1-BN-GLU-Conv3-BN-GLU.

Figure 3 illustrates the layout of an example non-bottleneck dense block schematically. It can be noticed that a dense block can explicitly concatenate multi-level character combination features.

### 2.3 Segmentation Prediction

At the end of the deep feature extraction blocks, a temporal convolutional layer with kernel size 1 and unit number 2 is performed. Then, a softmax layer is attached to make the output a probability distribution. Finally we get a $(|s| - 1) \times 2$ matrix for our prediction, and the values on the second dimension represent the prediction scores of "segmentation" and "no segmentation", respectively.

We ensemble several separately trained models by taking the means of their prediction scores.

## 3 Training

Because our model directly classifies the gaps into "segmentation" and "no segmentation", we can simply use the cross entropy (CE) as our loss function:

$$L = -\sum_{i=1}^{|s|-1} \sum_{x \in \{0,1\}} p_t^{(i)}(x) \log p_m^{(i)}(x) \quad (4)$$

where $p_t^{(i)}$ is the true probability distribution for the $i$-th gap, which may be $(0, 1)$ for "segmentation" or $(1, 0)$ for "no segmentation", and $p_m$ is the probability distribution predicted by our model for the $i$-th gap, namely, the output of the softmax layer.

Considering that there are plenty of annotation inconsistencies in the current datasets (Gong et al., 2017), we use a label smoothing (Szegedy et al., 2016) with factor $\beta = 0.1$ to prevent overfit and boost the model robustness.

We use Adam (Kingma and Ba, 2014) with a mini-batch size of $n$ to optimize model parameters, with an initial learning rate $\alpha_1 = 0.002$ in the first 8000 steps and $\alpha_2 = 0.0002$ in the rest steps. Model parameters are initialized by normal distributions as Glorot and Bengio (2010) suggested. A dropout for the gap representation with dropout rate $p_1$ is used to reduce overfitting. We

4

| Model Setting | P | R | F |
|---|---|---|---|
| Gap-Based ResNets | | | |
| $L = 4$ | 96.1 | 95.9 | 96.0 |
| $L = 12$ | 96.2 | 96.1 | 96.1 |
| Gap-Based DenseNets | | | |
| $L = 4, k = 128$ | 96.0 | 96.0 | 96.0 |
| $L = 12, k = 128$ | **96.4** | 96.2 | **96.3** |
| $L = 12, k = 256$ | 96.3 | 96.2 | 96.2 |
| $L = 28, k = 128$ | 96.2 | **96.4** | **96.3** |

Table 1: Results of model selection on CTB6.

| Context | P | R | F |
|---|---|---|---|
| both | **96.4** | **96.2** | **96.3** |
| bi-character | 94.9 | 94.6 | 94.8 |
| uni-character | 95.8 | 95.8 | 95.8 |
| oracle-combined | **97.4** | **97.3** | **97.4** |

Table 2: Influence of different gap representation.

set pretrained bi-character embedding fixed and only fine-tune the pretrained uni-character embedding.

## 4 Experiments

### 4.1 Experimental Settings

**Data** We use Chinese Treebank 6.0 (CTB6) (LDC2007T36) (Xue et al., 2005) as our main dataset. We follow the official document and split the dataset into training, development and test data. In order to verify the robustness of our model, we additionally evaluate our models on SIGHAN 2005 bakeoff (Emerson, 2005) datasets, where we randomly split 10% data from the training data as development data.

We replace all the punctuation with "<PUNC>", English characters with "<ENG>" and Arabic numbers with "<NUM>" for all text. We also add "</s>" symbol to the beginning and the end of a sentence.

We apply word2vec (Mikolov et al., 2013) on Chinese Gigaword corpus (LDC2011T13) to get pretrained embedding of uni-characters and bi-characters. We choose 50 for both uni-character embedding size $e_u$ and bi-character embedding size $e_b$. Notice that as we do not need word embedding, we do not have to automatically segment the corpus by other segmentors. The use of word embedding in the gap-based framework is left for future works.

**Evaluation** The standard word precision, recall and F1 measure (Emerson, 2005) are used to evaluate segmentation performances. The prediction scores of 4 separately trained models with same settings are ensembled for the evaluation of different model architectures and hyper-parameters.

**Fine-tune** We fine-tune the hyper-parameters on the development data. We almost keep all the

hyper-parameters to be the same when we evaluate our models on different datasets, except that the batch size is set to 256 for AS dataset while to 64 for other datasets, and dropout rate $p_1$ is set to 0.3 for CTB6 and PKU datasets while to 0.2 for other datasets.

### 4.2 Model Analysis

We perform development experiments on CTB6 dataset to verify the usefulness of various configurations and different loss objectives, respectively.

#### 4.2.1 Model Selection

We evaluate our Gap-Based Convolutional Networks with residual blocks or dense blocks, with different number of layers $L$. The main results on CTB6 are shown in Table 1, where we mark our best results in **boldface**. The dimension of the first convolutional layer (gap representation) are set to 512.

We find that our 12-layer and 28-layer Gap-Based DenseNets both achieve the best performance, with growth rate 128. Therefore, to get both the performance and the speed, we only use a 12-layer Gap-Based DenseNet with growth rate 128 in the following experiments.

#### 4.2.2 Gap Representation

We compare different gap representations. The results are shown in Table 2, where "both" represents the original model, "uni-character' ' and "bi-character' ' represent the gap representation only with uni-character embedding and bi-character embedding, respectively. And "combined" represents the combined results of a pure uni-character model and a pure bi-character model.

As can be seen from the table, by removing uni-character and bi-character embedding, the F-score decreases to 94.8 and 95.8, respectively. We can find that the uni-character embedding are more robust and useful than bi-character embedding, which is an opposite conclusion to Yang et al. (2017). We believe this is due to the sparsity of the bi-character embedding.

| Context | Sample | Correct |
|---|---|---|
| uni-character | 他 才 又 有 机会 站到 火车 修复 的 第一 线 | √ |
| bi-character | 他 才 又 有 机会 站到 火车修复 的 第一 线 | × |
| uni-character | 美 不 胜收 的 阿里山 | × |
| bi-character | 美不胜收 的 阿里山 | √ |

Table 3: Example segmentation results of different character contexts. The correct segmentation should be "他(he) 才(just) 又(again) 有(have) 机会(chance) 站到(stand) 火车(train) 修复(repair) 的('s) 第一(first) 线(frontier)" and "美不胜收(beautiful) 的(of) 阿里山(Ali Mountain)".

In addition, we conduct a simple experiment on the combination of pure uni-character and pure bi-character models.

As the pure uni-character models and the pure bi-character models have different representations for the gaps, their predictions for the segmentation are also independent. They produce different distribution of segmentation errors, which provide the opportunity for them to learn from each other, as shown in Table 3.

Therefore, we provide an oracle-combined model to combine the results from these two models. For the combined results, we accept the segmentation results that both pure uni-character model and pure bi-character model accept and use what we call "oracle" to determine whose results to accept when in divergence of views. The "oracle" represents that we can always make the right choices. The combined results are also listed in Table 2.

We find that while performs better than pure uni-character and pure bi-character models, our oracle-combined results also outperform the original model by a large margin, which suggests that uni-character information and bi-character information are quite complementary, and maybe the combination of uni-character information and bi-character information before the feature extraction blocks makes the original model not able to sufficiently exploit the feature combination ability of deep neural networks.

However, "oracle" means that we need to know the answer beforehand, which is impossible in practice. Therefore, a practical way to combine the results of pure uni-character and pure bi-character models is left for future investigation.

| Scheme | P | R | F |
|---|---|---|---|
| gap-based | **96.4** | **96.2** | **96.3** |
| character-based CRF | 96.1 | 96.3 | 96.2 |
| character-based greedy | 96.1 | 96.1 | 96.1 |

Table 4: Influence of different CWS frameworks.

| Framework | Sample | Correct |
|---|---|---|
| gap-based | 中国 证券 市场 目前 仍 处于 初创 阶段 | √ |
| character-based | 中国 证券 市场 目前 仍 处于 初 创 阶段 | × |

Table 5: An example segmentation result in different frameworks. The correct segmentation should be "中国(Chinese) 证券(stock) 市场(market) 目前(currently) 仍(still) 处(in) 于(the) 初创(start) 阶段(period)".

### 4.2.3 Comparison with Character-Based Framework

Both our gap-based model and character-based models regard CWS task as a sequence-labeling task. However, a big difference between them is that while character-based models have to give 4 scores for each character and use a post-processing module, the gap-based models only need to give a binary classification for each gap.

In order to see the efficiency of the gap-based schemes, we compare the character-based scheme and the gap-based scheme in the same neural network architecture, namely, Gap-Based DenseNets. To combine the character-based scheme with the original architecture, we revise the architecture by

- We represent the character information by the character's uni-character embedding and the bi-character embedding of its consecutive characters.

- We use a convolutional layer and softmax layer with 4 units instead of 2, which represent the scores for {B, I, E, S} in the character-based sequence-labeling scheme.

To fully investigate the performance of character-based scheme, following (Zhou et al., 2017), we train the network in two approaches. The first approach use a transition matrix to model the tag dependency and CRF for structured inference, and the second use a greedy loss that directly classifies the characters into {B, I, E, S}. The results are shown in Table 4, where "character-based CRF" represents the first ap-

| F1 score | CTB6 | PKU | MSR | AS | CityU |
|---|---|---|---|---|---|
| our proposed | **96.3** | **96.0** | **97.9** | **96.1** | **96.9** |
| Cai et al. (2017) | - | 95.8 | 97.1 | 95.6 | 95.3 |
| Zhou et al. (2017) baseline | 94.9 | 95.0 | 97.2 | - | - |
| Wang and Xu (2017) W2VBE-CONV | - | 95.9 | 97.5 | - | - |
| Cai and Zhao (2016) | - | 95.5 | 96.5 | - | - |
| Chen et al. (2015a) GRNN* | - | 94.5 | 95.4 | - | - |
| Chen et al. (2015b) LSTM* | - | 94.8 | 95.6 | - | - |
| Wang et al. (2014) dual | - | 95.3 | 97.4 | 95.4 | 94.7 |
| Sun (2010) | - | 95.2 | 96.9 | 95.2 | 95.6 |
| Zhang and Clark (2007) | - | 94.5 | 97.2 | 94.6 | 95.1 |

Table 6: Main results on CTB6 and SIGHAN 2005 bakeoff datasets with other best pure supervised results. Results with * are from the implementation of Cai and Zhao (2016), which do not use an external dictionary of Chinese lexicons(Chen et al., 2015a,b).

| F1 score | CTB6 | PKU | MSR | AS | CityU |
|---|---|---|---|---|---|
| our proposed | **96.3** | 96.0 | 97.9 | **96.1** | **96.9** |
| Zhou et al. (2017) WCC embeddings† | 96.2 | 96.0 | 97.8 | - | - |
| Yang et al. (2017) multi-pretrain* | 96.2 | 96.3 | 97.5 | 95.7 | 96.9 |
| Wang and Xu (2017) WE-CONV* | - | **96.5** | **98.0** | - | - |
| Zhang et al. (2016) neural* | 95.0 | 95.1 | 97.0 | - | - |
| Zhang et al. (2016) hybrid* | 96.0 | 95.7 | 97.7 | - | - |

Table 7: Main results on CTB6 and SIGHAN 2005 bakeoff datasets with other best semi-supervised results. The results marked with * and †use auto-segmented data to get pretrained word embeddings and character embeddings, respectively.

proach and "character-based greedy" represents the second.

We can see that our gap-based scheme have competitive results to the character-based scheme, while we do not have a post-processing module. Moreover, our simple classification scheme makes us more convenient to use a great deal of well-developed tricks in the large supervised learning literature, e.g., label smoothing and confidence penalty.

Table 5 gives an example segmentation result that our gap-based framework gives the correct segmentation, while the answer from the character-based framework is wrong. It shows that our gap-based segmentation framework is able to fix some mistakes that character-based framework will make.

### 4.3 Final Results

In addition to CTB6 dataset, which has been the most commonly adopted by recent segmentation research, we additionally evaluate our models on the SIGHAN 2005 bakeoff datasets, to examine

cross domain robustness. Among these datasets, PKU and MSR datasets are in simplified Chinese, while AS and CityU datasets are in traditional Chinese and we have to map them into simplified Chinese before segmentation.

Our final results are shown in Table 6 and Table 7, which list the results of several current state-of-the-art methods. As can be seen from Table 6, our proposed method gives the best pure supervised performance among both statistical and neural segments in all datasets by a large margin.

Moreover, our method are also competitive to the best semi-supervised methods, including those using rich pretraining information like mutual information(Sun and Xu, 2011), punctuation(Sun and Xu, 2011; Yang et al., 2017), automatically segmented text(Zhou et al., 2017; Yang et al., 2017), POS data(Sun and Xu, 2011; Yang et al., 2017) or word-context embedding(Zhou et al., 2017). As can be seen in Table 7, we outperform the best semi-supervised models on CTB6, AS and CityU datasets.

In summary, while competitive to the best semi-

supervised methods, our gap-based approach gives the best pure supervised performance on all corpora. To our knowledge, we are the first to report state-of-the-art results on both CTB6 and all SIGHAN 2005 bakeoff benchmarks. It verifies that while simple, the gap-based framework is very efficient for CWS.

## 5 Related Work

Xue (2003) was the first to propose to regard CWS task as character-tagging, using a maximum entropy model to give each character a label from {B, I, E, S}. Peng et al. (2004) followed this character-tagging scheme and proposed a conditional random field (CRF) to further improve the performance. Since then, this sequence-labeling CRF scheme was followed by most subsequent approaches in the literature.

Zheng et al. (2013) was the first to propose a neural network model and introduced character embedding for CWS, using a character window of size 5. This kind of 5-character window design still appears in recent state-of-the-art models. Pei et al. (2014) used a tensor neural network to further exploit the character combination feature and introduced bi-character embedding into neural CWS models. Chen et al. (2015a) proposed a Gated Recursive Neural Network to combine character features. Chen et al. (2015b) proposed a LSTM model in order to get rid of this 5-character window design. Xu and Sun (2016) proposed a Dependency-based Gated Recursive Neural Network to efficiently combine local and global features. Wang and Xu (2017) proposed a convolutional neural network to extract features. These character-based models exploited the recent springing up neural network architectures.

Andrew (2006) introduced the log conditional odds that a given token sequence constitutes a chunk according to a generative model. Zhang and Clark (2007) was the first to proposed a word-based approach to CWS, which provides a direct solution to the problem. Zhang and Clark (2011) proposed a beam-search model. Zhang et al. (2016) proposed a neural transition-based model with beam search that explicitly produce chunks in order. Cai and Zhao (2016) and Cai et al. (2017) proposed a model that score the candidate segmented outputs directly. These word-based models can fully exploit the word features such as word embedding.

Observing the similarity of sequence segmentation in natural language processing (NLP) literature and and semantic segmentation in computer vision literature, we may find that "simplest things are the best" Occam's razor is not applied to NLP sequence segmentation tasks. Zhang et al. (2016) proposed to use Fully Convolutional DenseNets to do semantic segmentation by directly do classification on each pixel and beat other complex framework such as fully connected CRFs (Krähenbühl and Koltun, 2011). This paper provides us the idea to directly classify the gaps.

The model architecture of Wang and Xu (2017) is similar to ours, as both of us use convolutional neural networks to extract character combination features. However, their models are rather shallow (up to 5 layers) and only use feed forward connections, while we introduce deep feature extraction blocks that contain residual connections or dense connections, inspired by He et al. (2016) and Huang et al. (2017). Moreover, their models are still in character-based framework, while we show that our gap-based framework can further exploit the representation power of deep neural networks.

Our approach remains much potential that can be further investigated and improved in the future. For example, our models may furthermore benefit from recently popular semi-supervised learning methods, such as word-context character embedding(Zhou et al., 2017), rich pretraining(Yang et al., 2017) and pretrained word embedding (Wang and Xu, 2017). They can all get more information from auto-segmented text. The use of LSTM-RNN and its variants in the gap-based framework is also interesting to be investigated.

## 6 Conclusion

In this paper, we propose a novel gap-based framework for Chinese word segmentation that directly predict whether to segment for each gap between two consecutive characters. Moreover, we introduce very deep convolutional networks (residual blocks and dense blocks) for feature extraction.

Experiments show that our proposed Gap-Based ConvNets are effective to solve Chinese word segmentation task. We outperform the previous best character-based and word-based methods by a large margin. To our knowledge, we are the first to report state-of-the-art results on both CTB6 and all SIGHAN 2005 bakeoff benchmarks.

# References

G. Andrew. 2006. A hybrid markov/semi-markov conditional random field for sequence segmentation. In *Proceedings of Empirical Methods in Natural Language Processing(EMNLP)*. https://www.microsoft.com/en-us/research/publication/a-hybrid-markovsemi-markov-conditional-random-field-for-sequence-segmentation/.

Deng Cai and Hai Zhao. 2016. Neural word segmentation learning for chinese. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Berlin, Germany, pages 409–420. http://www.aclweb.org/anthology/P16-1039.

Deng Cai, Hai Zhao, Zhisong Zhang, Yuan Xin, Yongjian Wu, and Feiyue Huang. 2017. Fast and accurate neural word segmentation for chinese. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, Vancouver, Canada, pages 608–615. http://aclweb.org/anthology/P17-2096.

Xinchi Chen, Xipeng Qiu, Chenxi Zhu, and Xuanjing Huang. 2015a. Gated recursive neural network for chinese word segmentation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, Beijing, China, pages 1744–1753. http://www.aclweb.org/anthology/P15-1168.

Xinchi Chen, Xipeng Qiu, Chenxi Zhu, Pengfei Liu, and Xuanjing Huang. 2015b. Long short-term memory neural networks for chinese word segmentation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, pages 1197–1206. http://aclweb.org/anthology/D15-1141.

Xinchi Chen, Zhan Shi, Xipeng Qiu, and Xuanjing Huang. 2017. Adversarial multi-criteria learning for chinese word segmentation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Vancouver, Canada, pages 1193–1203. http://aclweb.org/anthology/P17-1110.

Yann N Dauphin, Angela Fan, Michael Auli, and David Grangier. 2016. Language modeling with gated convolutional networks. *arXiv preprint arXiv:1612.08083* .

Thomas Emerson. 2005. The second international chinese word segmentation bakeoff. In *Proceedings of the fourth SIGHAN workshop on Chinese language Processing*. volume 133, pages 123–133.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning. *arXiv preprint arXiv:1705.03122* .

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. pages 249–256.

Chen Gong, Zhenghua Li, Min Zhang, and Xinzhou Jiang. 2017. Multi-grained chinese word segmentation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Copenhagen, Denmark, pages 703–714. https://www.aclweb.org/anthology/D17-1073.

Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. 2017. Mask r-cnn. In *The IEEE International Conference on Computer Vision (ICCV)*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. pages 770–778.

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

Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. 2017. Densely connected convolutional networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*. pages 448–456.

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

Philipp Krähenbühl and Vladlen Koltun. 2011. Efficient inference in fully connected crfs with gaussian edge potentials. In *Advances in neural information processing systems*. pages 109–117.

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

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

Wenzhe Pei, Tao Ge, and Baobao Chang. 2014. Max-margin tensor neural network for chinese word segmentation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*

*(Volume 1: Long Papers)*. Association for Computational Linguistics, Baltimore, Maryland, pages 293–303. http://www.aclweb.org/anthology/P14-1028.

Fuchun Peng, Fangfang Feng, and Andrew McCallum. 2004. Chinese segmentation and new word detection using conditional random fields. In *Proceedings of the 20th international conference on Computational Linguistics*. Association for Computational Linguistics, page 562.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, Curran Associates, Inc., pages 91–99. http://papers.nips.cc/paper/5638-faster-r-cnn-towards-real-time-object-detection-with-region-proposal-networks.pdf.

Weiwei Sun. 2010. Word-based and character-based word segmentation models: Comparison and combination. In *Coling 2010: Posters*. Coling 2010 Organizing Committee, Beijing, China, pages 1211–1219. http://www.aclweb.org/anthology/C10-2139.

Weiwei Sun and Jia Xu. 2011. Enhancing chinese word segmentation using unlabeled data. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Edinburgh, Scotland, UK., pages 970–979. http://www.aclweb.org/anthology/D11-1090.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pages 2818–2826.

Ben Taskar, Carlos Guestrin, and Daphne Koller. 2004. Max-margin markov networks. In *Advances in neural information processing systems*. pages 25–32.

Chunqi Wang and Bo Xu. 2017. Convolutional Neural Network with Word Embeddings for Chinese Word Segmentation. In *Proceedings of the 8th International Joint Conference on Natural Language Processing*.

Mengqiu Wang, Rob Voigt, and Christopher D. Manning. 2014. Two knives cut better than one: Chinese word segmentation with dual decomposition. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, Baltimore, Maryland, pages 193–198. http://www.aclweb.org/anthology/P14-2032.

Jingjing Xu and Xu Sun. 2016. Dependency-based gated recursive neural network for chinese word segmentation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*

*(Volume 2: Short Papers)*. Association for Computational Linguistics, Berlin, Germany, pages 567–572. http://anthology.aclweb.org/P16-2092.

Naiwen Xue, Fei Xia, Fu-Dong Chiou, and Marta Palmer. 2005. The penn chinese treebank: Phrase structure annotation of a large corpus. *Natural language engineering* 11(2):207–238.

Nianwen Xue. 2003. Chinese word segmentation as character tagging. In *International Journal of Computational Linguistics & Chinese Language Processing, Volume 8, Number 1, February 2003: Special Issue on Word Formation and Chinese Language Processing*. pages 29–48. http://www.aclweb.org/anthology/O03-4002.

Jie Yang, Yue Zhang, and Fei Dong. 2017. Neural word segmentation with rich pretraining. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Vancouver, Canada, pages 839–849. http://aclweb.org/anthology/P17-1078.

Sergey Zagoruyko and Nikos Komodakis. 2016. Wide residual networks. *arXiv preprint arXiv:1605.07146* .

Meishan Zhang, Yue Zhang, and Guohong Fu. 2016. Transition-based neural word segmentation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Berlin, Germany, pages 421–431. http://www.aclweb.org/anthology/P16-1040.

Yue Zhang and Stephen Clark. 2007. Chinese segmentation with a word-based perceptron algorithm. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*. Association for Computational Linguistics, Prague, Czech Republic, pages 840–847. http://www.aclweb.org/anthology/P07-1106.

Yue Zhang and Stephen Clark. 2011. Syntactic processing using the generalized perceptron and beam search. *Computational linguistics* 37(1):105–151.

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

Hao Zhou, Zhenting Yu, Yue Zhang, Shujian Huang, Xinyu Dai, and Jiajun Chen. 2017. Word-context character embeddings for chinese word segmentation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Copenhagen, Denmark, pages 771–777. https://www.aclweb.org/anthology/D17-1080.