Segment-Level Sequence Modeling using Gated Recursive Semi-Markov Conditional Random Fields

Jingwei Zhuo[1,2]*, **Yong Cao**[2], **Jun Zhu**[1]†, **Bo Zhang**[1], **Zaiqing Nie**[2]
[1]Dept. of Comp. Sci. & Tech., State Key Lab of Intell. Tech. & Sys., TNList Lab,
Tsinghua University, Beijing, 100084, China
[2]Microsoft Research, Beijing, 100084, China
{zjw15@mails, dcszj@mail, dcszb@mail}.tsinghua.edu.cn; {yongc, znie}@microsoft.com

Abstract

Most of the sequence tagging tasks in natural language processing require to recognize segments with certain syntactic role or semantic meaning in a sentence. They are usually tackled with Conditional Random Fields (CRFs), which do indirect word-level modeling over word-level features and thus cannot make full use of segment-level information. Semi-Markov Conditional Random Fields (Semi-CRFs) model segments directly but extracting segment-level features for Semi-CRFs is still a very challenging problem. This paper presents Gated Recursive Semi-CRFs (grSemi-CRFs), which model segments directly and automatically learn segment-level features through a gated recursive convolutional neural network. Our experiments on text chunking and named entity recognition (NER) demonstrate that grSemi-CRFs generally outperform other neural models.

1 Introduction

Most of the sequence tagging tasks in natural language processing (NLP) are segment-level tasks, such as text chunking and named entity recognition (NER), which require to recognize segments (i.e., a set of continuous words) with certain syntactic role or semantic meaning in a sentence. These tasks are usually tackled with Conditional Random Fields (CRFs) (Lafferty et al., 2001), which do word-level modeling as putting each word a tag, by using some predefined tagging schemes, e.g., the "IOB" scheme (Ramshaw and Marcus, 1995). Such tagging schemes are lossy transformations of original segment tags: They do indicate the boundary of adjacent segments but lose the length information of segments to some extent. Besides, CRFs can only employ word-level features, which are either hand-crafted or extracted with deep neural networks, such as window-based neural networks (Collobert et al., 2011) and bidirectional Long Short-Term Memory networks (BI-LSTMs) (Huang et al., 2015). Therefore, CRFs cannot make full use of segment-level information, such as inner properties of segments, which cannot be fully encoded in word-level features.

Semi-Markov Conditional Random Fields (Semi-CRFs) (Sarawagi and Cohen, 2004) are proposed to model segments directly and thus readily utilize segment-level features that encode useful segment information. Existing work has shown that Semi-CRFs outperform CRFs on segment-level tagging tasks such as sequence segmentation (Andrew, 2006), NER (Sarawagi and Cohen, 2004; Okanohara et al., 2006), web data extraction (Zhu et al., 2007) and opinion extraction (Yang and Cardie, 2012). However, Semi-CRFs need many more features compared to CRFs as they need to model segments with different lengths. As manually designing the features is tedious and often incomplete, *how to automatically extract good features* becomes a very important problem for Semi-CRFs. A naive solution that builds multiple feature extractors, each of which extracts features for segments with a specific length, is apparently time-consuming. Moreover, some of these separate extractors may underfit as the segments with specific length may be very rare in the training data. By far, Semi-CRFs are lacking of an automatic segment-level feature extractor.

In this paper, we fill the research void by

* This work was done when J.W.Z was on an internship with Microsoft Research.

† J.Z is the corresponding author.

proposing Gated Recursive Semi-Markov Conditional Random Fields (grSemi-CRFs), which can automatically learn features for segment-level sequence tagging tasks. Unlike previous approaches which usually use a neural-based feature extractor with a CRF layer, a grSemi-CRF consists of a gated recursive convolutional neural network (grConv) (Cho et al., 2014) with a Semi-CRF layer. The grConv is a variant of recursive neural networks. It builds a pyramid-like structure to extract segment-level features in a hierarchical way. This feature hierarchy well matches the intuition that long segments are combinations of their short sub-segments. This idea was first explored in Cho et al. (2014) to build an encoder in neural machine translation and then extended to solve other problems, such as sentence-level classification (Zhao et al., 2015) and Chinese word segmentation (Chen et al., 2015).

The advantages of grSemi-CRFs are two folds. First, thanks to the pyramid architecture of grConvs, grSemi-CRFs can extract all the segment-level features using one single feature extractor, and there is no underfitting problem as all parameters of the feature extractor are shared globally. Besides, unlike recurrent neural network (RNN) models, the training and inference of grSemi-CRFs are very fast as there is no time dependency and all the computations can be done in parallel. Second, thanks to the semi-Markov structure of Semi-CRFs, grSemi-CRFs can model segments in sentences directly without the need to introduce extra tagging schemes, which solves the problem that segment length information cannot be fully encoded in tags. Besides, grSemi-CRFs can also utilize segment-level features which can flexibly encode segment-level information such as inner properties of segments, compared to word-level features as used in CRFs. By combining grConvs with Semi-CRFs, we propose a new way to automatically extract segment-level features for Semi-CRFs.

Our major contributions can be summarized as:

(1) We propose grSemi-CRFs, which solve both the automatic feature extraction problem for Semi-CRFs and the indirect word-level modeling problem in CRFs. As a result, grSemi-CRFs can do segment-level modeling directly and make full use of segment-level features;

(2) We evaluate grSemi-CRFs on two segment-level sequence tagging tasks, text chunking and NER. Experimental results show the effectiveness of our model.

2 Preliminary

In sequence tagging tasks, given a word sequence, the goal is to assign each word (e.g., in POS Tagging) or each segment (e.g., in text chunking and NER) a tag. By leveraging a tagging scheme like "IOB", all the tasks can be regarded as word-level tagging. More formally, let $\mathcal{X}$ denote the set of words and $\mathcal{Y}$ denote the set of tags. A word sentence with length T can be denoted by $\mathbf{x} = (x_1, ..., x_T)$ and its corresponding tags can be denoted as $\mathbf{y} = (y_1, ..., y_T)$. A CRF (Lafferty et al., 2001) defines a conditional distribution

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp\left(\sum_{t=1}^{T} F(y_t, \mathbf{x}) + A(y_{t-1}, y_t)\right), \quad (1)$$

where $F(y_t, \mathbf{x})$ is the tag score (or potential) for tag y_t at position t, $A(y_{t-1}, y_t)$ is the transition score between y_{t-1} and y_t to measure the tag dependencies of adjacent words and $Z(\mathbf{x}) = \sum_{\mathbf{y}'} \exp\left(\sum_{t=1}^{T} F(y'_t, \mathbf{x}) + A(y_{t-1}, y'_t)\right)$ is the normalization factor. For the common log-linear models, $F(y_t, \mathbf{x})$ can be computed by

$$F(y_t, \mathbf{x}) = \mathbf{v}_{y_t}^{\mathrm{T}} \mathbf{f}(y_t, \mathbf{x}) + b_{y_t}, \quad (2)$$

where $\mathrm{V} = (\mathbf{v}_1, ..., \mathbf{v}_{|\mathcal{Y}|})^{\mathrm{T}} \in \mathbb{R}^{|\mathcal{Y}| \times D}$, $\mathbf{b}_{\mathrm{V}} = (b_1, ..., b_{|\mathcal{Y}|})^{\mathrm{T}} \in \mathbb{R}^{|\mathcal{Y}|}$, $f(y_t, \mathbf{x}) \in \mathbb{R}^D$ are the word-level features for y_t over the sentence $\mathbf{x}$ and D is the number of features. $f(y_t, \mathbf{x})$ can be manually designed or automatically extracted using neural networks, such as window-based neural networks (Collobert et al., 2011).

If we consider segment-level tagging directly[1], we get a segmentation of the previous tag sentence. With a little abuse of notation, we denote a segmentation by $\mathbf{s} = (s_1, ..., s_{|\mathbf{s}|})$ in which the jth segment $s_j = \langle h_j, d_j, y_j \rangle$ consists of a start position h_j, a length $d_j < L$ where L is a predefined upperbound and a tag y_j. Conceptually, s_j means a tag y_j is given to words $(x_{h_j}, ..., x_{h_j + d_j - 1})$. A Semi-CRF (Sarawagi and Cohen, 2004) defines a conditional distribution

$$p(\mathbf{s}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp\left(\sum_{j=1}^{|\mathbf{s}|} F(s_j, \mathbf{x}) + A(y_{j-1}, y_j)\right), \quad (3)$$

[1]Word-level tagging can be regarded as segment-level tagging over length-1 segments.

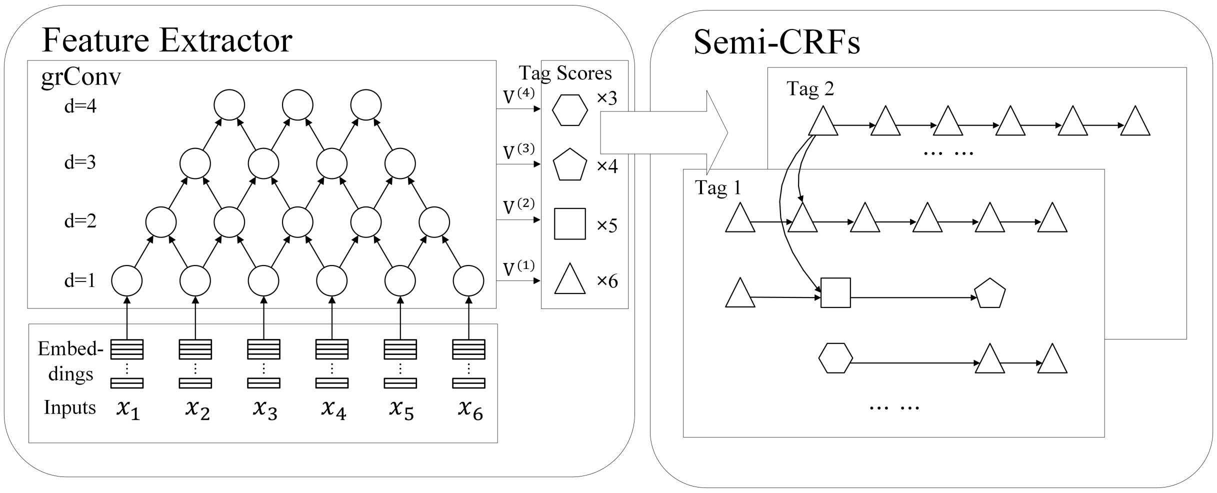

Figure 1: An overview of grSemi-CRFs. For simplicity, we set the segment length upperbound $L = 4$, and the sentence length $T = 6$. The left side is the feature extractor, in which each node denotes a vector of segment-level features (e.g., $\mathbf{z}_k^{(d)}$ for the kth node in the dth layer). Embeddings of word-level input features are used as length-1 segment-level features, and the length-d feature is extracted from two adjacent length-$(d-1)$ features. The right side is the Semi-CRF. Tag score vectors are computed as linear transformations of segment-level features and the number of them equals the number of nodes in the same layer. For clarity, we use triangle, square, pentagon and hexagon to denote the tag score vectors for length-$1, 2, 3, 4$ segments and directed links to denote the tag transformations of adjacent segments.

where $F(s_j, \mathbf{x})$ is the potential or tag score for segment s_j, $A(y_{j-1}, y_j)$ is the transition score to measure tag dependencies of adjacent segments and $Z(\mathbf{x}) = \sum_{\mathbf{s}'} \exp\left(\sum_{j=1}^{|\mathbf{s}'|} F(s'_j, \mathbf{x}) + A_{y'_{j-1}, y'_j}\right)$ is the normalization factor. For the common log-linear models, $F(s_j, \mathbf{x})$ can be computed by

$$F(s_j, \mathbf{x}) = \mathbf{v}_{y_j}^{\mathrm{T}} \mathbf{f}(s_j, \mathbf{x}) + b_{y_j}, \tag{4}$$

where $\mathrm{V} = (\mathbf{v}_1, ..., \mathbf{v}_{|\mathcal{Y}|})^{\mathrm{T}} \in \mathbb{R}^{|\mathcal{Y}| \times D}$, $\mathbf{b}_{\mathrm{V}} = (b_1, ..., b_{|\mathcal{Y}|})^{\mathrm{T}} \in \mathbb{R}^{|\mathcal{Y}|}$ and $f(s_j, \mathbf{x}) \in \mathbb{R}^D$ are the segment-level features for s_j over the sentence $\mathbf{x}$.

As Eq. (1) and Eq. (3) show, CRFs can be regarded as a special case of Semi-CRFs when $L = 1$. CRFs need features for only length-1 segments (i.e., words), while Semi-CRFs need features for length-ℓ segments ($1 \leq \ell \leq L$). Therefore, to model the same sentence, Semi-CRFs generally need many more features than CRFs, especially when L is large. Besides, unlike word-level features used in CRFs, the sources of segment-level features are often quite limited. In existing work, the sources of $f(s_j, \mathbf{x})$ can be roughly divided into two parts: (1) Concatenations of word-level features (Sarawagi and Cohen, 2004; Okanohara et al., 2006); and (2) Hand-crafted segment-level features, including task-insensitive features, like the length of segments, and task-specific features, like the verb phrase patterns in opinion extraction (Yang and Cardie, 2012). As manually designing features is time-consuming and often hard to capture rich statistics underlying the data, how to automatically extract features for Semi-CRFs remains a challenge.

3 Gated Recursive Semi-Markov CRFs

In this section, we present Gated Recursive Semi-CRFs (grSemi-CRFs), which inherit the advantages of Semi-CRFs in segment-level modeling, and also solve the feature extraction problem of Semi-CRFs by introducing a gated recursive convolutional neural network (grConv) as the feature extractor. Instead of building multiple feature extractors at different scales of segment lengths, grSemi-CRFs can extract features with any length by using a single grConv, and learn the parameters effectively via sharing statistics.

3.1 Architecture

The architecture of grSemi-CRFs is illustrated in Figure 1. A grSemi-CRF can be divided into two parts, a feature extractor (i.e., grConv) and a Semi-CRF. Below, we explain each part in turn.

As is shown, the feature extractor is a pyramid-like directed acyclic graph (DAG), in which nodes

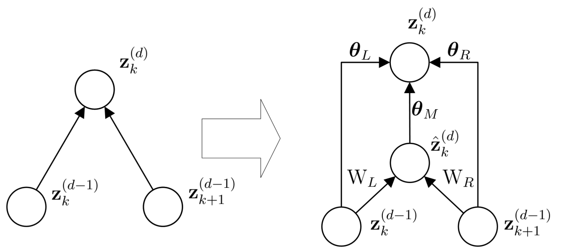

Figure 2: The the building block of the feature extractor (i.e., grConv), in which parameters are shared among the pyramid structure. We omit the dependency of $\boldsymbol{\theta}_L, \boldsymbol{\theta}_R, \boldsymbol{\theta}_M$ on $\mathrm{G}_L, \mathrm{G}_R$.

are stacked layer by layer and information is propagated from adjacent nodes in the same layer to their co-descendants in the higher layer through directed links. Recall that L denotes the upper-bound of segment length (i.e., the height of the grConv), we regard the bottom level as the 1st level and the top level as the Lth level. Then, for a length-T sentence, the dth level will have $T - d + 1$ nodes, which correspond to features for $T - d + 1$ length-d segments. The kth node in the dth layer corresponds to the segment-level latent features $\mathbf{z}_k^{(d)} \in \mathbb{R}^D$, which denote the meaning of the segment, e.g., the syntactic role (i.e., for text chunking) or semantic meaning (i.e., for NER).

Like CRFs, grSemi-CRFs allow word-level categorical inputs (i.e., $\mathbf{x}_k$) which are transformed into continuous vectors (i.e., embeddings) according to look-up tables and then used as length-1 segment-level features (i.e., $\mathbf{z}_k^{(1)}$). To be clear, we call these inputs as input features and those extracted segment-level features (i.e., $\mathbf{z}_k^{(d)}$) as segment-level latent features. Besides, grSemi-CRFs also allow segment-level input features (e.g., gazetteers) directly as shown in Eq. (12). We will discuss more details in section 4.3.

The building block of the feature extractor is shown in Figure 2, where an intermediate node $\hat{\mathbf{z}}_{(d)} \in \mathbb{R}^D$ is introduced to represent the interactions of two length-$(d-1)$ segments. To capture such complex interactions, $\hat{\mathbf{z}}_k^{(d)}$ is computed through a non-linear transformation, i.e.,

$$\hat{\mathbf{z}}_k^{(d)} = g(\boldsymbol{\alpha}_k^{(d)}) = g(\mathrm{W}_L \mathbf{z}_k^{(d-1)} + \mathrm{W}_R \mathbf{z}_{k+1}^{(d-1)} + \mathbf{b}_\mathrm{W}), \quad (5)$$

where $\mathrm{W}_L, \mathrm{W}_R \in \mathbb{R}^{D\times D}$ and $\mathbf{b}_\mathrm{W} \in \mathbb{R}^D$ are shared globally, and $g(\cdot)$ is a non-linear activation function[2].

[2]We use a modified version of the sigmoid function, i.e., $g(x) = 4\left(\frac{1}{1+e^{-x}} - \frac{1}{2}\right)$.

Then, the length-d segment-level latent features $\mathbf{z}_k^{(d)}$ can be computed as

$$\mathbf{z}_k^{(d)} = \theta_L \mathbf{z}_k^{(d-1)} + \theta_R \mathbf{z}_{k+1}^{(d-1)} + \theta_M \hat{\mathbf{z}}_k^{(d)}, \quad (6)$$

where θ_L, θ_M and $\theta_R \in \mathbb{R}$ are the gating coefficients which satisfy the condition $\theta_L, \theta_R, \theta_M \geq 0$ and $\theta_L + \theta_R + \theta_M = 1$. Here, we make a little modification of grConvs by making the gating coefficients as vectors instead of scalars, i.e.,

$$\mathbf{z}_k^{(d)} = \boldsymbol{\theta}_L \circ \mathbf{z}_k^{(d-1)} + \boldsymbol{\theta}_R \circ \mathbf{z}_{k+1}^{(d-1)} + \boldsymbol{\theta}_M \circ \hat{\mathbf{z}}_k^{(d)}, \quad (7)$$

where $\circ$ denotes the element-wise product and $\boldsymbol{\theta}_L$, $\boldsymbol{\theta}_R$ and $\boldsymbol{\theta}_M \in \mathbb{R}^D$ are vectorial gating coefficients[3] which satisfy the condition that $\theta_{L,i}, \theta_{R,i}, \theta_{M,i} \geq 0$ and $\theta_{L,i} + \theta_{R,i} + \theta_{M,i} = 1$ for $1 \leq i \leq D$. There are two reasons for this modification: (1) Theoretically, the element-wise combination makes a detailed modeling as each feature in $\mathbf{z}_k^{(d)}$ may have its own combining; and (2) Experimentally, this setting makes our grSemi-CRF[4] more flexible, which increases its generalizability and leads to better performance in experiments as shown in Table 4.

We can regard Eq. (7) as a soft gate function to control the propagation flows. Besides, all the parameters (i.e., $\mathrm{W}_L, \mathrm{W}_R, \mathbf{b}_\mathrm{W}, \mathrm{G}_L, \mathrm{G}_R, \mathbf{b}_\mathrm{G}$) are shared globally and recursively applied to the input sentence in a bottom-up manner. All of these account for the name *gated recursive convolutional neural networks* (grConvs).

Eq. (5) and Eq. (7) build the information propagation criteria in a grConv. The basic assumption behind Eq. (5) and Eq. (7) is that the meaning of one segment can be represented as a linear combination of three parts: (1) the meaning of its prefix segment, (2) the meaning of its suffix segment and (3) the joint meaning of both (i.e., the complex interaction). This process matches our intuition about the hierarchical structure in the composition of a sentence. For example, the meaning of *the United States* depends on the suffix segment *United States*, whose meaning is not only from its prefix *United* or suffix *States*, but the interaction of both.

The vectorial gating coefficients $\boldsymbol{\theta}_L$, $\boldsymbol{\theta}_R$ and $\boldsymbol{\theta}_M$

[3]We omit the dependency of $\boldsymbol{\theta}_L, \boldsymbol{\theta}_R$ and $\boldsymbol{\theta}_M$ on d and k for notation simplicity.

[4]Unless otherwise stated, we regard "grSemi-CRF" as grSemi-CRF with vectorial gating coefficients in default.

are computed adaptively, i.e.,

$$\begin{pmatrix} \boldsymbol{\theta}_L \\ \boldsymbol{\theta}_R \\ \boldsymbol{\theta}_M \end{pmatrix} = \begin{pmatrix} 1/\mathbf{Z} \\ 1/\mathbf{Z} \\ 1/\mathbf{Z} \end{pmatrix} \circ \exp\left(\mathrm{G}_L \mathbf{z}_k^{(d-1)} + \mathrm{G}_R \mathbf{z}_{k+1}^{(d-1)} + \mathbf{b}_\mathrm{G} \right), \tag{8}$$

where $\mathrm{G}_L, \mathrm{G}_R \in \mathbb{R}^{3D \times D}$ and $\mathbf{b}_\mathrm{G} \in \mathbb{R}^{3D}$ are shared globally. $\mathbf{Z} \in \mathbb{R}^d$ is normalization coefficients and the ith element of $\mathbf{Z}$ is computed via

$$Z_i = \sum_{j=1}^{3} \left[\exp\left(\mathrm{G}_L \mathbf{z}_k^{(d-1)} + \mathrm{G}_R \mathbf{z}_{k+1}^{(d-1)} + \mathbf{b}_\mathrm{G} \right) \right]_{D \times (j-1) + i}. \tag{9}$$

After the forward propagation of the feature extractor is over, the tag scores (i.e., the potential functions for Semi-CRFs) are computed through a linear transformation. For segment $s_j = \langle h_j, d_j, y_j \rangle$, its latent feature is $f(s_j, \mathbf{x}) = \mathbf{z}_{h_j}^{(d_j)}$ and corresponding potential/tag score is

$$F(\mathbf{s}_j; \mathbf{x}) = f(\langle h_j, d_j, y_j \rangle; \mathbf{x}) = \left[\mathrm{V}_0^{(d_j)} \mathbf{z}_{h_j}^{(d_j)} + \mathbf{b}_\mathrm{V}^{(d_j)} \right]_{y_j}, \tag{10}$$

where $\mathrm{V}_0^{(d_j)} \in \mathbb{R}^{|\mathcal{Y}| \times D}$ and $\mathbf{b}^{(d_j)} \in \mathbb{R}^{|\mathcal{Y}|}$ are parameters for length-d_j segments. To encode contextual information, we can assume that the tag of a segment depends not only on itself but also its neighbouring segments with the same length, i.e.,

$$F(\mathbf{s}_j; \mathbf{x}) = \left[\sum_{i=-H}^{H} \mathrm{V}_i^{(d_j)} \mathbf{z}_{h_j+i}^{(d_j)} + \mathbf{b}_\mathrm{V}^{(d_j)} \right]_{y_j}, \tag{11}$$

where $\mathrm{V}_{-H}^{(d_j)}, ..., \mathrm{V}_0^{(d_j)}, ..., \mathrm{V}_H^{(d_j)} \in \mathbb{R}^{|\mathcal{Y}| \times D}$ and H is the window width for neighbouring segments. Apart from the automatically extracted segment-level latent features $\mathbf{z}_k^{(d)}$, grSemi-CRFs also allow segment-level input features (e.g., gazetteers), i.e.,

$$F(\mathbf{s}_j; \mathbf{x}) = \left[\sum_{i=-H}^{H} \mathrm{V}_i^{(d_j)} \mathbf{z}_{h_j+i}^{(d_j)} + \mathbf{b}_\mathrm{V}^{(d_j)} + \mathrm{U}^{(d_j)} \mathbf{c}_{h_j}^{(d_j)} \right]_{y_j}, \tag{12}$$

where $\mathrm{U}^{(d_j)} \in \mathbb{R}^{|\mathcal{Y}| \times D'}$ and $\mathbf{c}_{h_j}^{(d_j)} \in \mathbb{R}^{D'}$ is a vector of segment-level input features.

Then, we can use Eq. (3) for inference by using a Semi-CRF version of Viterbi algorithms (Sarawagi and Cohen, 2004).

3.2 Learning of Parameters

To learn grSemi-CRFs, we maximize the log likelihood $\mathcal{L} = \log p(\mathbf{s}|\mathbf{x})$ over all the parameters. Here, for notation simplity, we consider the simpliest case, i.e., using Eq. (10) to compute tag scores. More details can be found in the supplementary note.

Gradients of Semi-CRF-based parameters (i.e., A and V_0) and tag scores $F(s_j, \mathbf{x})$ can be computed based on the marginal probability of neighbouring segments via a Semi-CRF version of forward-backward algorithms (Sarawagi and Cohen, 2004). As for the grConv-based parameters, we can compute their gradients by back propagation. For example, gradients for W_L and G_L are[5]

$$\frac{\partial \mathcal{L}}{\partial \mathrm{W}_L} = \sum_{d=1}^{L} \sum_{k=1}^{T-d+1} \frac{\partial \mathcal{L}}{\partial \mathbf{z}_k^{(d)}} \frac{\partial \mathbf{z}_k^{(d)}}{\partial \mathrm{W}_L}, \frac{\partial \mathcal{L}}{\partial \mathrm{G}_L} = \sum_{d=1}^{L} \sum_{k=1}^{T-d+1} \frac{\partial \mathcal{L}}{\partial \mathbf{z}_k^{(d)}} \frac{\partial \mathbf{z}_k^{(d)}}{\partial \mathrm{G}_L}, \tag{13}$$

where $\frac{\partial \mathbf{z}_k^{(d)}}{\partial \mathrm{W}_L}$ and $\frac{\partial \mathbf{z}_k^{(d)}}{\partial \mathrm{G}_L}$ can be derived from Eq. (5), Eq. (7) and Eq. (8). For $\frac{\partial \mathcal{L}}{\partial \mathbf{z}_k^{(d)}}$, thanks to the recursive structure, it can be computed as

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \mathbf{z}_k^{(d)}} = & \frac{\partial \mathbf{z}_k^{(d+1)}}{\partial \mathbf{z}_k^{(d)}} \frac{\partial \mathcal{L}}{\partial \mathbf{z}_k^{(d+1)}} + \frac{\partial \mathbf{z}_{k-1}^{(d+1)}}{\partial \mathbf{z}_k^{(d)}} \frac{\partial \mathcal{L}}{\partial \mathbf{z}_{k-1}^{(d+1)}} \\ & + \mathrm{V}_0^{(d)\mathrm{T}} \frac{\partial \mathcal{L}}{\partial F(\mathbf{s}_k^{(d)}, \mathbf{x})}, \end{aligned} \tag{14}$$

where $\mathbf{s}_k^{(d)} = \langle k, d, \mathcal{Y} \rangle$ is a length-$|\mathcal{Y}|$ vector which denotes segments with all possible tags for $\mathbf{z}_k^{(d)}$, $\frac{\partial \mathcal{L}}{\partial F(\mathbf{s}_k^{(d)}, \mathbf{x})}$ is the gradient for $F(\mathbf{s}_k^{(d)}, \mathbf{x})$ and

$$\frac{\partial \mathbf{z}_k^{(d+1)}}{\partial \mathbf{z}_k^{(d)}} = \mathrm{diag}(\boldsymbol{\theta}_L) + \mathrm{diag}(\boldsymbol{\theta}_M \circ g'(\boldsymbol{\alpha}_k^{(d+1)}))\mathrm{W}_L, \tag{15}$$

where $\mathrm{diag}(\boldsymbol{\theta}_L)$ denotes the diagonal matrix spanned by vector $\boldsymbol{\theta}_L$, and $\frac{\partial \mathbf{z}_{k-1}^{(d+1)}}{\partial \mathbf{z}_k^{(d)}}$ has a similar form. As Eq. (14) shows, for each node in the feature extractor of grSemi-CRFs, its gradient consists of two parts: (1) the gradients back propagated from high layer nodes (i.e., longer segments); and (2) the supervising signals from Semi-CRFs. In other words, the supervision in the objective function is added to each node in grSemi-CRFs. This is a nice property compared to other neural-based feature extractors used in CRFs, in which only the nodes of several layers on the top receive supervision. Besides, the term $\mathrm{diag}(\boldsymbol{\theta}_L)$ in Eq. (15) prevents $\frac{\partial \mathbf{z}_k^{(d+1)}}{\partial \mathbf{z}_k^{(d)}}$ from being too small when $g'(\boldsymbol{\alpha}_k^{(d)})$ and W_L are small, which acts as the linear unit recurrent connection in the memory block of LSTM (Hochreiter and Schmidhuber, 1997; Zhao et al., 2015). All of these help in avoiding gradient vanishing problems in training grSemi-CRFs.

[5]Gradients for $\mathrm{W}_R, \mathbf{b}_\mathrm{W}, \mathrm{G}_R, \mathbf{b}_\mathrm{G}$ can be computed in similar ways.

4 Experiments

We evaluate grSemi-CRFs on two segment-level sequence tagging NLP tasks: text chunking and named entity recognition (NER).

4.1 Datasets

For text chunking, we use the CONLL 2000 text chunking shared dataset[6] (Tjong Kim Sang and Buchholz, 2000), in which the objective is to divide the whole sentence into different segments according to their syntactic roles, such as noun *phrases* ("NP"), *verb phrases* ("VP") and *adjective phrases* ("ADJP"). We call it a "segment-rich" tasks as the number of phrases are much higher than that of non-phrases which is tagged with *others* ("O"). We evaluate performance over all the chunks instead of only noun pharse (NP) chunks.

For NER, we use the CONLL 2003 named entity recognition shared dataset[7] (Tjong Kim Sang and De Meulder, 2003), in which segments are tagged with one of four entity types: *person* ("PER"), *location* ("LOC"), *organization* ("ORG") and *miscellaneous*("MISC"), or *others* ("O") which is used to denote non-entities. We call it a "segment-sparse" task as entities are rare while non-entities are common.

4.2 Input Features

For each word, we use multiple input features, including the word itself, its length-3 prefix and length-4 suffix, its capitalization pattern, its POS tag, the length-4,8,12,20 prefixs of its Brown clusters (Brown et al., 1992) and gazetteers[8]. All of them are used as word-level input features except gazetteers, which are used as segment-level features directly. All the embeddings for word-level inputs are randomly initialized except word embeddings, which can be initialized randomly or by pretraining over unlabeled data, which is external information compared to the dataset. Besides word embeddings, Brown clusters and gazetteers are also based on external information, as summarized below:

- **Word embeddings**. We use Senna embeddings[9] (Collobert et al., 2011), which are 50-dimensional and have been commonly used in sequence tagging tasks (Collobert et al., 2011; Turian et al., 2010; Huang et al., 2015);
- **Brown clusters**. We train two types of Brown clusters using the implementation from Liang (2005): (1) We follow the setups of Ratinov and Roth (2009), Turian et al. (2010) and Collobert et al. (2011) to generate 1000 Brown clusters on Reuters RCV1 dataset (Lewis et al., 2004); (2) We generate 1000 Brown clusters on New York Times (NYT) corpus (Sandhaus, 2008);
- **Gazetteers**. We build our gazetteers based on the gazetteers used in Senna (Collobert et al., 2011) and Wikipedia entries, mainly the locations and organizations. We also denoise our gazetteers by removing overlapped entities and using BBN Pronoun Coreference and Entity Type Corpus (Weischedel and Brunstein, 2005) as filters[10].

4.3 Implementation Details

To learn grSemi-CRFs, we employ Adagrad (Duchi et al., 2011), an adaptive stochastic gradient descent method which has been proved successful in similar tasks (Chen et al., 2015; Zhao et al., 2015). To avoid overfitting, we use the dropout strategy (Srivastava et al., 2014) and apply it on the first layer (i.e., $\mathbf{z}_k^{(0)}$). We also use the strategy of ensemble classifiers, which is proved an effective way to improve generalization performance (Collobert et al., 2011). All results are obtained by decoding over an average Semi-CRF after 10 training runs with randomly initialized parameters.

For the CONLL 2003 dataset, we use the F_1 scores on the development set to help choose the best-performed model in each run. For the CONLL 2000 dataset, as there is no development set provided, we use cross validation as Turian et al. (2010) to choose hyperparameters. After that, we retrain model according to the hyperparameters and choose the final model in each run.

Our hyperparameter settings for these two tasks are shown in Table 1. The segment length is set according to the maximum segment length in training set. We set the minibatch size to 10, which means that we process 10 sentences in a batch. The window width defines the parameter H in Eq. (12) when producing tag score vectors.

[6]Available at: http://www.cnts.ua.ac.be/conll2000/chunking/

[7]Available at: http://www.cnts.ua.ac.be/conll2003/ner/

[8]Among them, POS tags are provided in the dataset.

[9]Available at http://ronan.collobert.com/senna/

[10]We apply gazetteers on BBN corpus, collect lists of false positive entities and clean our gazetteers according to these lists.

Hyperparameters	CONLL 2000	CONLL 2003
Segment length	15	10
Dropout	0.3	0.3
Learning rate	0.3	0.3
Epochs	15	20
Minibatches	10	10
Window width	2	2

Table 1: Hyperparameter settings for our model.

4.4 Results and Analysis

Table 2 shows the results of our grSemi-CRFs and other models[11]. We divide other models into two categories, i.e., neural models and non-neural models, according to whether neural networks are used as automatic feature extractors. For neural models, Senna (Collobert et al., 2011) consists of a window-based neural network for feature extraction and a CRF for word-level modeling while BI-LSTM-CRF (Huang et al., 2015) uses a bidirectional Long Short-Term Memory network for feature extraction and a CRF for word-level modeling.

For non-neural models, JESS-CM (Suzuki and Isozaki, 2008) is a semi-supervised model which combines Hidden Markov Models (HMMs) with CRFs and uses 1 billion unlabelled words in training. Lin and Wu (2009) cluster 20 million phrases over corpus with around 700 billion tokens, and use the resulting clusters as features in CRFs. Passos et al. (2014) propose a novel word embedding method which incorporates gazetteers as supervising signals in pretraining and builds a log-linear CRF over them. Ratinov and Roth (2009) use CRFs based on many non-local features and 30 gazetteers extracted from Wikipedia and other websites with more than 1.5 million entities.

As Table 2 shows, grSemi-CRFs outperform other neural models, in both text chunking and named entity recognition (NER) tasks. BI-LSTM-CRFs use many more input features than ours, which accounts for the phenomenon that the performance of our grSemi-CRFs is rather mediocre (i.e., 93.92% versus 94.13% and 84.66% versus 84.26%) without external information. However, once using Senna embeddings, our grSemi-CRFs perform much better than BI-LSTM-CRFs.

For non-neural models, one similarity of them is that they use a lot of hand-crafted features, and many of them are even task-specific. Unlike them, grSemi-CRFs use much fewer input features and most of them are task-insensitive[13]. However, grSemi-CRFs achieve almost the same performance, sometimes even better. For text chunking, grSemi-CRF outperforms all reported supervised models, except JESS-CM (Suzuki and Isozaki, 2008), a semi-supervised model using giga-word scale unlabeled data in training[14]. However, the performance of our grSemi-CRF (95.01%) is very close to that of JESS-CM (95.15%). For NER, the performance of grSemi-CRFs are also very closed to state-of-the-art results (90.87% versus 90.90%).

Input Features	CONLL 2000	CONLL 2003
None	93.92	84.66
Brown(NYT)	94.18	86.57
Brown(RCV1)	94.05	**88.22**
Emb	**94.73**	88.12
Gaz	–	87.94
Emb + Brown(NYT)	**95.01**	88.86
Emb + Brown(RCV1)	94.87	89.44
Emb + Gaz	–	**89.88**
Brown(NYT) + Gaz	–	88.69
Brown(RCV1) + Gaz	–	89.82
All(NYT)	–	90.00
All(RCV1)	–	**90.87**

Table 3: Results of grSemi-CRF with external information, measured in F_1 score. None = no external information, Emb = Senna embeddings, Brown = Brown clusters, Gaz = gazetteers and All = Emb + Brown + Gaz. NYT and RCV1 in the parenthesis denote the corpus used to generate Brown clusters. "–" means no results. Notice that gazetteers are only applied to NER.

4.4.1 Impact of External Information

As Table 3 shows, external information improve the performance of grSemi-CRFs for both tasks. Compared to text chunking, we can find out that external information plays an extremely important role in NER, which coincides with the general idea that NER is a knowledge-intensive task (Ratinov and Roth, 2009). Another interesting thing is that, Brown clusters generated from NYT corpus performs better on the CONLL 2000 task while those generated from Reuters RCV1 dataset performs better on the CONLL 2003 task. The reason is

[11]Because of the space limit, we only compare our model with other models which follow similar settings and achieve high performance.

[13]E.g.: for NER, JESS-CM uses 79 different features; Lin and Wu (2009) use 48 baseline and phrase cluster features; while we only use 11. Besides, grSemi-CRFs use almost the same features for chunking and NER (except gazetteers).

[14]Being semi-supervised, JESS-CM can learn from interactions between labelled and unlabelled data during training but the training is slow compared to supervised models.

	Models	CONLL 2000	CONLL 2003
Ours	grSemi-CRF (Random embeddings)	93.92	84.66
	grSemi-CRF (Senna embeddings)	**95.01**	**89.44** (**90.87**)
Neural Models	Senna (Random embeddings)	90.33	81.47
	Senna (Senna embeddings)	94.32	88.67 (89.59)
	BI-LSTM-CRF (Random)	94.13	84.26
	BI-LSTM-CRF (Senna embeddings)	94.46	88.83 (90.10)
Non-Neural Models	JESS-CM (Suzuki and Isozaki, 2008), 15M	94.67	89.36
	JESS-CM (Suzuki and Isozaki, 2008), 1B	**95.15**	89.92
	Ratinov and Roth (2009)[12]	–	90.57
	Lin and Wu (2009)	–	**90.90**
	Passos et al. (2014)	–	**90.90**

Table 2: Experimental results over the CONLL-2000 and CONLL-2003 shared datasets, measured in F_1 score. Numbers in parentheses are the F_1 score when using gazetteers. JESS-CM (Suzuki and Isozaki, 2008) is a semi-supervised model, in which 15M or 1B denotes the number of unlabeled words it uses for training.

Gating Coefficients	CONLL 2000	CONLL 2003
Scalars	94.47	89.27(90.54)
Vectors	**95.01**	**89.44**(**90.87**)

Table 4: F_1 scores of grSemi-CRF with scalar or vectorial gating coefficients. Numbers in parentheses are the F_1 score when using gazetteers.

that the CONLL 2000 dataset is the subset of Wall Street Journal (WSJ) part of the Penn Treebank II Corpus (Marcus et al., 1993) while the CONLL 2003 dataset is a subset of Reuters RCV1 dataset. Maybe the writing styles between NYT and WSJ are more similar than those between RCV1 and WSJ.

4.4.2 Impact of Vectorial Gating Coefficients

As Table 4 shows, a grSemi-CRF using vectorial gating coefficients (i.e., Eq. (7)) performs better than that using scalar gating coefficients (i.e., Eq. (6)), which provides evidences for the theoretical intuition that vectorial gating coefficients can make a detailed modeling of the combinations of segment-level latent features and thus performs better than scalar gating coefficients.

4.4.3 Visualization of Learnt Segment-Level Features

To demonstrate the quality of learnt segment-level features, we use an indirect way as widely adopted in previous work, e.g., Collobert et al. (2011). More specifically, we show 10 nearest neighbours for some selected queried segments according to Euclidean metric of corresponding features[15]. To fully demonstrate the power of grSemi-CRF in learning segment-level features, we use the Emb+Brown(RCV1) model in Table 3, which uses no gazetteers. We train the model on the CONLL 2003 training set and find nearest neighbours in the CONLL 2003 test set. We make no restrictions on segments, i.e., all possible segments with different lengths in the CONLL 2003 test set are candidates.

As Table 5 shown, most of the nearest segments are meaningful and semantically related. For example, the nearest segments for "Filippo Inzaghi" are not only tagged with *person*, but also names of famous football players as "Filippo Inzaghi".

There also exist some imperfect results. E.g., for "Central African Republic", nearest segments, which contain the same queried segment, are semantically related but not syntactically similar. The major reason may be that the CONLL 2003 dataset is a small corpus (if compared to the vast unlabelled data used to train Senna embeddings), which restricts the range for candidate segments and the quality of learnt segment-level features. Another reason is that labels in the CONLL 2003 dataset mainly encodes semantic information (e.g., named entities) instead of syntactic information (e.g., chunks).

Besides, as we make no restriction on the formulation of candidate segments, sometimes only a part of the whole phrase will be retrieved, e.g., "FC Hansa", which is the prefix of "FC Hansa Rostock". Exploring better way of utilizing unla-

[15]Using the cosine similarity generates similar results. However, as Collobert et al. (2011) use Euclidean metric, we follow their settings.

Queried Segments	Filippo Inzaghi	AC Milan	Central African Republic	Asian Cup
Nearest Neighbour Results	Pierluigi Casiraghi	FC Hansa	From Central African Republic	Scottish Cup
	Fabrizio Ravanelli	SC Freiburg	Southeast Asian Nations	European Cup
	Bogdan Stelea	FC Cologne	In Central African Republic	African Cup
	Francesco Totti	Aston Villa	The Central African Republic	World Cup
	Predrag Mijatovic	Red Cross	South African Breweries	UEFA Cup
	Fausto Pizzi	Yasuto Honda	Of Southeast Asian Nations	Europoean Cup
	Pierre Laigle	NAC Breda	New South Wales	Asian Games
	Pavel Nedved	La Plagne	Central African Republic .	Europa Cup
	Anghel Iordanescu	Sporting Gijon	Papua New Guinea	National League
	Zeljko Petrovic	NEC Nijmegen	Central Africa	F.A. Cup

Table 5: Visualization of segment-level features learnt on the CONLL 2003 dataset. For each column the queried segment is followed by its 10 nearest neighbors (measured by the cosine similarity of their feature vectors). Corresponding tags for these four queried segments are (from left to right): *person*, *organization*, *location* and *miscellaneous*.

belled data to improve learning segment-level features is part of the future work.

5 Discussions and Related Work

Cho et al. (2014) first propose grConvs to learn fix-length representations of the whole source sentence in neural machine translation. Zhao et al. (2015) use grConvs to learn hierarchical representations (i.e., multiple fix-length representations) of the whole sentence for sentence-level classification problem. Both of them focus on sentence-level classification problems while grSemi-CRFs are solving segment-level classification (sequence tagging) problems, which is fine-grained. Chen et al. (2015) propose Gated Recursive Neural Networks (GRNNs), a variant of grConvs, to solve Chinese word segmentation problem. GRNNs still do word-level modeling by using CRFs while grSemi-CRFs do segment-level modeling directly by using semi-CRFs and makes full use of the recursive structure of grConvs.

We believe that, the recursive neural network (e.g., grConv) is a natural feature extractor for Semi-CRFs, as it extracts features for every possible segments by one propagation over one trained model, which is fast-computing and efficient. In this sense, grSemi-CRFs provide a promising direction to explore.

6 Conclusions

In this paper, we propose Gated Recursive Semi-Markov Conditional Random Fields (grSemi-CRFs) for segment-level sequence tagging tasks. Unlike word-level models such as CRFs, grSemi-CRFs model segments directly without the need of using extra tagging schemes and also readily utilize segment-level features, both hand-crafted and automatically extracted by a grConv. Experimental evaluations demonstrate the effectiveness of grSemi-CRFs on both text chunking and NER tasks.

In future work, we are interested in exploring better ways of utilizing vast unlabelled data to improve grSemi-CRFs, e.g., to learn phrase embeddings from unlabelled data or designing a semi-supervised version of grSemi-CRFs.

Acknowledgments

J.W.Z, J.Z and B.Z are supported by the National Basic Research Program (973 Program) of China (No. 2013CB329403), National NSF of China (Nos. 61322308, 61332007), the Youngth Top-notch Talent Support Program, Tsinghua TNList Lab Big Data Initiative, and Tsinghua Initiative Scientific Research Program (No. 20141080934).

References

Galen Andrew. 2006. A hybrid markov/semi-markov conditional random field for sequence segmentation. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 465–472.

Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. 1992. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479.

Xinchi Chen, Xipeng Qiu, Chenxi Zhu, and Xuanjing Huang. 2015. Gated recursive neural network for chinese word segmentation. In *Proceedings of Annual Meeting of the Association for Computational Linguistics*.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder–decoder approaches. *Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, page 103.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, 12:2121–2159.

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

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289.

David D Lewis, Yiming Yang, Tony G Rose, and Fan Li. 2004. Rcv1: A new benchmark collection for text categorization research. *The Journal of Machine Learning Research*, 5:361–397.

Percy Liang. 2005. *Semi-Supervised Learning for Natural Language*. Ph.D. thesis, Massachusetts Institute of Technology.

Dekang Lin and Xiaoyun Wu. 2009. Phrase clustering for discriminative learning. In *Proceedings of Annual Meeting of the Association for Computational Linguistics*, pages 1030–1038.

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.

Daisuke Okanohara, Yusuke Miyao, Yoshimasa Tsuruoka, and Jun'ichi Tsujii. 2006. Improving the scalability of semi-markov conditional random fields for named entity recognition. In *Proceedings of Annual Meeting of the Association for Computational Linguistics*, pages 465–472.

Alexandre Passos, Vineet Kumar, and Andrew McCallum. 2014. Lexicon infused phrase embeddings for named entity resolution. *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, pages 78–86.

Lance A. Ramshaw and Mitchell P. Marcus. 1995. Text chunking using transformation-based learning. In *Proceedings of the Third Workshop on Very Large Corpora*.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pages 147–155.

Evan Sandhaus. 2008. The new york times annotated corpus. *Linguistic Data Consortium, Philadelphia*, 6.

Sunita Sarawagi and William W Cohen. 2004. Semi-markov conditional random fields for information extraction. In *Advances in Neural Information Processing Systems*, pages 1185–1192.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

Jun Suzuki and Hideki Isozaki. 2008. Semi-supervised sequential labeling and segmentation using giga-word scale unlabeled data. In *Proceedings of Annual Meeting of the Association for Computational Linguistics*, pages 665–673.

Erik F. Tjong Kim Sang and Sabine Buchholz. 2000. Introduction to the conll-2000 shared task: Chunking. In *Proceedings of the Forth Conference on Computational Natural Language Learning*, pages 127–132.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Computational Natural Language Learning*, pages 142–147.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of Annual Meeting of the Association for Computational Linguistics*, pages 384–394.

Ralph Weischedel and Ada Brunstein. 2005. Bbn pronoun coreference and entity type corpus. *Linguistic Data Consortium, Philadelphia*, 112.

Bishan Yang and Claire Cardie. 2012. Extracting opinion expressions with semi-markov conditional random fields. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1335–1345. Association for Computational Linguistics.

Han Zhao, Zhengdong Lu, and Pascal Poupart. 2015. Self-adaptive hierarchical sentence model. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence*, pages 4069–4076.

Jun Zhu, Zaiqing Nie, Ji-Rong Wen, Bo Zhang, and Hsiao-Wuen Hon. 2007. Webpage understanding: an integrated approach. In *Proceedings of SIGKDD Conference on Knowledge Discovery and Data Mining*.