ACTA UNIVERSITATIS UPSALIENSIS
Studia Linguistica Upsaliensia
21

Segmenting and Tagging Text with Neural Networks

Yan Shao

UPPSALA
UNIVERSITET

Dissertation presented at Uppsala University to be publicly examined in Humanistiska teatern, Thunbergsvägen 3, Uppsala, Saturday, 9 June 2018 at 10:15 for the degree of Doctor of Philosophy. The examination will be conducted in English. Faculty examiner: Assistant Professor Yue Zhang (Singapore University of Technology and Design).

Abstract
Shao, Y. 2018. Segmenting and Tagging Text with Neural Networks. *Studia Linguistica Upsaliensia* 21. 76 pp. Uppsala: Acta Universitatis Upsaliensis. ISBN 978-91-513-0340-6.

Segmentation and tagging of text are important preprocessing steps for higher-level natural language processing tasks. In this thesis, we apply a sequence labelling framework based on neural networks to various segmentation and tagging tasks, including sentence segmentation, word segmentation, morpheme segmentation, joint word segmentation and part-of-speech tagging, and named entity transliteration. We apply a general neural CRF model to different tasks by designing specific tag sets. In addition, we explore effective ways of representing input characters, such as utilising concatenated n-grams and sub-character features, and use ensemble decoding to mitigate the effects of random parameter initialisation.

The segmentation and tagging models are evaluated in a truly multilingual setup with more than 70 datasets. The experimental results indicate that the proposed neural CRF model is effective for segmentation and tagging in general as state-of-the-art accuracies are achieved on datasets in different languages, genres, and annotation schemes for various tasks. For word segmentation, we propose several typological factors to statistically characterise the difficulties posed by different languages and writing systems. Based on this analysis, we apply language-specific settings to the segmentation system for higher accuracy. Our system achieves substantially better results on languages that are more difficult to segment when compared to previous work. Moreover, we investigate conventionally adopted evaluation metrics for segmentation tasks. We propose that precision should be excluded and using recall alone is more adequate for sentence segmentation and word segmentation. The segmentation and tagging tools implemented along with this thesis are publicly available as experimental frameworks for future development as well as preprocessing tools for higher-level NLP tasks.

Keywords: neural networks, sequence labelling, multilinguality, word segmentation, sentence segmentation, morpheme segmentation, transliteration, joint word segmentation and POS tagging

Yan Shao, Department of Linguistics and Philology, Box 635, Uppsala University, SE-75126 Uppsala, Sweden.

ISSN 1652-1366
ISBN 978-91-513-0340-6
urn:nbn:se:uu:diva-348129 (http://urn.kb.se/resolve?urn=urn:nbn:se:uu:diva-348129)

List of papers

This thesis is based on the following papers, which are referred to in the text by their Roman numerals.

I Yan Shao, Jörg Tiedemann, and Joakim Nivre (2015). Boosting English-Chinese Machine Transliteration via High Quality Alignment and Multilingual Resources. In *Proceedings of the Fifth Named Entity Workshop, joint with 53rd ACL and the 7th IJCNLP*, Beijing, China, pages 56–60.

II Yan Shao and Joakim Nivre (2016). Applying Neural Networks to English-Chinese Named Entity Transliteration. In *Proceedings of the Sixth Named Entity Workshop, joint with 54th ACL*, Berlin, Germany, pages 73–77.

III Yan Shao, Christian Hardmeier, Jörg Tiedemann, and Joakim Nivre (2017). Character-Based Joint Segmentation and POS Tagging for Chinese using Bidirectional RNN-CRF. In *Proceedings of the 8th International Joint Conference on Natural Language Processing*, Taipei, Taiwan, China, pages 173–183.

IV Yan Shao, Christian Hardmeier, and Joakim Nivre (2017). Recall is the Proper Evaluation Metric for Word Segmentation. In *Proceedings of the 8th International Joint Conference on Natural Language Processing*, Taipei, Taiwan, China, pages 86–90.

V Yan Shao (2017). Cross-Lingual Word Segmentation and Morpheme Segmentation as Sequence Labelling. In *Proceedings of MLP 2017: The First Workshop on Multi-Language Processing in a Globalising World*, Dublin, Ireland, pages 75–80.

VI Yan Shao, Christian Hardmeier, and Joakim Nivre (2017). Universal Word Segmentation: Implementation and Interpretation. *Submitted*.

VII Miryam de Lhoneux, Yan Shao, Ali Basirat, Eliyahu Kiperwasser, Sara Stymne, Yoav Goldberg, and Joakim Nivre (2017). From Raw Text to Universal Dependencies - Look, No Tags! In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, Vancouver, Canada, pages 207–217.

Reprints were made with permission from the publishers.

Contents

1 Introduction 9
1.1 Segmentation and Tagging as Sequence Labelling 9
1.2 Research Questions 11
1.3 Contributions 12
1.4 Thesis Outline 12

2 Framework 13
2.1 Introduction to Basic Neural Network Models 13
2.2 BiRNN-CRF Overview 17
2.3 Character Representations 17
2.3.1 Unigram Character Embeddings 17
2.3.2 Concatenated N-grams 19
2.3.3 Sub-Character-Level Features 20
2.4 Bi-Directional Recurrent Layers 20
2.5 CRF Interface and End-to-End Training 21
2.6 Ensemble Decoding 22
2.7 Implementation 23
2.7.1 Bucket Model 23
2.7.2 Hyper-Parameters 24
2.8 Complementary Models and Tools 25
2.8.1 Convolutional Neural Network 25
2.8.2 Attention-Based Encoder-Decoder 25
2.8.3 M2M Aligner 26
2.8.4 Moses 26
2.9 Summary 26

3 Applications 27
3.1 Sentence Segmentation 27
3.2 Word Segmentation 28
3.3 Morpheme Segmentation 33
3.4 Joint Word Segmentation and POS Tagging 33
3.5 Named Entity Transliteration 35
3.6 Evaluation Metrics 38
3.6.1 Segmentation and Tagging Metrics 38
3.6.2 Transliteration Metrics 40
3.7 Summary 41

4 Related Work 42
4.1 Sentence Segmentation 42

4.2 Word Segmentation 42
4.3 Morpheme Segmentation 48
4.4 Joint Word Segmentation and POS Tagging 49
4.5 Named Entity Transliteration 52
4.6 Character Representations 53
4.7 POS Tagging and Named Entity Recognition 54
4.8 General Neural CRF Models in Other Tasks 58
4.9 Summary 58

5 Conclusions 60
5.1 Main Conclusions 60
5.2 Future Work 64
5.3 Final Remarks 65

References 66

Acknowledgements

I am truly grateful to many people who have supported my doctoral studies. First and foremost, I would like to thank my main supervisor Professor Joakim Nivre, who guided my research with countless valuable insights, while allowing me to wander freely in the fairyland of deep learning and natural language processing. His vast knowledge and competence in the field as well as his attitude of excellence towards academic research in general have influenced me in a profound way. He is and will always be my role model both inside and outside of my academic career.

I am also thankful to Professor Jörg Tiedemann, my second supervisor. Jörg provided so many constructive ideas and various types of literature, datasets and tools related to my work, which helped me substantially, particularly at the early stage of my doctoral studies.

In addition, I am indebted to my assistant supervisor Christian Hardmeier, with whom I could freely discuss both concrete research problems and wild research ideas for the future. I have learnt many things from him, including fundamental knowledge of machine learning, different approaches to text processing, efficient ways of programming and academic writing skills.

Special thanks must also go to Robert Östling and Aaron Smith, who have thoroughly proofread the thesis draft and given beneficial comments and feedback in addition to Joakim and Christian, which greatly helped me revise the thesis into its final form.

Thanks to all my colleagues and fellow doctoral students at the Department of Linguistics and Philology at Uppsala University, including Aaron Smith, Aleksandrs Berdicevskis, Alexander Nilsson, Ali Basirat, Abao Aman, Bielike Ayijiang, Antonios Pontoropoulos, Beáta Megyesi, Buket Öztekin, Emil Lundin, Erik Mo Welin, Eva Pettersson, Fabienne Cap, Fredrik Sixtensson, Fredrik Wahlberg, Tang Gongbo, Harald Hammarström, Helena Löthman, Josefin Lindgren, Karin Koltay, Klara Bertils, Lena Rydholm, Linnéa Öberg, Marc Tang, Marie Dubremetz, Mats Dahllöf, Miryam de Lhoneux, Mojgan Seraji, Padideh Pakpour, Per Starbäck, Pär Eliasson, Rima Haddad, Samuel Douglas, Sara Stymne, Sharid Loáiciga, Simon Magnusson, Tuomo Nuorluoto, Vera Wilhelmsen, Wan Xinzheng and Gong Zhengxian. Thank you all for creating such an unbelievable working environment for me. All the delightful experiences that I had will be among the best memories of my life. Additionally, thanks to Georg Sundelin, Inga-Lill Holmberg, Jenny Rahbek and Johan Heldt who handled all the administrative and bureaucratic work.

I would also like to thank all my friends in Sweden, China and other parts of the world. They were always there to inspire me and cheer me up, like the sunlight that melts my heart in the freezing cold.

The experiments performed in this thesis required substantial computational resources. Thanks to the NeIC-NLPL project and SNIC, I could use the GPU clusters provided by CSC in Helsinki (Taito), Sigma2 in Oslo (Abel) and Lunarc in Lund (Erik) to run all the computationally expensive experiments and obtain results in time.

My doctoral work is primarily funded by the Chinese government via the China Scholarship Council. I feel blessed to be born and grow up in such a great country with rapid development and strong devotion to research. Thanks to the support from the government, thousands of young people like me from China are able to pursue their academic dreams abroad.

Last but not least, I am very grateful to my parents. They are always encouraging and supporting me whole-heartedly, far away from my homeland. Thanks for their constant love and understanding.

Shao Yan
Uppsala, Sweden
April 2, 2018

1. Introduction

With the rapid development of the Internet and computational technologies, human society is driven more and more by massive amounts of data of various types, including large amounts of text in natural language. We can extract rich information from structured and analysed text and use it to support different applications that help our daily life. However, due to the great flexibility, variety and ambiguity of natural language, it remains a great challenge to efficiently and effectively process raw text into the structured format that is required by computational models.

Segmentation and tagging are two initial steps to retrieve information from plain text. Given the raw input as a continuous string of characters, it is non-trivial for a computer to disambiguate and identify meaningful elements, such as words, morphemes, phrases, sentences, as well as linguistic categories like parts of speech (POS). Such extracted information is beneficial to many higher-level natural language processing (NLP) tasks like information retrieval, machine translation and text understanding. Figure 1.1 illustrates the raw input text and the output information to be obtained by the segmentation and tagging systems.

In recent years, deep neural networks have shown great advantages over traditional statistical models and achieve strong empirical evaluation results in various fields, including computer vision, speech technology and nearly all sub-fields of NLP. Instead of relying on hand-crafted discrete features, the core input information is encoded as dense vector representations and passed to a series of artificial neural networks that automatically learn the best feature combination and representation with respect to the output.

In this thesis, we investigate how to effectively apply neural networks to sequence segmentation and tagging problems in NLP, including sentence segmentation, word segmentation, morpheme segmentation, joint word segmentation and POS tagging as well as named entity transliteration. We aim to build accurate segmentation and tagging systems that are practically useful to a wide range of higher-level NLP tasks.

1.1 Segmentation and Tagging as Sequence Labelling

Natural language text in digital form is a linear string of characters. Thus, machine learning algorithms for sequence labelling are widely applied to solve both segmentation and tagging tasks for natural languages.

Input:
If possible I try the services on myself before I bring in my son. Drs. Ali work wonders.

Output:
If_**SCONJ** possible_**ADJ** I_**PRON** try_**VERB** the_**DET** services_**NOUN** on_**ADP** myself_**PRON** before_**SCONJ** I_**PRON** bring_**VERB** in_**ADV** my_**PRON** son_**NOUN** ._**PUNCT**

Drs._**PROPN** Ali_**PROPN** work_**VERB** wonders_**NOUN** ._**PUNCT**

Figure 1.1. The raw input text is segmented into sentences and words. The words are labelled with POS tags according to Universal Dependencies (Nivre et al., 2016).

In machine learning, sequence labelling means assigning categorical tags to each element in the input sequence. The mapping from individual input elements to output tags is usually ambiguous, as the same input may have multiple possible output tags. The output tags are not only determined by the corresponding input elements, but also by the contextual input information and the surrounding output tags. By applying machine learning algorithms, the mapping between the input sequence and the output sequence is inferred by mathematical models derived from data rather than rules hand-crafted by human experts.

POS tagging is a typical sequence labelling task, where given an input word at each time step, the associated POS tag is predicted. A substantial portion of words have multiple possible POS tags. For instance, *work* can be tagged as a NOUN or a VERB depending on the context. In addition, the predicted POS tags are inter-dependent. In many languages, a VERB is more likely to be followed by a NOUN than another VERB. Thus, we often apply machine learning models that incorporate transition information between consecutive tags to solve the problem. Instead of making independent predictions for each element of the sequence, structured linear sequence labelling models, such as hidden Markov models (Rabiner, 1989) and conditional random fields (CRF) (Lafferty et al., 2001), optimise the prediction globally so that the optimal tags over the entire sequence are obtained.

Segmentation can also be modelled as a character-level sequence labelling task. Given the input as a string of characters, we predict tags encoding the boundaries between the basic units to be segmented. For instance, we use binary tags to indicate whether a character is located at the end of a sentence for sentence segmentation. For word segmentation and morpheme segmentation, more fine-grained tags can be adopted to add more explicit position information to the boundary tags for higher prediction accuracy. For example, we can use four tags to indicate characters that are located at

the beginning, middle, or end of a word/morpheme or as a single character word/morpheme.

The input to POS tagging is a sequence of segmented words. Performing word segmentation and tagging separately comes with a risk of error propagation. Combining certain subtasks in a joint model can reduce that risk. For languages without space as word delimiter, such as Chinese, Japanese and Vietnamese, the POS tags can be combined with the boundary tags and joint word segmentation and POS tagging thus becomes a regular character-level sequence labelling task that involves predicting the combinatorial tags.

Additionally, named entity transliteration can be approached by two sequence labelling tasks in a pipeline fashion, apart from applying general sequence transduction models. The source name string is first divided into basic transliteration units in the segmentation stage and the segmented units are then transduced into the target language in the mapping stage.

1.2 Research Questions

In this thesis, we study segmentation and tagging of text using neural networks. We explore the following research questions:

1. How can we apply artificial neural networks as sequence labelling models to a wide range of segmentation and tagging tasks, without heavy feature selection and hyper-parameter tuning? We aim to build a relatively general and flexible framework which can be easily adapted to different tasks and languages.

2. How can we represent characters, particularly logograms like Chinese characters, as the input of the neural network models? We explore different ways to encode information in character representations, in addition to applying neural networks for feature extraction.

3. How does the proposed neural network model perform on different tasks and languages with shared core hyper-parameters? How should we modify and adapt the general model in specific scenarios for higher accuracy? We investigate a more general and systematic way of adjusting the model than tuning individually on each dataset.

4. How does the difficulty of word segmentation relate to statistical properties of languages and writing systems? We expect that the information derived from the statistics can be utilised to improve word segmentation accuracy.

5. How can we effectively evaluate the segmentation tasks? We aim to find out if conventionally employed evaluation metrics are adequate for various segmentation tasks.

1.3 Contributions

The contributions of this thesis and the included publications are mainly related to applying neural network models as a sequence labelling framework to segmentation and tagging, with some theoretical analysis on linguistic properties and writing systems across different languages as well as evaluation metrics for segmentation tasks:

1. We successfully apply neural network models to a number of segmentation and tagging problems in NLP. Our approaches yield state-of-the-art accuracies on various tasks according to the empirical evaluation on datasets in different languages, genres, and annotation schemes.
2. We propose and explore novel ways of representing characters in neural network models, including applying concatenated n-gram models to encode rich local information as well as utilising graphical sub-character-level information extracted from logograms.
3. We build a word segmentation system that is applicable to any language. In addition, the correlation between word segmentation accuracy and properties of writing systems is thoroughly investigated, which is helpful in interpreting the gaps between word segmentation accuracies across different languages as well as selecting language-specific settings to the segmentation system for higher accuracy.
4. The drawbacks of conventional evaluation metrics for word segmentation are analysed, both theoretically and experimentally.
5. We provide implementations of the proposed models,[1] which can be used both as experimental frameworks for future development and practical segmentation and tagging tools for other NLP tasks.

1.4 Thesis Outline

The remaining part of the compilation thesis is organised as follows: Chapter 2 describes the neural network architectures that have been applied as the sequence labelling framework for various segmentation and tagging tasks. Chapter 3 applies the general segmentation framework to specific segmentation and tagging tasks in NLP, including sentence segmentation, word segmentation, joint word segmentation and POS tagging as well as machine transliteration. The experimental results presented in the included papers are briefly summarised in Chapters 2 and 3 as well. Chapter 4 gives a general overview of related neural network models and previous work on segmentation and tagging. Chapter 5 concludes the thesis and proposes research orientations for the future.

[1] https://github.com/yanshao9798

2. Framework

Segmentation and tagging with sequence labelling models can be challenging. As discussed in the previous chapter, for sequence labelling, the output tag to be predicted at a specific position is dependent on both the corresponding input information and the neighbouring tags. Figure 2.1 illustrates the difficulty of Chinese word segmentation. To correctly predict the word boundaries, it is occasionally necessary to use long-distance information from the input characters, which is not available from the local features. To model such tasks, we need a sequence labelling model that not only integrates the transition between output labels, but also incorporates global features over the entire sequence.

In this chapter, we introduce the fundamental neural network models that are associated with the tasks. We present the basic mathematical formulations and explain how to train the neural network models with labelled data. The neural CRF model, adopted as our main sequence labelling framework for the segmentation and tagging tasks, is described in detail afterwards. Some complementary models and tools are included as well. In addition to introducing the general neural network models in related literature, we describe concatenated n-grams, sub-character-level features and ensemble decoding for neural CRF models. They are originally presented and explored in the papers included in the thesis.

2.1 Introduction to Basic Neural Network Models

Artificial neural networks (NNs) (Figure 2.2) are computational models inspired by biological neural networks. An NN consists of multiple layers. Each layer has multiple computation units (neurons) which first apply a linear function to the input vector with a weight vector and a scalar bias and then apply non-linear activation functions f, such as sigmoid ($\frac{e^x}{1+e^x}$), hyperbolic tangent ($\frac{e^x-e^{-x}}{e^x+e^{-x}}$) or rectified linear unit ($\max(0,x)$). The output h of an NN layer can be represented as:

$$h = f(W \cdot x + b) \tag{2.1}$$

where x is the input vector, W is the weight matrix and b is the bias vector. Compared to linear models, NNs with non-linear activation are capable of learning effective feature representations and arbitrary non-linear decision boundaries.

夏天 (summer), 能 (can) 穿 (wear) 多 (more) 少 (little) 穿 (wear) 多 (more) 少 (little) .
(Wear as little as you can in the summer.)

冬天 (winter), 能 (can) 穿 (wear) 多少 (amount) 穿 (wear) 多少 (amount) .
(Wear as much as you can in the winter.)

Figure 2.1. Chinese word segmentation with long-term dependencies (Chen et al., 2015b). Depending on the collocated terms 夏天 (summer) and 冬天 (winter), the Chinese character sequence 能穿多少穿多少 can be segmented differently into 能 / 穿 / 多 / 少 / 穿 / 多 / 少 and 能 / 穿 / 多少 / 穿 / 多少 that have opposite meanings.

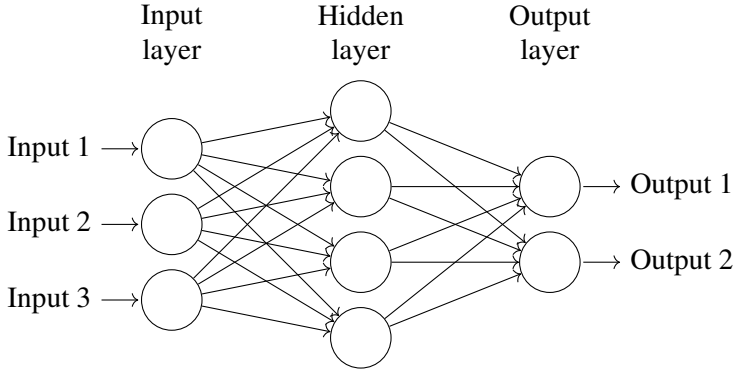

Figure 2.2. A simple neural network with one hidden layer.

In feed-forward neural networks, the information flows in only one direction. The associated layers are acyclically stacked. By contrast, recurrent neural networks (RNNs) use self-connections to form loops and hold information in memory. The information is propagated in a temporal fashion. As shown in Figure 2.3, the hidden state h_{t-1} of step $t-1$ is concatenated as partial input of step t so that the information flows over the entire sequence, which makes RNNs very suitable for sequence modelling. A simple RNN can be represented as:

$$h_t = f(W \cdot [h_{t-1}, x_t] + b) \tag{2.2}$$

$[h_{t-1}, x_t]$ indicates that h_{t-1} is concatenated with the input x_t of the current step t. Moreover, RNNs equipped with gated recurrent cells, such as long-short term memory (LSTM) (Hochreiter and Schmidhuber, 1997) or gated recurrent unit (GRU) (Cho et al., 2014) (Figure 2.4) are capable of rescaling the information between different time steps and capturing long-distance

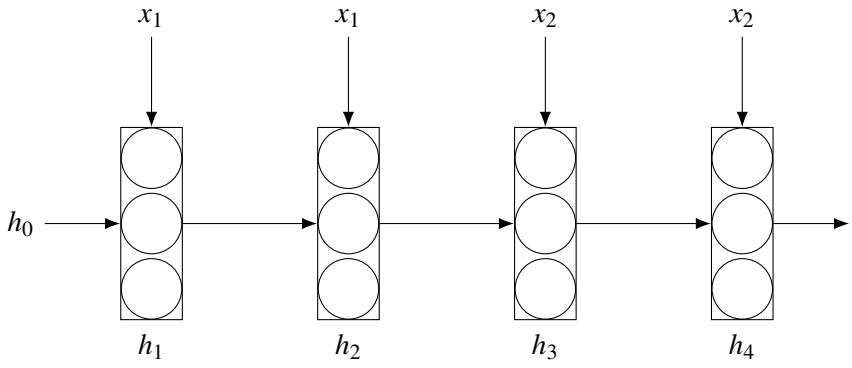

Figure 2.3. An unfolded simple recurrent neural network.

dependencies more effectively. A GRU cell is defined as follows:

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + b_z) \tag{2.3}$$

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + b_r) \tag{2.4}$$

$$\widetilde{h}_t = \tanh(W_h \cdot [r_t \otimes h_{t-1}, x_t] + b_h) \tag{2.5}$$

$$h_t = (1 - z_t) \otimes h_{t-1} + z_t \otimes \widetilde{h}_t \tag{2.6}$$

W_z, W_r, W_h and b_z, b_r, b_h are the weight matrices and bias vectors to be trained. h_{t-1} is the hidden state of the previous time step and x_t is the current input. The gates are implemented as sigmoid activations σ to control the information flow between the hidden states of different time steps. The vectors computed with h_{t-1} and x_t are scaled non-linearly into the range $(0,1)$ by the sigmoid functions. The reset gate r_t regulates how much information from h_{t-1} should be incorporated to compute the intermediate state $\widetilde{h}_t$ by applying element-wise multiplication $\otimes$. Additionally, h_{t-1} and $\widetilde{h}_t$ are further balanced by the update gate z_t to obtain the final representation h_t of the current time step.

NNs are usually trained with labelled data via error back-propagation and mini-batch stochastic gradient descent (SGD) optimisation. We define a loss function *Loss* with a batch of n training samples (x_i, y_i) as well as the weights of the network θ as:

$$Loss(\theta) = \frac{1}{n} \sum_{i=1}^{n} Loss_i(y_i', y_i, \theta) \tag{2.7}$$

where y_i' is the predicted output by the network with the current weights θ. The most common variants of $Loss_i(y_i', y_i, \theta)$ include time-wise cross-entropy loss, mean square error loss, hinge loss as well as sequence-level CRF loss.

θ is updated towards minimising $Loss(\theta)$ as:

$$\theta_t = \theta_{t-1} - \eta \nabla Loss(\theta) \tag{2.8}$$

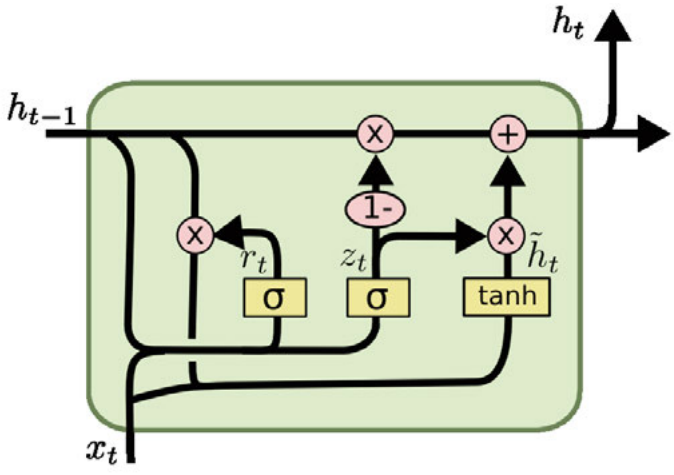

Figure 2.4. A gated recurrent unit (GRU) cell (Olah, 2015).

where η is the learning rate. The gradient $\nabla Loss(\theta)$ is a vector-valued function composed of partial derivatives of individual parameters. A specific parameter θ_k is updated as:

$$\theta_{t,k} = \theta_{t-1,k} - \eta \cdot \frac{\partial Loss(\theta)}{\partial \theta_k} \tag{2.9}$$

The training errors are back-propagated to compute $\nabla Loss(\theta)$ for updating the weights θ of the neural network. The gradient descent procedure terminates when $Loss(\theta)$ is sufficiently close to zero or some other stopping criterion is satisfied.

In practice, SGD optimisers with adaptive learning rate, such as Adagrad (Duchi et al., 2011) and Adam (Kingma and Ba, 2015), are often employed for training RNNs. For instance, in Adagrad, η is rescaled with G_t while updating the parameters:

$$\theta_{t,k} = \theta_{t-1,k} - \frac{\eta}{\sqrt{G_{t-1,k} + \varepsilon}} \cdot \frac{\partial Loss(\theta)}{\partial \theta_k} \tag{2.10}$$

$G_t \in \mathbb{R}^{d \times d}$ is a diagonal matrix, where each diagonal element $G_{t,k}$ is the sum of the squares of the gradients with respect to θ_k up to time step t. ε is a smoothing term that avoids zero division.

Dropout (Srivastava et al., 2014) is a simple regularisation technique that is commonly applied during training to mitigate overfitting. At training time, some neurons are randomly dropped (disconnected from the network) with a probability p so that the neural network becomes less sensitive to some specific neurons. Thus, more general and independent representations can be learned. The output h of the associated layer is rescaled with respect to p as $\frac{h}{p}$. Dropout is suppressed at decoding time.

NNs can be easily implemented thanks to deep learning libraries that are based on computational graphs, in which the nodes represent basic variables or computations and the edges represent their dependencies. The gradients over the entire graph can be summed and factorised to train the neural networks efficiently.

2.2 BiRNN-CRF Overview

To integrate global features into the sequence labelling framework, we adapt and apply bi-directional RNNs (BiRNNs) with a conditional random fields (CRF) (Lafferty et al., 2001) interface (BiRNN-CRF) as the fundamental sequence labelling framework for our segmentation and tagging tasks. The framework was originally proposed by Huang et al. (2015). Different variations of BiRNN-CRFs have been applied to a number of sequence labelling tasks, such as named entity recognition, POS tagging and text normalisation. The BiRNN-CRF is a type of neural CRF model, with an RNN part that extracts global numerical features for the CRF layer. In our models, the generality of BiRNN-CRFs is retained with minimal task-specific and language-specific settings.

Figure 2.5 is a general overview of the adapted BiRNN-CRF model. The input characters are first mapped to vector representations. The character representations will be introduced in detail in section 2.3. The character vectors are fed to the recurrent layers. The outputs of the recurrent layers are used as numerical features for the first-order chain CRF layer to obtain the optimal tags over the entire input sequence. Dropout is applied to both the inputs and the outputs of the bidirectional recurrent layers. The output tags y_i to be predicted are specific to different tasks, and will be introduced in Chapter 3. We can retrieve the segmentation and tagging information from the output tags.

2.3 Character Representations

Different character representations can be used independently or concatenated in the neural network architecture. In NLP, the mapping from atomic characters/words to dense vector representations is often referred to as embedding. In this thesis, we explore three different ways of representing input characters as dense vectors.

2.3.1 Unigram Character Embeddings

The most straightforward character representation is to map a character to a randomly initialised vector. The initial random vectors do not capture any

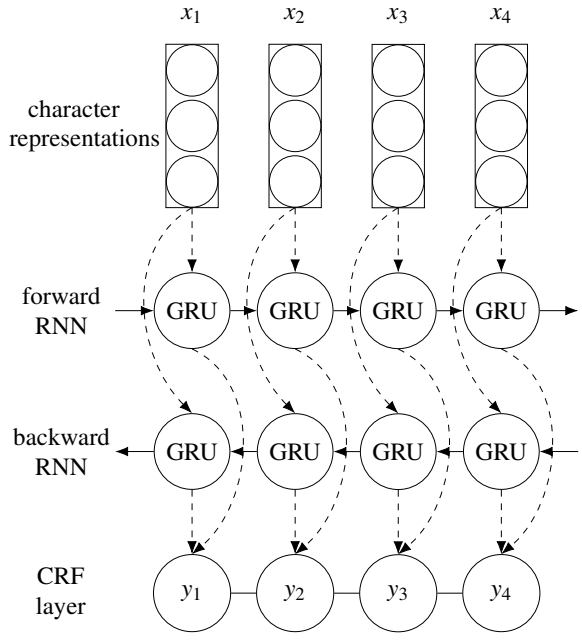

Figure 2.5. The BiRNN-CRF model for segmentation and tagging. The dashed arrows indicate that dropout is applied.

correlations between different characters, but are updated by back-propagating the gradients while optimising the overall loss function. For most alphabetic languages with small character sets, the random character embeddings can be tuned effectively during training.

However, for languages with rich character sets, such as Chinese and Japanese, it is beneficial to initialise character representations with character embeddings pre-trained on large amounts of plain text to capture the correlations and similarities between the characters. We can treat the separated characters as words and employ the prevalent models for training word embeddings, such as continuous bag-of-words and skip-gram in word2vec (Mikolov et al., 2013) or GloVe (Pennington et al., 2014) to obtain the pre-trained character embeddings based on character co-occurrences. Table 2.1 shows sample characters and their most similar characters according to the cosine similarity of their vector representations. We can see that the semantic similarities are well captured by the GloVe vectors. The pre-trained character embeddings are fine-tuned while training. According to the evaluation results in Paper III, pre-trained character embeddings are more effective if the training set is small and concatenated n-grams, which will be introduced next, are not used.

Table 2.1. *Sample Chinese characters and their most similar characters with respect to cosine similarity of GloVe vectors.*

路 (road)	1.0	天 (sky)	1.0	县 (county)	1.0
线 (line)	0.7448	光 (light)	0.4765	市 (city)	0.7267
道 (path)	0.6388	明 (bright)	0.4644	镇 (town)	0.7027
站 (station)	0.6369	星 (star)	0.4617	郡 (canton)	0.6747
街 (street)	0.6178	海 (sea)	0.4530	乡 (village)	0.6580
桥 (bridge)	0.6072	月 (moon)	0.4474	区 (district)	0.6533

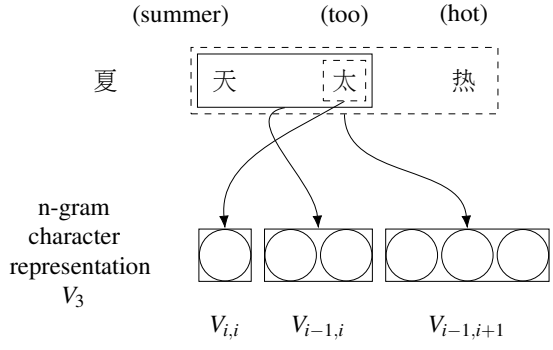

Figure 2.6. Concatenated 3-gram model. The third character is the pivot character in the given context.

2.3.2 Concatenated N-grams

To encode rich local information in the character representations, we concatenate the vector representations of n-grams of different orders instead of using unigram characters alone, even though RNNs are expected to capture contextual information from unigram character embeddings.

Figure 2.6 shows how the concatenated 3-gram representation of the pivot character 太 is obtained. We use $V_{m,n}$ to denote the $(n-m+1)$-gram that is composed by the m-th to n-th characters in the input sequence. The representation of the pivot character in the given context is the concatenation of the unigram character embedding $V_{i,i}$ of 太 along with $V_{i-1,i}$ of the bigram 天太 as well as $V_{i-1,i+1}$ of the trigram 天太热. The n-gram $V_{m,n}$ of different orders, such as $V_{i,i}$, $V_{i-1,i}$, and $V_{i-1,i+1}$, are randomly initialised separately and updated similarly as random character embeddings. The concatenated n-grams ensure that the same character has different yet closely related representations in different contexts.

For each n-gram order, we use a single vector to represent the terms that appear only once in the training set while training. These vectors are later used as the representations for unknown characters and n-grams in the development and test sets.

The experimental results in Paper III show that substantial improvements can be achieved for joint word segmentation and POS tagging for Chinese

by applying the concatenated n-grams. Similar improvements are observed for Chinese, Japanese and Vietnamese word segmentation in Paper VI. This indicates that not all the local information can be effectively captured by the BiRNN only using unigram character embeddings.

However, concatenating high-order n-grams can sometimes be detrimental, especially on smaller datasets due to the fact that higher-order n-grams are sparser in the training data and their vector representations cannot be trained effectively. Applying higher order n-grams also greatly enlarges the embedding space. We therefore limit the n-gram order to 3 as the optimal setting for the best performing models in Papers III, V and VI.

The evaluation results on word segmentation in Paper V show that concatenated n-grams are less helpful for most space-delimited languages. For a few languages, such as Spanish and Catalan, the concatenated n-gram model is outperformed by the basic unigram character representation.

2.3.3 Sub-Character-Level Features

We explore two ways of utilising sub-character-level features in Paper III. Unlike characters in alphabetic languages, Chinese characters are logograms. There is rich information encoded in the graphical components of the characters.

In the first approach, we look up radical information via the Unicode database of Chinese characters and represent the radicals as randomly initialised vectors that are concatenated as a part of the character representation. The radical of a Chinese character is the shared graphical component that functions as a semantic indicator. For instance, characters with 钅 (gold) as the radical are all somewhat related to metals, such as 银 (silver), 铁 (iron), 针 (needle) and so on.

In the second approach, the characters are rendered as pictures and a series of convolutional neural networks (CNNs) with max-pooling layers are applied to extract graphical features from scratch. The output of the CNNs is reshaped into a vector that can be concatenated to the other character representations. The CNNs are trained jointly with the main BiRNN-CRF network.

The experimental results in Paper III indicate that the improvements due to sub-character features are marginal. For segmentation and tagging tasks, the additional information provided by sub-character-level features seems to be covered by the other features already and therefore does not lead to substantial improvements.

2.4 Bi-Directional Recurrent Layers

Bi-directional RNNs (BiRNNs) (Schuster and Paliwal, 1997) are an extension of RNNs that can be applied to finite sequences to encode more information.

In the proposed sequence labelling model, the character representations are passed to two recurrent layers in both forward and reverse orders. The forward layer contains historical features from proceeding input elements while the backward layer encodes future information. The outputs of the forward layer and backward layer are concatenated so that the complete information over the entire sequence is encoded at every time step.

To enable the RNN to capture long-term dependencies more effectively, we employ recurrent cells equipped with a gating mechanism as in Figure 2.4. We adopt GRU as the basic recurrent cell instead of LSTM, as Chung et al. (2014) show that GRU and LSTM yield similar performance while GRU has fewer parameters.

2.5 CRF Interface and End-to-End Training

NNs can be used as feature extractors and applied jointly with a CRF layer as the output interface. Similarly to linear models for structured prediction, the neural CRF model incorporates transition information between the output tags.

In regular RNNs, the outputs are inferred by adding a time-wise softmax (normalised exponential function) to make an isolated and greedy prediction at each step, which may not be optimal as the transitions between different labels are not considered. The total loss of the RNNs is decomposed into the individual losses of all the time steps.

By contrast, the CRF interface in neural CRF models computes sequence-level loss. We use $x = x_1, x_2, ..., x_n$ and $y = y_1, y_2, ..., y_n$ to represent the input sequence and the output labels. τ is the label set for y_t ($y_t \in \tau$). In addition to the main network, we introduce a matrix $T_{|\tau|\times|\tau|}$ for the transition scores between consecutive labels, where $|\tau|$ equals the number of unique labels. $T_{|\tau|\times|\tau|}(b|a)$ denotes the transition score from label a to label b.

Meanwhile, the outputs of the recurrent layers are linearly transformed with a weight w_l and a bias b_l into a score matrix $S_{n\times|\tau|}$. Regarding the main neural network as a function f_θ, $S_{n\times|\tau|}$ can be computed from x as:

$$S_{n\times|\tau|} = f_\theta(x) \tag{2.11}$$

θ represents all the trainable parameters of the neural network, including w_l and b_l. We use $S_{n\times|\tau|}(y_t|x)$ to indicate the conditional score of the label $y_t \in \tau$ at step t given the input sequence x. The total sequence score of the label sequence y is then the sum of the conditional scores and the transition scores as:

$$score(x, y; \theta) = \sum_{t=1}^{n} (S_{n\times|\tau|}(y_t|x) + T_{|\tau|\times|\tau|}(y_t|y_{t-1})) \tag{2.12}$$

y_0 denotes the start symbol of the sequence. Given the input sequence x and parameter θ, we can find the best label sequence y^* as:

$$y^* = \underset{y' \in \tau^n}{\operatorname{argmax}}\, score(x, y'; \theta) \tag{2.13}$$

The Viterbi algorithm (Forney, 1973) can be used for efficient decoding. Its time complexity is $O(|\tau|^2 n)$, which is linear in the input sequence length n.

The linear CRF model can be trained efficiently in an online perceptron style (Collins, 2002). A similar approach is applicable to train neural CRF models in an end-to-end fashion. Given each training instance (x, y), we can compute $score(x, y; \theta)$ of the tag sequence y under the current parameters θ. Applying equation 2.11 to the input sequence x, the conditional score matrix $S_{n \times |\tau|}$ is obtained. With the transition matrix $T_{|\tau| \times |\tau|}$, the best score $score(x, y^*; \theta)$ can be computed as in 2.13. The sentence-level training objective is therefore defined to minimise the difference between $score(x, y^*; \theta)$ and $score(x, y; \theta)$:

$$Loss(y', y, \theta) = score(x, y; \theta) - score(x, y^*; \theta) \tag{2.14}$$

The parameters θ are updated when the current optimal tag sequence y^* is different from the reference tag sequence y.

The CRF interface is particularly effective if there are strong dependencies between the output tags. However, it slows down the training and decoding speed very significantly as the Viterbi process cannot be parallelised.

2.6 Ensemble Decoding

The weights of the neural networks, including the random character and n-gram embeddings, are initialised using the scheme introduced in Glorot and Bengio (2010) in our experiments. We uniformly sample from $[-\sqrt{\frac{6}{r+c}}, +\sqrt{\frac{6}{r+c}}]$ for layer weights w, where r and c are the input and output dimensions of the layer. The bias b is initialised with zeros. Similarly, we sample from $[-\sqrt{\frac{3}{d}}, +\sqrt{\frac{3}{d}}]$ for the embeddings, where d is the embedding size. This initialisation approach makes sure that the weights of the neural networks are in a reasonable range. When calculating through the neural network, the signals neither become too big nor too small.

The randomness of weight initialisation has a notable impact on the performance of the model. Reimers and Gurevych (2017b) show that the random seed values can result in statistically significant differences ($p < 10^{-4}$) for neural network models.[1] For the two BiRNN-CRF-based named entity recognition systems presented by Lample et al. (2016) and Ma and Hovy

[1] p-value is calculated by using the Bonferroni correction (Bland and Altman, 1995).

(2016) respectively, an absolute difference of one percentage point F1-score is observed depending on the random values under identical hyper-parameters settings. This difference is large enough to determine if the system is state-of-the-art or mediocre.

We use a simple averaging technique to mitigate the deviations induced by random weight initialisation of the neural network at the final decoding phase. For the chain CRF decoder, the final sequence of the predicted tags y is obtained via the conditional scores $S_{n\times|\tau|}(y_t|x)$ and the transition scores $T_{|\tau|\times|\tau|}(y_t|y_{t-1})$ given the input sequence x as in equation 2.13.

For ensemble decoding, instead of computing the optimal sequence y^* with respect to the scores returned by a single model, both the conditional scores $S_{n\times|\tau|}(y_t|x)$ and transition scores $T_{|\tau|\times|\tau|}(y_t|y_{t-1})$ are averaged over four models with identical parameter settings that are trained independently with different random seeds:

$$score(x,y;\theta_{[1:4]}) = \sum_{t=1}^{n}(\bar{S}_{n\times|\tau|}(y_t|x) + \bar{T}_{|\tau|\times|\tau|}(y_t|y_{t-1})) \tag{2.15}$$

The experimental results in Papers III and IV show that systematic and statistically significant improvements can be obtained via ensemble decoding. The improvement is more substantial if the training set is smaller.

2.7 Implementation

To implement neural network more efficiently, we use the bucket model for processing sentences of varying lengths with static computational graphs. In addition, we present the core hyper-parameters that are shared across different tasks in the included papers.

2.7.1 Bucket Model

We have adopted TensorFlow (Abadi et al., 2016) to implement the neural networks for the experiments. Unlike some other libraries for building neural networks, such as Torch (Collobert et al., 2002) and DyNet (Neubig et al., 2017), which construct dynamic computational graphs on the fly with respect to the lengths of the training sentences, TensorFlow pre-defines a static computational graph and repetitively executes the same graph while training. The length of the RNNs is difficult to adjust dynamically. Moreover, it is not efficient to define a single graph and pad all the training instances to the length of the longest sentence in the dataset.

We apply the bucket model to train the RNN-based models more efficiently. Sentences with similar lengths are grouped into the same buckets and sentences in the same bucket are padded to the same length. We construct

Table 2.2. *Core hyper-parameters that are shared cross different segmentation and tagging tasks.*

Character embedding size	50
N-gram embedding size	50
GRU state size	200
BiRNN layers	1
Optimizer	Adagrad
Initial learning rate	0.1
Decay rate	0.05
Gradient Clipping	5.0
Dropout rate	0.5
Batch size	10

sub-computational graphs for each bucket. At training time, the buckets are shuffled first. For each bucket, the training instances inside are shuffled and fed into the neural networks as mini-batches. Both the training and tagging speed of our neural network can be drastically improved thanks to the bucket model.

2.7.2 Hyper-Parameters

There are a number of hyper-parameters that have an impact on the performance of the neural network models. By exploring with the development sets of different tasks as well as referring to the parameter settings reported in related work (Ma and Hovy, 2016; Reimers and Gurevych, 2017a), we use a single set of core hyper-parameters as introduced in Table 2.2 in different tasks. Although fine-tuning the hyper-parameters on specific tasks and individual languages might result in additional improvements, we aim to build adaptive models that generalise well on different tasks, languages and annotation schemes of the datasets that are applied to train the models.

The character embedding size should be selected with respect to the character set size. For languages with large character sets like Chinese and Japanese, 50 is big enough for segmentation and tagging tasks in the BiRNN-CRF framework. For most alphabetic languages with much smaller character sets, a smaller size can be adopted for less parameters and computations. For n-gram embeddings, we adopt 50 as the size for any order.

We use a single BiRNN with 200 as the GRU state size. Our experiments for tuning the hyper-parameters also show that neither enlarging the GRU state size nor stacking more BiRNNs lead to further improvement as the model overfits with too many parameters. Stacking more BiRNN layers may even hurt the performance drastically if the training set is not large enough.

We employ Adagrad as the optimiser. With an adaptive learning rate, it is less sensitive to hyper-parameters and helps the model converge faster when employed for training RNNs, compared to the regular stochastic gradient

descent optimiser (SGD). We also apply learning rate decay as $\eta_t = \frac{\eta_0}{\rho(t-1)+1}$, where t is the current epoch and ρ is the decay rate.

In our experiments, the models are trained for 30 epochs. After each epoch, the performance of the model is measured with the relevant evaluation metrics on the development sets. We pick the weights of the epoch that maximise the evaluation score on the development sets as the final weights and apply the model to the final test sets.

2.8 Complementary Models and Tools

There are a few additional models and tools that are used in some experiments apart from the BiRNN-CRF model.

2.8.1 Convolutional Neural Network

A convolutional Neural Network (CNN) is a type of feed-forward neural networks, in which the convolutional layers extract features locally with relatively small filters (small feed-forward neural networks) that traverse the input. Compared to regular fully-connected feed-forward neural networks, a CNN substantially reduces the number of free parameters and allows the network to be deeper with fewer parameters. A max-pooling layer is often applied to the output of a convolutional layer, in which a small filter is applied to the input and returns the maximum number in every subregion that the filter convolves around.

As mentioned in section 2.3.3, CNNs are applied to extract orthographical features from Chinese characters. In addition, 1-dimensional CNNs are employed in Paper II to extract and filter character-level information from segmented transliteration units. The CNNs are applied jointly with RNNs.

2.8.2 Attention-Based Encoder-Decoder

Attention-based encoder-decoders (Bahdanau et al., 2015) are primarily applied to neural machine translation but can be used as general sequence-to-sequence transducers. In an encoder-decoder, the input sequence is encoded first and the decoder generates the corresponding output sequence afterwards. Unlike the basic encoder-decoder presented in Sutskever et al. (2014), which encodes the input sequence as the final state of RNN encoder, the attention mechanism generates a weight vector at each decoding step to incorporate the most relevant information from all the states of the encoder. Apart from using RNNs as the encoder and decoder, CNNs (Gehring et al., 2017) and fully-attention-based approaches (Vaswani et al., 2017) are also applicable.

In Paper VI, we use an attention-based encoder-decoder equipped with LSTMs to transduce non-segmental units in some languages for universal word segmentation.

2.8.3 M2M Aligner

The M2M Aligner (Jiampojamarn et al., 2007) is an unsupervised many-to-many sequence alignment model. Its principal algorithm is based on the stochastic transducer in Ristad and Yianilos (1998). In Papers I and II on named entity transliteration, we use the M2M Aligner to obtain the segmentation and alignments between transliteration units. We also apply a greedy local expectation-maximisation (EM) approach to further improve the segmentation and alignment quality of the M2M Aligner.

2.8.4 Moses

Moses (Koehn et al., 2007) is a phrase-based statistical machine translation system. Similar to the neural encoder-decoder, it can be applied as a general sequence transducer. In Paper I, Moses is used as the fundamental framework for the phrase-based machine transliteration system.

2.9 Summary

In this chapter, we have introduced the general NNs, including basic neural network structures and procedures for training the weights via error back-propagation and mini-batch SGD using labelled data. Moreover, we have described the BiRNN-CRF model, our fundamental framework for segmentation and tagging, as well as some complementary models. In the next chapter, we will explain how we can apply the basic framework to different segmentation and tagging tasks.

3. Applications

In this chapter, we apply the general sequence labelling framework introduced in Chapter 2 to various segmentation and tagging tasks with different tag sets. We also summarise some of the results reported for these tasks in the included papers. Figure 3.1 summarises the relations between the segmentation and tagging tasks that are covered in this thesis. The adjacent tasks are occasionally modelled jointly to mitigate error propagation and achieve higher accuracy. In addition, named entity transliteration is modelled as a segmentation and tagging task in a pipeline fashion.

3.1 Sentence Segmentation

A sentence consists of a sequence of words that is complete in itself. It conveys a statement, question, exclamation, or command, and usually consists of a main clause and sometimes one or more subordinate clauses. Sentence segmentation is the task of dividing a text into its constituent sentences. It is also known as sentence boundary identification. The punctuation that often terminates a sentence is very informative for sentence segmentation. However, solely relying on terminating punctuation as signalling the end of sentences does not always result in adequate segmentation.

Punctuation can be ambiguous. For instance, the full stop in English can also appear in abbreviations like *Mr. Green*. Using abbreviation lists that contain periods can prevent some incorrect segmentations in this case, but it is very difficult to build such lists that are applicable to multiple languages and various genres and have good coverage. Moreover, the full stop may denote a decimal point (*3.14*), ellipsis (...), or be used in web links (*www.mail.com*). Additionally, punctuation is not always consistently available. It is often missing in historical text and used inconsistently and irregularly in non-standard text from social media.

Our goal is to build a supervised model that is easily applicable to all kinds of text. If sentence segmentation is performed as an independent task, we can use binary tags to disambiguate whether a character is the end of a sentence and then model the problem as a sequence labelling task. Sentence segmentation is a very common preprocessing step for text analysis in NLP. It is usually the step prior to word segmentation. In this thesis, sentence segmentation and word segmentation are modelled jointly for efficiency and better accuracy.

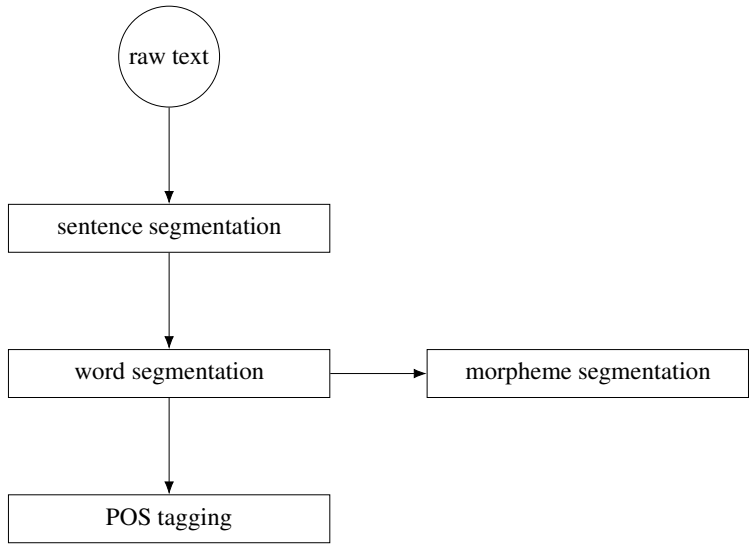

Figure 3.1. Segmentation and tagging of text in a pipeline.

3.2 Word Segmentation

A word is the smallest meaningful unit that can be used independently. It can be further defined from different aspects. For instance, a phonological word is a spoken signal that occurs commonly as part of a longer utterance and it is often subject to rhythm. Alternatively, orthographic words are often assumed when processing written text. For space-delimited languages like English, an orthographic word is usually a space-segmented token. For languages without space delimiters like Chinese, defining the notion of word and identifying the word forms accordingly is very challenging, even for a native speaker. There is little consensus in the Chinese NLP community on word segmentation standards as the word forms in Chinese are unnatural (Xia, 2000). As shown in Table 3.1, substantially different word segmentation standards for Chinese are adopted in practice by Microsoft Research Corpus (MSR) (Huang et al., 2006), Peking University (PKU) (Yu et al., 2003) and Penn Chinese Treebank (CTB) (Xue et al., 2005). Despite the variations, some word boundaries are shared across different annotation standards and the shared information can be exploited for multi-criteria word segmentation using transfer learning (Chen et al., 2017b). In Universal Dependencies (UD) (Nivre et al., 2016), words are defined as fundamental syntactic units that do not always coincide with phonological or orthographic words. Some orthographic tokens, known in UD as multi-word tokens, therefore need to be broken into smaller units that cannot always be obtained by splitting the input character sequence.

Given the input sentence as a string of characters, word segmentation is the task of separating out words from running text. Word segmentation is also referred to as word tokenisation (Jurafsky and Martin, 2000). Similar to sentence

Table 3.1. *Different word segmentation standards in MSR, PKU and CTB introduced in Gong et al. (2017).*

MSR	全国各地 (all over the country)				医学 (medical science)	界 (field)	专家 (expert)
PKU	全国 (all the country)		各地 (every where)		医学界 (medical science community)		专家 (expert)
CTB	全 (all)	国 (country)	各 (every)	地 (place)	医学界 (medical science community)		专家 (expert)

segmentation, for space-delimited languages, relying on white space alone is insufficient to obtain high-quality segmentation. At least punctuation must be properly separated from adjacent words. Moreover, word segmentation is very challenging for languages like Chinese, Japanese, Thai and Vietnamese. There is no word delimiter in these languages and the word boundaries are highly ambiguous. In Vietnamese, the space-delimited orthographical units are syllables, not entire words. Applying the delimited syllables as basic units for word boundary prediction, word segmentation for Vietnamese is essentially the same as for Chinese and Japanese. In addition, for some languages, the space-delimited units in the surface form are too coarse-grained and therefore often further analysed. This is the case for Arabic, Hebrew, Thai and Lao.

Word segmentation can be viewed as a character-level sequence labelling task (Xue, 2003; Chen et al., 2015b). For languages with different writing systems, characters are passed as basic input units into a sequence labelling model and a sequence of tags that are associated with word boundaries is predicted. For the most fundamental model, we employ four tags *B, I, E, S* to indicate a character positioned at the beginning (*B*), inside (*I*), or at the end (*E*) of a word, or occurring as a single-character word (S). Theoretically, binary tags are sufficient to indicate the word boundaries, but Zhao et al. (2006) show that a more-fined grained tag set leads to better accuracy as more explicit position information is incorporated. This is confirmed by our experiments in Paper VI. The basic tag set is applicable to Chinese and Japanese without further modification. For space-delimited languages, we add an additional boundary tag *X* to denote spaces that do not belong to any segmented word forms.

In the UD framework, syntactically defined words have unique POS tags and enter into syntactic relations with other words (Nivre et al., 2016). There is often a mismatch between orthographic words, called *tokens* in the UD terminology, and syntactic *words* for space-delimited languages. Typical examples are clitics, like Spanish *dámelo* = *da me lo* (1 token, 3 words), and contractions, like French *du* = *de le* (1 token, 2 words). Tokens that need to be split into multiple words are called *multi-word tokens* and can be further subdivided into those that can be handled by splitting the character sequence, like English *cannot* = *can not*, and those that require transduction, like French

Char.	On considère qu'environ 50 000 Allemands du Wartheland ont péri pendant la période.
Tags	BEXBIIIIIIIEXBIEBIIIIIEXBIIIIEXBIIIIIIIEXB$\overline{\text{BE}}$XBIIIIIIIIEXBIEXBIIEXBIIIIEXBEXBIIIIIES

Figure 3.2. Tags employed for word segmentation. In the UD framework, *50 000* is a multi-token word, while *qu'environ* and *du* are multi-word tokens that should be processed differently.

du = *de le*. We call the latter *non-segmental multi-word tokens*. In addition to multi-word tokens, the UD scheme also allows *multi-token words*, that is, words consisting of multiple tokens, such as numerical expressions like *20 000*.

We extend the basic tag set for word segmentation in UD. The boundary tag *X* disambiguate the spaces in multi-token words from regular ones. To identify the non-segmental multi-word tokens defined in UD, we introduce four additional tags $\overline{B}$, $\overline{I}$, $\overline{E}$, $\overline{S}$ that mark positions corresponding to *B*, *I*, *E*, *S* in multi-word tokens. Figure 3.2 shows the input characters and the corresponding tags to be predicted for word segmentation.

The identified non-segmental multi-word tokens are transduced in a context-free fashion, as 98.3% of them only have one possible transduction based on the statistics from the UD dataset. We can use a dictionary to map the multi-word tokens into their segmented components or train an encoder-decoder to transcribe them as a sequence-to-sequence problem, depending on the productivity of multi-word tokens in specific languages.

Word segmentation is extensively evaluated in Papers V and VI on the datasets from the shared tasks on cross-lingual word segmentation and morpheme segmentation (MLP 2017) (Liu and Liu, 2017) as well as the UD datasets. Segmentation accuracy varies very substantially across different datasets. The proposed model yields nearly perfect segmentation accuracies on most space-delimited languages with few or no non-segmental multi-word tokens, whereas it is notably more challenging to segment languages with no word delimiters, such as Chinese and Japanese, as well as languages with productive non-segmental multi-word tokens, like Arabic and Hebrew.

We propose six statistical factors that vary depending on language and writing system. They are referred to as typological factors although most of them are only indirectly related to the traditional notion of linguistic typology and depend more on writing system.

- **Character Set Size** is the number of unique characters, which is related to how informative the characters are to word segmentation. Each character contains relatively more information if the character set size is larger.
- **Lexicon Size** is the number of unique word forms in a dataset, which indicates how many unique word forms have to be identified by the segmentation system. Lexicon size increases as the dataset grows in size.

- **Average Word Length** is calculated by dividing the total character count by the word count. It is negatively correlated with the density of word boundaries. If the average word length is smaller, there are more word boundaries to predict.
- **Segmentation Frequency** denotes how likely it is that space-delimited units are further segmented. It is calculated by dividing the word count by the space-segment count. Languages like Chinese and Japanese have much higher segmentation frequencies than space-delimited languages.
- **Multi-word Token Portion** is the percentage of multi-word tokens that are non-segmental.
- **Multi-word Token Set Size** is the number of unique non-segmental multi-word tokens.

The last two factors are specific to the UD scheme but can have a significant impact on word segmentation accuracy.

The experimental results in Paper VI indicate that word segmentation accuracy is positively related to the presence of word boundary markers and negatively to the number of unique non-segmental multi-word tokens. For space-delimited languages with low segmentation frequencies, very high accuracies can be obtained even with very small training sets. Without space delimiters, word segmentation accuracy heavily depends on the training set size. For Chinese, the segmentation accuracy is much higher on UD than MLP 2017 when comparing the results reported in Papers V and VI.

With respect to the typological factors, all languages can be categorised into four groups. For each group, we can apply some specific settings to further improve the segmentation accuracy. The proposed settings are easily applicable to new languages beyond the explored datasets based on statistics of the typological features.

- For languages with word-internal spaces like Vietnamese, we first separate punctuation and then use the space-delimited syllables as the basic units for boundary prediction.
- For languages with large character sets and without space delimiters, such as Chinese and Japanese, we use concatenated 3-gram representations for the input characters.
- For languages with more than 200 unique non-segmental multi-word tokens, like Arabic and Hebrew, the encoder-decoder model is applied as the transducer in addition to the mapping dictionary derived from training data.
- For other languages, the universal model without any specific adaptation is sufficient.

The word segmentation system proposed in Paper VI is very competitive when compared to previous work (Straka and Straková, 2017), especially on

languages that are fundamentally more difficult to segment, such as Chinese, Japanese, Arabic and Hebrew.

Being a supervised model, the proposed word segmentation system performs reasonably well only if a sufficient amount of annotated data is available. As shown in Paper VI, even for space-delimited languages, the segmentation results are unsatisfactory if less than 20 sentences are available for training. We propose two transfer approaches to low-resource word segmentation in Papers VI and VII. The 2017 CoNLL shared task (Zeman et al., 2017) provides a set of low-resource languages, namely Buryat, Kurmanji, North Sami and Upper Sorbian, in which small annotated data samples are provided in addition to the test sets. We use these datasets for evaluation.

In Paper VI, considering that the segmentation model is fully character-based, we simply select the model of the language that shares the most characters with the surprise language as its segmentation model. No annotated data of the low-resource language are used for model selection. The evaluation results show that applying segmentation models trained on a different language with more training data yields better results than relying on the small annotated samples of the target language, indicating that we can process any space-delimited language with the existing models even if no annotated data are available, which underlines the universal character of the proposed word segmentation system.

In Paper VII, the small annotated datasets are utilised as development sets to select the segmentation models trained on large datasets. For a specific low-resource language, we apply all the models trained on languages with sufficient data and pick the model that achieves the highest evaluation score on the development set. Comparing the results in Papers VI and VII, model selection using the small annotated datasets in Paper VII outperforms the simple character overlap selection method in Paper VI.

To perform word segmentation jointly with sentence segmentation, two extra tags T and U are introduced to mark the characters at the end of a sentence, in addition to the boundary tags that are only associated with word segmentation. The tag T is employed if the character is a single-character word and U is used otherwise. When word segmentation and sentence segmentation are modelled jointly, context information beyond sentence boundaries is incorporated into the segmentation model, which potentially improves word segmentation accuracy.

The evaluation of joint sentence segmentation and word segmentation is presented in Paper VII on the UD datasets. For sentence segmentation, the accuracies across different datasets vary very substantially. In general, sentence segmentation is more difficult for non-standard text, in which punctuation is not consistently used, as well as for historical text in which punctuation is not available.

Char.:	elämä tuo kremppoja mukanaan .
Tags:	BIIIEXBESXBIIIIIESSXBIIIIEBEXS
Output:	elämä tu//o kremppo//j//a mukana//an .

Figure 3.3. Boundary tags for morpheme segmentation in Finnish.

3.3 Morpheme Segmentation

A morpheme is the smallest meaningful grammatical unit of a language. Unlike words, morphemes may not always appear by themselves. For morphologically rich languages such as Turkish, Finnish and Farsi, word forms are decomposed into stems, prefixes and suffixes. The morpheme segmentation tasks in this thesis do not involve any inflectional (fusional) analysis. Morpheme segmentation is similar to word segmentation for languages without space delimiters as there are no clear boundaries between the morphemes to be segmented. We therefore model morpheme segmentation as a character-level sequence labelling problem, similarly to word segmentation, and predict boundary tags that are associated with morphemes.

Morpheme segmentation is performed at the word level. To incorporate character information beyond word boundaries, we employ a sentence-level sequence labelling model in Paper V. Four positional tags *B, I, E, S* are employed to indicate a character that is located at the beginning (*B*), inside (*I*), or at the end (*E*) of a morpheme, or as a single-character morpheme (*S*). We use an extra tag *X* for word boundaries. Figure 3.3 exemplifies the input characters and boundary tags to be predicted for morpheme segmentation.

Morpheme segmentation is evaluated in Paper V on the MLP 2017 datasets on Basque, Farsi, Filipino, Finnish, Kazakh, Marathi and Uyghur. The segmentation accuracy heavily depends on the size of the training set. The proposed model obtains very high accuracy when sufficient amounts of annotated data are available. Unlike non-trivial word segmentation for languages like Chinese and Japanese, the types of prefixes and suffixes to be identified in morpheme segmentation are limited and less ambiguous, and an unsupervised approach is therefore a feasible option for languages with few or no annotated resources.

In addition, morpheme segmentation and word segmentation can be modelled jointly by modifying the boundary tag set so that word boundaries as well as morpheme boundaries within word forms are predicted simultaneously. This has not been studied in this thesis.

3.4 Joint Word Segmentation and POS Tagging

Word segmentation accuracy has a significant impact on POS tagging, as the segmentation errors propagate. At the same time, POS tags are very

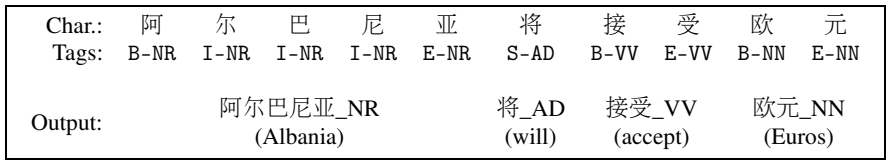

Figure 3.4. Combinatory tags for joint word segmentation and POS tagging.

informative for word segmentation for languages like Chinese and Japanese. Thus, better performance can be achieved by modelling word segmentation and POS tagging jointly (Ng and Low, 2004; Zhang and Clark, 2008).

To model joint word segmentation and POS tagging as a character-level sequence labelling task, we combine the boundary tags that are associated with word segmentation (*B, I, E, S*) with POS tags. As illustrated in Figure 3.4, based on the predicted combinatorial tags, both segmentation and POS tagging information can be retrieved.

Compared to pipeline models, joint word segmentation and POS tagging drastically enlarges the tag set and therefore leads to much slower tagging speed. Referring to Section 2.5, the time complexities of the neural CRF models for word segmentation and POS tagging as independent tasks are $O(i^2n)$ and $O(j^2n)$, where i and j are the number of segmentation tags and POS tags respectively. The complexity of the pipeline model is $O((i^2+j^2)n)$, whereas the joint model is $O((i \cdot j)^2n)$.

The tagging speed of the joint model can be improved if we reduce $i \cdot j$ by pruning the combinatorial tag set. For some POS tags, combining them with the full boundary tags is redundant. For instance, the only word that can be tagged as *DEG* in CTB is the function word 的. Since it is a single-character word, combinatorial tags *B-DEG*, *I-DEG*, and *E-DEG* never occur in the experimental data. They can therefore be pruned to reduce the search space.

Joint word segmentation and POS tagging for Chinese is extensively evaluated in Paper III. The proposed model is applied to datasets in different sizes, genres and annotation schemes. Our model is systematically compared with ZPar (Zhang and Clark, 2010), a linear statistical model based on structured perceptron learning with beam search, on the CTB and UD datasets. The experimental results show that, with sufficient training data, ZPar and the proposed neural CRF-based model perform equally well on standard news text. However, the experimental results indicate that our neural CRF model is substantially better at processing non-standard text as it yields significantly higher scores on texts from the sources of conversations, short messages and weblogs.

The tagging speed test in Paper III shows that the implemented tagger is very efficient with GPU devices and therefore can be applied to tagging very large amounts of text for practical purposes. The ensemble model averages

Table 3.2. *Output tags for different segmentation and tagging tasks.*

	B, I, E, S, X	$\overline{\text{B}}$, $\overline{\text{I}}$, $\overline{\text{E}}$, $\overline{\text{S}}$	T, U	B-ADJ, S-DET, ...
Joint word and sentence segmentation	+	+	+	
Word segmentation	+	+		
Morpheme segmentation	+			
Joint word segmentation and tagging				+

four models and it is more accurate when compared to the single model. It is only 30% slower than the single model in terms of tagging speed.

The joint word segmentation and POS tagging model is applicable to languages with large character set sizes and small average word length, such as Japanese and Vietnamese. It is difficult to generalise for space-delimited languages as the characters for those languages are usually not informative for POS tagging. Additionally, compared to Chinese, sentences in space-delimited languages are much longer on average measured in characters. Combining the POS tags with segmentation tags drastically enlarges the search space and the model therefore becomes extremely inefficient both for training and tagging.

Table 3.2 summarises the output tags that are applied for all the introduced segmentation and tagging tasks.

3.5 Named Entity Transliteration

The conversion of names between languages in different writing scripts is often interchangeably referred to as transliteration, transcription, or sometimes romanisation if the target language uses Roman (Latin) script (Halpern, 2007). Transliteration is formally defined in Li et al. (2009) as follows:

> Transliteration is the conversion of a given name in the source language (a text string in the source writing system or orthography) to a name in the target language (another text string in the target writing system or orthography), such that the target language name: (i) is phonemically equivalent to the source name; (ii) conforms to the phonology of the target language and (iii) matches the user intuition of the equivalent of the source language name in the target language.

For instance, given the English surname *Trump* as the source named entity, the transliteration can be 特朗普 *(pinyin: tè lǎng pǔ)* (Mainland China) or 川普 *(pinyin: chuān pǔ)* (Hong Kong and Taiwan) in Chinese, depending on the region where the term is used. The inconsistency of the transliteration standards is one of the main challenges for building accurate machine transliteration systems. Transliteration can also be very straightforward, as in the case of romanising Chinese named entities, in which the Chinese characters are converted into their Pinyin via a many-to-one mapping. In general, automatic

transliteration is a very effective way to process named entities as out-of-vocabulary words in machine translation (Durrani et al., 2014), cross-lingual information retrieval (Saravanan et al., 2013) and corpus alignment (Sajjad et al., 2011).

Machine transliteration can be modelled as a general sequence-to-sequence transduction problem. It can therefore be approached with models for machine translation. Unlike general machine translation, transduction between source strings and target strings obeys a strict linear order in transliteration. There is no reordering involved. We can use this characteristic either to constrain the general sequence transduction system, or to model transliteration as a segmentation and tagging problem.

In Paper I, we build a phrase-based system by modelling machine transliteration as statistical machine translation without distortion. Instead of using characters as the basic transliteration units, the M2M Aligner (Jiampojamarn et al., 2007) is applied to obtain more coarse-grained transliteration units in the form of letter groups. The alignment quality of the M2M Aligner has a significant impact on transliteration quality. For English-Chinese transliteration, we propose three different ways to improve the alignment quality:

- We contract some common letter combinations that are always pronounced jointly as single sounds, such as *ch, ck, sh* and two identical letters appearing next to each other.
- The letter *x* is occasionally pronounced as two sounds and can therefore be aligned to two Chinese characters, which often causes alignment mistakes. We post-process the alignments that are associated with *x* using some heuristic rules.
- We use an expectation-maximisation-based (EM) approach to reduce low frequent alignment terms. With greedy local search, we iteratively set new boundaries between two neighbouring segmented transliteration units.

The refined output of the M2M Aligner is combined with Moses (Koehn et al., 2007) to build the transliteration system. We employ IRSTLM (Federico et al., 2008) to build language models of order 6. The phrase-based model is capable of resolving some segmentation errors by utilising more coarse-grained phrases as transliteration units. Additionally, the high-order language model optimises the generated transliteration as a complete sequence. At decoding time, for English to Chinese transliteration, we use a CRF model to obtain the transliteration units on the English side, similarly to word segmentation. For Chinese to English, the input Chinese characters are applied as the transliteration units.

The proposed standard system, in which only the official training datasets are used, is very competitive on English-Chinese transliteration when com-

pared to the other systems evaluated in the NEWS shared task (Banchs et al., 2015) according to the evaluation in Paper I. The evaluation scores of English to Chinese transliteration are higher than vice versa. Chinese to English in the shared task is a backward transliteration task, because the Chinese names in the input are originally transliterated from English names to be predicted. Backward transliteration is more challenging in general, as some information from the original target names to be generated is lost and very difficult to recover from the source names.

Transliteration is pronunciation-based and therefore specific to language origin of the source names. The same written form may have various pronunciations in different languages and therefore we transliterate it differently. For instance, *Julia* is transliterated as 朱莉亚 *(pinyin: zhū lì yà)* and 尤莉亚 *(pinyin: yóu lì yà)* in Chinese, respectively as an English name and a German name. The English set from the shared task contains Western names from multiple language origins. Therefore, we build language specific systems using multilingual data to improve English-Chinese transliteration. In Paper I, the dictionary of *Chinese Transliteration of Foreign Personal Names* (Xia, 1993) is employed as an external resource for constructing language specific transliteration systems of Czech, English, Finnish, French, Turkish, German, Portuguese, Hungarian, Italian, Romanian, Russian, Spanish, Swedish and Serbian.

We apply all the language specific transliteration systems to the source names. A linear regression model is trained to rerank the outputs of all the systems by using the scores of Moses, such as total score, language model score, phrase score and different translation model scores, as features. For both English to Chinese and Chinese to English transliteration, the multilingual reranking models drastically outperform the standard baseline systems as shown in Paper I.

In Paper II, named entity transliteration is instead modelled as a segmentation and tagging problem. The high-quality segmentation and alignment of the transliteration units are obtained in the same way as in Paper I. We apply neural networks both for obtaining coarse-grained transliteration units, similarly to the CRF model in Paper I, and for mapping the transliteration units from source to target language.

The transliteration units are first passed through convolutional layers and pooling layers. A recurrent layer is added on top to capture the dependencies. Considering that transliteration is a linear procedure without any hierarchical structures involved, we employ basic RNNs instead of LSTMs or GRUs as the recurrent cells. The experimental results show that there is no significant difference between using basic RNNs and LSTMs while training basic RNNs is much faster. The output layer uses softmax as the activation function to map the outputs of the recurrent layer into the target representations. At decoding time, we use binary tags to indicate the boundaries of the transliteration units and train a regular RNN to segment the English source names.

The experimental results show that the neural network-based transliteration system outperforms the naive Moses system (Costa-jussà, 2016) that uses individual characters as basic transliteration units by a large margin. However, the phrase-based transliteration system proposed in Paper I outperforms the neural network model significantly. The neural network model proposed in Paper II is a pipeline system that relies heavily on the alignment quality of the M2M Aligner and the segmentation accuracy for the transliteration units.

3.6 Evaluation Metrics

To evaluate the aforementioned segmentation and tagging tasks, we introduce and analyse the standard evaluation metrics in this section. Precision and recall along with their evenly-weighted average F1-score are conventionally employed as the standard evaluation metrics for segmentation tasks. For sentence segmentation and word segmentation, we propose that precision should be excluded from the conventional evaluation metrics and using recall alone is more adequate. For machine transliteration, word-level accuracy (ACC) is applied in addition to F1-score for evaluation.

3.6.1 Segmentation and Tagging Metrics

Precision and recall were originally introduced as evaluation metrics for information retrieval (IR) (Kent et al., 1955). Given annotated references, precision (P) and recall (R) are calculated via normalising true positives (TP), the number of correctly segmented terms, respectively by prediction positives (PP), the total number of terms returned by the system, and real positives (RP), the total number of terms in the reference. PP is made up of TP and false positives (FP); RP is made up of TP and false negatives (FN). FP counts the incorrect terms returned by the system and FN counts the reference terms that are not covered by the system. The F1-score (F) is the evenly weighted harmonic mean of precision and recall.

$$P = \frac{TP}{PP} = \frac{TP}{TP+FP} \tag{3.1}$$

$$R = \frac{TP}{RP} = \frac{TP}{TP+FN} \tag{3.2}$$

$$F = 2 \cdot \frac{R \cdot P}{R+P} \tag{3.3}$$

For segmentation tasks, the evaluation scores are calculated with respect to the complete units to be segmented, such as sentences, words and morphemes, rather than individual characters or boundaries. Table 3.3 illustrates how to compute precision and recall for word segmentation.

In the evaluation setup for standard IR tasks, precision and recall are not directly related. It is trivial to optimise the system with respect to either precision or recall, but difficult to improve both. This differs from sentence segmentation and word segmentation, where we only insert boundaries between the characters: the characters are neither added nor deleted. Precision and recall are therefore correlated in sentence segmentation and word segmentation. If we modify a boundary that optimises recall, the precision will also improve. In sentence segmentation and word segmentation, 100% recall guarantees 100% precision, and it is non-trivial to optimise one without the other. Thus, it is not necessary to employ both precision and recall as evaluation metrics.

In addition, recall is hard constrained by the reference as RP is fixed, whereas precision is related to PP, which can be determined by the segmentation system. Precision gets larger when PP is smaller. Precision therefore favours under-splitting systems, and integrating precision into the evaluation metrics can be misleading.

The drawbacks of employing precision as the evaluation metric for word segmentation are analysed both theoretically and experimentally in Paper V. The experimental results on English and Chinese word segmentation clearly indicate that precision and recall are correlated and precision favours under-splitting systems. We therefore propose that precision should be excluded from conventional evaluation metrics. In contrast to precision, recall is hard-constrained by the reference and not biased towards either under-splitting or over-splitting systems. It explicitly measures the correctly segmented words that are meaningful to higher-level tasks. Employing recall solely is sufficient and more adequate as the evaluation metric for word segmentation. This conclusion applies to sentence segmentation as well. To compare with related work on word segmentation, we nevertheless report the segmentation accuracy in F1-score in Papers III, V, VI and VII.

For morpheme segmentation, if we measure the segmentation quality only with respect to the segmented prefixes and suffixes as by convention, it is still necessary to employ both precision and recall. Since the terms containing characters that are not associated with prefixes and suffixes are subtracted for the evaluation, precision and recall are not correlated.

Precision and recall can be adapted to evaluate joint word segmentation and POS tagging, in which case the term is considered as a TP only if it is a correctly segmented word and has correct POS tag. As illustrated in Table 3.3, the joint precision and recall underestimate segmentation accuracy as the correctly segmented words with incorrect POS tags do not contribute to the joint scores. Thus, precision and recall only for word segmentation are presented additionally for joint word segmentation and POS tagging tasks. Joint precision and recall are the overall evaluation for both tasks. Furthermore, the POS tagging accuracy for correctly segmented words is usually not reported,

Table 3.3. *Precision, recall and F1-score for word segmentation and joint word segmentation and POS tagging (CTB standard). The true positives in the candidates are marked in blue. For word segmentation, the false positives are in red. For joint word segmentation and POS tagging, the incorrectly segmented terms are marked in red. The correctly segmented terms with incorrect POS tags are in yellow.*

	Word Segmentation			Joint Seg. & POS Tagging			
	P	R	F	P	R	F	ACC_{POS}
夏/ 天/ 很/ 热	0.5	0.67	0.57	-	-	-	-
夏天很/ 热	0.5	0.33	0.4	-	-	-	-
夏_NT/ 天_NN/ 很_VD/ 热_VA	0.5	0.67	0.57	0.5	0.67	0.57	1
夏_NT/ 天_NN/ 很_VD/ 热_VV	0.5	0.67	0.57	0.25	0.33	0.28	0.5
夏天很_NN/ 热_VA	0.5	0.33	0.4	0.5	0.33	0.4	1
夏天很_NN/ 热_VD	0.5	0.33	0.4	0	0	0	0

Reference: 夏天_NT (summer) 很_VD (very) 热_VA (hot)

but can be obtained by:

$$ACC_{POS} = \frac{TP_{joint}}{TP_{seg}} = \frac{P_{joint}}{P_{seg}} = \frac{R_{joint}}{R_{seg}} \tag{3.4}$$

3.6.2 Transliteration Metrics

ACC and F1-score are the two standard evaluation metrics of the top transliterations generated by the system (Li et al., 2009). ACC (word accuracy in top-1) measures the percentage of the top generated candidates that exactly match one of the references. ACC indicates how well the system generates transliterations as complete entities.

F1-score measures how different the top transliteration candidate is to the closest reference. First, we calculate the length of the longest common subsequence (LCS) between a candidate and a reference as:

$$LCS(c,r) = \frac{1}{2}(|c| + |r| - ED(c,r)) \tag{3.5}$$

where ED is the edit distance and $|x|$ is the length of x. The closest reference r_m is the one that has the minimum edit distance to the candidate. F1-score (F) for transliteration is then calculated in a similar way to F1-score for word segmentation:

$$R = \frac{LCS(c,r_m)}{|r_m|} \tag{3.6}$$

$$P = \frac{LCS(c,r_m)}{|c|} \tag{3.7}$$

$$F = 2 \cdot \frac{R \cdot P}{R + P} \tag{3.8}$$

Table 3.4. *ACC and F1-score (F) for transliteration. Each of the evaluated systems only returns one candidate. The correctly transliterated characters are in blue. The incorrect ones are marked in red.*

	Closest Reference	ED	LCS	ACC	P	R	F
特朗普	特朗普	0	3	1	1	1	1
川普	川普	0	2	1	1	1	1
瑞姆	川普	2	0	0	0	0	0
特普	特朗普	1	2	0	1	0.67	0.8
特鲁普	特朗普	1	2.5	0	0.83	0.83	0.83
特鲁姆普	特朗普	2	2.5	0	0.63	0.83	0.71

Source: Trump
Target references: 川普 or 特朗普

Unlike ACC, F1-score measures the generalisation power of the transliteration system with respect to individual characters. Table 3.4 illustrates how ACC and F1-score are computed for machine transliteration. In addition to ACC and F1-score, Mean Reciprocal Rank (MRR) measures any candidates that are generated by the system instead of only focusing on the top ones. MRR is calculated as:

$$MRR = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{rank_i} \tag{3.9}$$

where N is the total number of instances to be evaluated and $rank_i$ refers to the rank position of the first correct transliteration in the candidate list of the i-th instance.

Apart from measuring named entity transliteration in Papers I and II, ACC and F1-scores are also employed to evaluate the attention-based encoder-decoder for non-segmental multi-word tokens in Paper VI.

3.7 Summary

We have applied the general sequence labelling framework to specific segmentation and tagging tasks by adopting different output tag sets in this chapter. Moreover, the evaluation metrics for segmentation related tasks as well as transliteration have been presented and discussed. In the next chapter, we will discuss related work on segmentation and tagging in addition to neural network models for related sequence labelling tasks.

4. Related Work

In the previous two chapters, we have described the neural CRF-based models used in our experiments and shown how they can be applied to different segmentation and tagging problems. In this chapter, we discuss related work on these tasks, covering both supervised and unsupervised approaches. In addition, we discuss work on character representations and on related tasks such as POS tagging and named entity recognition.

4.1 Sentence Segmentation

Kiss and Strunk (2006) propose a language-independent, unsupervised sentence segmentation model that is primarily based on detection and disambiguation of abbreviations. According to their paper, up to 30% of the ambiguities concerning sentence boundaries are induced by abbreviations. The authors therefore make three assumptions about abbreviation detection: (i) abbreviations can be defined as a very tight collocation consisting of a truncated word and a final period; (ii) abbreviations are usually short; (iii) abbreviations may contain word-internal periods. The system is able to detect abbreviations with an accuracy of 99.38% on news text across different languages with the proposed characteristics. With the identified abbreviations, sentence segmentation is carried out in two stages. Three labels *<A>, <E>, <S>* are employed in the initial annotation stage respectively for abbreviations, ellipses and potential sentence boundaries. At the token-based classification stage, several additional heuristic rules for initials and ordinal numbers are applied to refine the initial annotations and make final predictions. The system is evaluated on news text in eleven languages and achieves very high accuracies (99.31 mean F1-score). However, it is not suitable for non-standard text in which punctuation is not consistently used.

There are some additional sentence segmentation models that are applied jointly with word segmentation. They will be introduced in Section 4.2 below.

4.2 Word Segmentation

Both supervised and unsupervised approaches have been applied to word segmentation. Most of the previous models are designed for Asian languages without word delimiters, such as Chinese and Japanese. Table 4.1 gives an

Table 4.1. *Overview of related work on word segmentation. Models are characterised by their fundamental categories, namely HMM, CRF, SVM, Perceptron and neural networks (NN), as well as whether they are supervised (SUP), transition-based (TB) and fully character-based (CB).*

Model	SUP	TB	CB	HMM	CRF	SVM	Perceptron	NN	Language
Brent (1999)	–	–	–	–	–	–	–	–	English (speech)
Venkataraman (2001)	–	–	–	–	–	–	–	–	English (speech)
Goldwater et al. (2006)	–	–	–	–	–	–	–	–	English (speech)
Mochihashi et al. (2009)	–	–	–	–	–	–	–	–	Chinese, Japanese
Jurish and Würzner (2013)	+	–	+	+	–	–	–	–	Multilingual
Papageorgiou (1994)	+	–	+	+	–	–	–	–	Japanese
Tseng et al. (2005)	+	–	+	–	+	–	–	–	Chinese
Zhao et al. (2006)	+	–	+	–	+	–	–	–	Chinese
Nguyen et al. (2006)	+	–	+	–	+	+	–	–	Vietnamese
Wang et al. (2011)	+	–	+	–	+	–	–	–	Chinese
Zhang and Clark (2007)	+	–	–	–	–	–	+	–	Chinese
Chen et al. (2015a)	+	–	+	–	+	–	–	+	Chinese
Chen et al. (2015b)	+	–	+	–	+	–	–	+	Chinese
Wang and Xu (2017)	+	–	+	–	+	–	–	+	Chinese
Che et al. (2017)	+	–	+	–	+	–	–	+	Multilingual
Straka et al. (2016)	+	–	+	–	+	–	–	+	Multilingual
Paper VI in this thesis	+	–	+	–	+	–	–	+	Multilingual
Cai and Zhao (2016)	+	–	–	–	+	–	–	–	Chinese
Cai et al. (2017)	+	–	–	–	+	–	–	–	Chinese
Ma and Hinrichs (2015)	+	+	+	–	–	–	–	+	Chinese
Zhang et al. (2016)	+	+	–	–	–	–	–	+	Chinese
Zhang et al. (2017)	+	+	–	–	–	–	–	+	Chinese

overview of the related work on word segmentation that is discussed in this section. We can see that different approaches have been studied extensively in previous work.

The unsupervised models that have been proposed for word segmentation are mostly based on probabilistic models. The model-based dynamic programming system (MBDP) and the n-gram segmentation model (NGS) are two similar models introduced in Brent (1999) and Venkataraman (2001) respectively. They are both generative models. MBDP assumes that a text of segmented words is a single probabilistic event, which is generated incrementally as lexical types, phonetic representations of the lexical types, frequency sensitive tokens of each type and ordered tokens. NGS assumes that each word is generated independently via a standard n-gram model. Both models use approximate online search procedures to make local predictions for word boundaries.

Goldwater et al. (2006) propose two models that are based on the Dirichlet process (DP) (Teh, 2011), a distribution used in non-parametric Bayesian statistics, for word segmentation. The experimental results on the Bernstein-Ratner corpus (Bernstein-Ratner, 1987) show that the hierarchical Dirichlet process model (HDP) that incorporates bigram dependencies performs better than the naive unigram model. Moreover, HDP is also substantially better than the MBDP and NGS models.

The word segmentation model introduced by Mochihashi et al. (2009) is a nested hierarchical Pitman-Yor language model. The Pitman-Yor process

(Teh, 2006) is a generalisation of the Dirichlet process. The character n-gram Pitman-Yor spelling model is integrated into the word model. An efficient blocked Gibbs sampler that leverages dynamic programming for inference is also described. The model outperforms the previous models (MBDP, NGS and HDP) on the Bernstein-Ratner corpus. It is also evaluated on Chinese and Japanese word segmentation and obtains reasonable accuracies.

Unsupervised word segmentation models do not require any annotated data for training. They are not specific to a certain segmentation standard either. However, the segmentation accuracy achieved by an unsupervised model is usually substantially lower when compared to supervised models that are trained with sufficient amounts of annotated data.

Jurish and Würzner (2013) introduce an HMM-based framework for word segmentation and sentence segmentation. Four hidden states (*BOW, BOS, EOS, O*) are employed to denote the characters that are word-initial, sentence-initial, sentence-ending and characters that are not associated with any boundaries. An optional set of user-specified stopwords can be utilised as language-specific features. Additional statistical features are used to represent basic typographical class, letter-case, word length, and leading white space. The segmentation system is tested on datasets in five different languages. The experimental results show that the system substantially outperforms the unsupervised system introduced in Kiss and Strunk (2006).

Papageorgiou (1994) proposes a character-based HMM model with standard Viterbi decoding for Japanese word segmentation. The model exploits overlapping bigrams to predict whether a character is a word boundary.

Tseng et al. (2005) present a fully character-based CRF model for Chinese word segmentation. They use binary tags to indicate whether a character is a word boundary. There are three categories of linguistic features integrated into the model: character identity n-grams, morphological and character reduplication features.

Zhao et al. (2006) experiment with different boundary tag sets for character-based Chinese word segmentation in a CRF framework. The evaluation shows that employing more fine-grained tags, *B, I, E, S* (4-tag) for instance, significantly outperforms the binary tags in accuracy. The authors also provide a very simple criterion to select the best tag set based on the average word length in the training corpus.

Word segmentation models based on CRFs and support vector machines (SVM) (Cortes and Vapnik, 1995) are thoroughly compared on Vietnamese in Nguyen et al. (2006). The word segmentation models are trained on an annotated corpus composed of news text in various domains. The experimental results reveal that the SVM-based models perform slightly better than the CRF-based models using the same feature template in general. Most features they explore, such as *Syllable Conjunction (SC), Dictionaries (Dict)* and *External Resources (ERS)*, make positive contributions to both models,

except that *Vietnamese Syllable Detection (VSD)* is only beneficial to the CRF-based models.

Wang et al. (2011) utilise a large amount of auto-segmented data to exploit extra n-gram features that are applied jointly with lexicon features and additional information retrieved from Brown clustering (Martin et al., 1998). With some hyper-parameter tuning, the CRF-based approach achieves very high accuracies both for Chinese word segmentation and POS tagging on three CTB datasets (Xue et al., 2005) in spite of being a pipeline model.

In contrast to the character-based labelling approach for word segmentation, Zhang and Clark (2007) introduce a word-based model for Chinese using the discriminative perceptron learning algorithm (Collins, 2002). The feature templates contain both character-level and complete word-level information. Instead of using Viterbi decoding as for HMM and CRF-based models, a beam search decoder is employed. The experimental results indicate that the averaged perceptron performs well in general and reasonable segmentation accuracy can be achieved even with a relatively small beam size (16).

Apart from the linear HMM, CRF and SVM-based models that heavily rely on feature engineering, a number of neural network models for word segmentation are described in the literature as well. Most of them model word segmentation as sequence labelling by applying the neural network component as a feature extractor.

Chen et al. (2015a) apply the gated version of a recursive neural network (GRNN) (Socher et al., 2013) to Chinese word segmentation. A gated feed-forward neural network is used recursively to extract local features from a span of characters. The reset and update gates select and preserve the useful interactive information between neighbouring characters. Multiple GRNN layers are stacked to cover sufficient local context for prediction, which makes the neural network deeper and suffer from vanishing gradient while training. The authors use layer-wise training to update the weights of the stacked layers incrementally to mitigate this problem. With bigram features and pre-trained character embeddings, the proposed model outperforms the models of Zheng et al. (2013) and Pei et al. (2014).

Chen et al. (2015b) use unidirectional RNNs with LSTM cells to model Chinese word segmentation as character-level sequence labelling. The authors experiment with four different LSTM architectures. The evaluation shows that the most basic LSTM model (LSTM 1 in the paper) notably outperforms the other more complex models (for instance, the model with more RNN layers). Moreover, the impact of different dropout rates is also investigated in a set of parallel experiments. With smaller dropout rates, the model converges relatively faster but also overfits more to the training data. 0.2 is therefore chosen as the optimal dropout rate in the paper to balance between accuracy and efficiency. Additionally, character bigrams are integrated similarly as in Pei et al. (2014), but the obtained improvement is not equally large. Overall, the proposed model obtains similar performance to Chen et al. (2015a),

showing that utilising global features from RNNs does not lead to notable improvements.

A CNN-CRF model for Chinese word segmentation is presented by Wang and Xu (2017). Unlike RNNs computed in a temporal fashion, the forward pass of CNNs can be parallelised, which makes the CNN-based models much more efficient. In addition, by stacking multiple CNNs, the neural network is able to effectively extract and filter information from a very large context. The CNN-based model is in some ways similar to the GRNN model in Chen et al. (2015a). The major difference is that in a GRNN, the weights are shared across different layers, whereas the CNN layers are parametrised independently. The CNN-based network is also trained end-to-end, instead of applying layer-wise training. Moreover, the proposed model in Wang and Xu (2017) integrates word embeddings, rather than applying conventional n-gram features. The experimental results show that the CNN-CRF model performs equally well to the RNN-based models.

Che et al. (2017) design a character-based BiLSTM network for word segmentation, targeting languages without space delimiters, namely Chinese, Japanese and Vietnamese. The model incorporates rich statistical features gathered from large unlabelled corpora, such as pre-trained context-free character embeddings, bigram embeddings as well as the pointwise mutual information (Liang, 2005) (PMI) of consecutive characters. Che et al. (2017) also propose a cross-lingual sentence segmentation model, in which a BiLSTM is applied as the feature extractor at character-level. An additional BiLSTM at token-level is added on top. A feed-forward neural network then slides over the tokens incrementally as a binary classifier for sentence boundary detection.

Straka et al. (2016) propose a language-independent joint sentence segmentation and word segmentation system that is applicable in theory to all languages. The segmenter employs a BiRNN with GRU to predict if a character is the last one in a sentence, the last one in a token, or not the last one in a token. There is no CRF interface to incorporate transition information between the boundary tags. For splitting the multi-word tokens defined in UD, Straka et al. (2016) adopt automatically generated suffix rules in addition to dictionaries that are extracted from training data. Despite using a truly universal and language-independent framework, the hyper-parameters of the model, including training batch size, character embedding size, dropout rate and so on, are tuned specifically on individual datasets for optimal performance (Straka and Straková, 2017). The proposed model achieves near-perfect scores on most space-delimited languages. However, the segmentation accuracy is drastically lower on languages that are more difficult to process, such as Chinese, Japanese, Vietnamese, Arabic and Hebrew, when compared to the system introduced in Paper VI. The proposed system has components for POS tagging, morphological analysis and dependency parsing as well.

Unlike the aforementioned models that are based on character sequence labelling for the segmentation and tagging tasks, the Chinese word segmentation

model of Cai and Zhao (2016) does not make tagging decisions on individual characters, but directly scores different word candidates and searches for an optimal segmentation path with the highest score. The vector representation of a word candidate is computed via a gated combination neural network (GCNN) with the associated character embeddings. The word score is directly inferred by a linear transformation of the word representation. In addition, a link score is introduced to capture the interactions between different word candidates. The sentence score is then the sum of the word scores and link scores, which evaluates the segmentation score over the entire input sequence. The possible segmentations grow exponentially with sentence length, and the authors therefore limit maximum word length and use beam search for decoding. Reasonable performance can be obtained with a beam size of 4. The GCNN is shown to be effective for inducing word representations from character embeddings. The experimental results show that the proposed model outperforms the character-based sequence labelling models without bigram features. The model itself does not exploit any n-gram features and it is substantially worse than the sequence labelling models with bigram features. Overall, the model is still fully-character-based as the word representations are decomposed and represented by character embeddings.

Cai et al. (2017) refine the model of Cai and Zhao (2016) by employing a more efficient composition function for word representations and using a greedy decoder. The meaning of some Chinese loan words, for instance 沙发, *pinyin: shā fā* (sofa), is not compositional with respect to the associated characters. The authors therefore keep a list of embeddings for frequent words, which is applied in addition to a simple character composition model by average pooling. They also show that using a greedy decoder can lead to optimal performance by effectively incorporating global features. Moreover, different updating methods are investigated. Early update (Collins and Roark, 2004) works better than the standard update and LaSO update (Daumé III and Marcu, 2005). In early update, the weights are updated as soon as the reference segmentation falls out of the beam and becomes unreachable. The experimental results show that the proposed model is very efficient yet obtains better accuracy than the other top-performing models (Chen et al., 2015b; Cai and Zhao, 2016).

In addition to directly mapping the input elements to the output boundary tags or returning the candidate words as in the models introduced above, transition-based approaches are applicable to word segmentation as well, in which the tagging or segmenting process is converted into a sequence of transition actions that indirectly induce the equivalent tag sequence or the segmented words.

Ma and Hinrichs (2015) propose an embedding matching model for Chinese word segmentation. The input characters and the two character-based segmentation actions (*SEP* and *APP*) are represented as vectors. Both the context features of the input characters and the recent historical information

of the segmentation actions are extracted from a local window of the input sentence and then concatenated all together as the input for the matching neural network, which scores the corresponding actions and makes greedy local decisions. For the embedding matching model, employing pre-trained character embeddings does not lead to notable improvements for the proposed model. Overall, the model is very efficient, but obtains relatively lower scores when compared to the sequence labelling models with local bigram features as in Pei et al. (2014).

Zhang et al. (2016) apply a transition-based neural word segmentation system to Chinese. Similarly to the other models, two transition actions (*SEP* and *APP*) are employed for a stack-queue structure. Beam search is used for decoding. Early update is applied with max-margin training. The baseline word segmenter uses the same discrete feature template as in Zhang and Clark (2011). For the neural network based model, both character-level and word-level features are represented as vectors. RNNs with LSTM are used to encode the input characters with global information over the entire sentence. The experimental results show that the neural network model outperforms the baseline model with discrete features. The proposed model obtains even better results when the neural representations and the discrete features are combined. In addition, the authors also show that longer words are more difficult to segment correctly.

The model of Zhang et al. (2016) is extended into a joint informal word detection and segmentation system for Chinese Micro-text by Zhang et al. (2017). The *SEP* action in the baseline model is divided into two actions *SEP_f* and *SEP_if* respectively for formal and informal words. An extra neural network component is added to represent the associated states of *SEP_if*. A special symbol is used to denote the detected informal words. The training corpora are generated by randomly replacing the formal words with informal words. The evaluation shows that the proposed model is very effective for segmenting non-standard Chinese text with informal words.

4.3 Morpheme Segmentation

Creutz and Lagus (2007) describe Morfessor, a unified framework that consists of several models for morpheme segmentation. The general model aims at finding the optimal language model that segments the corpus into morphemes, which leads to a concise representation of the corpus. The maximum a posteriori estimate (MAP) is applied to estimate the parameters. There are several components in the model. The *Lexicon* stores the distinct morphemes and their properties. The *Grammar* describes how the morphemes are associated. Every word in the *Corpus* are represented as a sequence of some morphemes that are present in the *Lexicon*. Words are modelled with Hidden Markov Models (HMM) to incorporate transition information between morphemes.

The proposed models are evaluated on English and Finnish morpheme segmentation tasks and obtain around 0.7 F1-score with sufficient unlabelled data for unsupervised learning.

Ruokolainen et al. (2013) model supervised morpheme segmentation by predicting the morpheme boundaries at word-level using CRF-based approaches. The substrings on both sides of the pivot character are used as features. Similarly to Chinese word segmentation, the authors indicate that employing a more fine-grained tag set (4-tag) achieves better accuracy than binary tags for morpheme segmentation, which is consistent with Zhao et al. (2006). The proposed model is evaluated on Arabic and Hebrew. The experimental results show that the CRF-based model substantially outperforms the unsupervised approaches (Creutz and Lagus, 2007; Poon et al., 2009) even with very small amounts of annotated data for training.

Wang et al. (2016) apply several variants of LSTM to morpheme segmentation. Similarly to most previous work (Ruokolainen et al., 2013), the segmentation is performed as character sequence labelling at the word-level and no information beyond the word boundary is exploited. Unlike our model, no CRF layer is applied as output interface. According to the evaluation results, the BiLSTM model with n-gram features (BMW-LSTM in the paper) outperforms the CRF baseline on Hebrew with sufficient training data. However, it consistently underperforms the CRF model on the Arabic dataset.

4.4 Joint Word Segmentation and POS Tagging

For joint word segmentation and POS tagging, both word-based and character-based approaches have been proposed. In addition, the tasks are occasionally modelled jointly with dependency parsing, text normalisation and chunking. The related work on joint word segmentation and POS tagging is summarised in Table 4.2.

Based on the maximum entropy word segmenter originally introduced by Xue and Shen (2003), Ng and Low (2004) combine the binary boundary tags with the POS tags so that the model performs joint word segmentation and POS tagging as a character-level sequence labelling problem. The experimental results show that the character-based model has clear advantages over the word-based approach on POS tagging for Chinese. In addition, the joint model outperforms the pipeline model as it mitigates error propagation.

The unified system for lexical analysis proposed by Zhang et al. (2003) incorporates unknown word recognition, word segmentation and POS tagging for Chinese in a hierarchical hidden Markov model (HHMM) (Fine et al., 1998). The initial HMM pre-segments the character sequence into atomic units that cannot be split further. The following component performs simple and recursive unknown word recognition. Class-based word segmentation and

Table 4.2. *Overview of related work on joint word segmentation and POS tagging. Models are characterised by their fundamental categories, namely HMM, CRF, Maxent, MIRA, Perceptron and Neural Networks (NN), as well as whether they are transition-based (TB) and fully character-based (CB). All the included models are supervised and they are only applied to Chinese.*

Model	TB	CB	HMM	CRF	Maxent	MIRA	Perceptron	NN
Ng and Low (2004)	–	+	–	–	+	–	–	–
Zhang et al. (2003)	–	–	+	–	–	–	–	–
Zhang and Clark (2007)	–	–	–	–	–	–	+	–
Jiang et al. (2008)	–	–	–	+	–	–	–	–
Kruengkrai et al. (2009)	–	–	–	–		+	–	–
Zheng et al. (2013)	–	+	–	+	–	–	–	+
Pei et al. (2014)	–	+	–	+	–	–	–	+
Chen et al. (2017a)	–	+	–	+	–	–	–	+
Paper III in this thesis	–	+	–	+	–	–	–	+
Hatori et al. (2012)	+	+	–	–	–	–	–	–
Qian et al. (2015)	+	–	–	–	–	–	+	–
Lyu et al. (2016)	+	–	–	–	–	–	+	–

POS tagging are applied afterwards. The HHMM-based system obtains high segmentation accuracy on news text. The POS tagging accuracy is relatively lower when compared to character-based systems.

The word segmentation model of Zhang and Clark (2007) is extended into a joint word segmentation and POS tagging model introduced in Zhang and Clark (2008). Extra features associated with POS tagging are added to the original feature template for word segmentation. A multiple beam search method with respect to each input character is applied for decoding. The perceptron-based system outperforms the joint segmentation and POS tagging system introduced in Ng and Low (2004). Moreover, Zhang and Clark (2010) effectively exploit partial word information for standard beam search decoding, which makes the joint segmentation and POS tagging system of Zhang and Clark (2008) 10 times faster.

Jiang et al. (2008) propose a dual-layer CRF-based model for joint word segmentation and POS tagging for Chinese. The two CRF models are trained independently for the two subtasks respectively. At decoding time, the CRF for segmentation preserves a candidate list instead of only the top segmentation with the highest probability in a regular pipeline model. The joint probability model incorporates the segmentation information with the POS tagging scores and finds the joint best output. The proposed joint model is better than the pipeline models according to the evaluation result. It is also more efficient than the joint models built with combinatorial tags.

Kruengkrai et al. (2009) present a word-character hybrid model for joint word segmentation and POS tagging for Chinese. The system is trained with the Margin Infused Relaxed Algorithm (MIRA) (Crammer and Singer, 2003). The search space is represented as a lattice that consists of word-level and character-level nodes. The known words are inferred by the word-level nodes

while the unknown words are covered by the character-level nodes. An error-driven policy is used to select the correct path in the lattice for training. It performs slightly better than the frequency-based criteria according to the evaluation result. The proposed system yields results similar to Zhang and Clark (2008).

Zheng et al. (2013) apply a window-based feed-forward neural network with a CRF interface to joint word segmentation and POS tagging for Chinese. Joint segmentation and POS tagging is modelled as character-level tagging by combining the word boundary tags (4-tag) with the POS tags. The input characters are mapped into context-free character embeddings that are pre-trained from large corpora. The feature representation is formed by concatenating the embeddings within a local context window. A simple feed-forward neural network with only one hidden layer is applied for non-linear transformation of the embeddings. The output is passed to the CRF tagging interface. Despite using a relatively simple neural network with only local features, the system is still on par with the best performing CRF-based models. In addition, the neural network model has substantially fewer parameters as it uses dense vectors as feature representations, which drastically reduces the computational cost. The experimental results also show that the performance of the feed-forward neural CRF smoothly drops if we enlarge the size of the hidden layer and the feature window as the neural network overfits to the training data if too many parameters are introduced.

Pei et al. (2014) extend the model of Zheng et al. (2013) by adding POS tag embeddings and bigram features. The POS tag predicted at the last time step is represented as a vector and concatenated with the character embeddings representing the local context window. Additionally, the character bigram embeddings are initialised by averaging the embeddings of the two associated characters and integrated as extra input information. The authors also apply a max-margin criterion for training by incorporating an l_2 norm term into the loss function. The evaluation shows that the bigram features boost the system very substantially and make the system as competitive as the other best performing systems that utilise rich external resources (Zhang et al., 2013).

Chen et al. (2017a) propose a hybrid feature-enriched neural network model for joint segmentation and POS tagging for Chinese. CNNs with filters of different widths are applied locally to capture n-gram features of different orders. A high-way layer (Srivastava et al., 2015) is added to mitigate gradient vanishing so that the neural network can go deeper. A BiLSTM layer is applied to capture long-term dependencies. According to the evaluation, the proposed architecture does not obtain state-of-the-art accuracy on joint segmentation and POS tagging, in spite of being more complicated than the regular CNN and RNN-based models.

Hatori et al. (2012) introduce a character-based transition model for incremental joint word segmentation, POS tagging and dependency parsing. A stack is used to keep the intermediate states of the processed characters. The

entire analysis terminates when all the input characters are processed and the stack is empty. There are three actions: *APP* appends the first character in the queue to the word on top of the stack; *SH(t)* shifts the first character in the input queue as a new word to the *stack* and assigns a POS tag; *RL/RR* reduce the top two trees on the stack and form a subtree by adding left/right arcs. In addition, an indexing scheme based on the number of character-based arcs is proposed to more effectively track the transition process. With a rich feature template, the proposed joint model achieves substantially better results than the pipeline models on all the three subtasks.

Qian et al. (2015) employ a similar stack-queue structured transition-based system for joint word segmentation, POS tagging and normalization. The baseline system is similar to the joint segmentation and POS tagging model in Zhang and Clark (2007), but defined as a transition-based model to integrate joint normalisation and POS tagging. *APP* and *SEP* are the two actions that are associated with word segmentation and POS tagging, which determines whether a character is simply shifted from the input queue to the stack as a partial word or completes the partial word and obtains a POS tag. An extra action *SEPS* is introduced for normalisation, which indicates that the completed word is informal and should be substituted by its regular form. The experimental results show that the joint model is effective as it obtains additional improvements over the pipeline models. It can be further improved by adding a language model for normalisation.

Word segmentation, POS tagging and syntactic chunking is modelled jointly in a transition-based system in Lyu et al. (2016). The two transition actions that are associated with segmentation and tagging are similar to Qian et al. (2015). In addition to the standard stack-queue structure, there is an extra stack for chunking. Three more actions are added to perform joint chunking on top of the segmented words. *SEP(Type)* and *APP(Chunk)* for word operations correspond to *SEP(Tag)* and *APP(Word)* for character manipulation. The *FINISH* action finalises the entire process. The proposed joint model obtains similar improvements over the pipeline models as in Qian et al. (2015).

4.5 Named Entity Transliteration

Li et al. (2004) employ an HMM-based joint source-channel model for machine transliteration. The segmentation and alignment of the transliteration units are obtained by an EM algorithm. The model performs direct orthographical mapping between two different languages without integrating any intermediate phonemic units. The decoding model jointly optimises segmentation and mapping. The proposed model greatly reduces transliteration errors when compared to the systems that use intermediate representations on English-Chinese transliteration tasks.

Yang et al. (2009) apply a two-step CRF model to machine transliteration in a pipeline fashion. The first CRF segments the input string into transliteration units and the second CRF maps the segmented units to the target language. The pipeline CRF model obtains similar accuracy to the n-gram joint source-channel model (Li et al., 2004) on English-Chinese and English-Japanese transliteration tasks. The authors also show that further improvement can be obtained by interpolating the CRF-based and the HMM-based models.

Kunchukuttan et al. (2017) introduce a two-stage iterative bootstrapping system for unsupervised machine transliteration. The model incorporates two sources of information: cross-lingual phonetic features and contextual mapping information between the transliteration units. The phonetic correspondence is obtained via an EM-MAP (Logothetis and Krishnamurthy, 1999) approach as prior distributions. A discriminative log-linear model is trained for the mapping between transliteration units. The proposed model is evaluated on transliteration tasks between several Indian languages. It is substantially worse than the supervised models, but outperforms a rule-based baseline and some earlier unsupervised approaches (Ravi and Knight, 2009).

4.6 Character Representations

There are a number of character embedding approaches that incorporate orthographical features and word embedding models that utilise sub-word information. Sun et al. (2014) extend the word embedding (C&W) model introduced in Collobert and Weston (2008) and Collobert et al. (2011) for Chinese characters by incorporating information about radicals. The training objective of the proposed model contains two parts: (i) for a given n-gram, the model discriminates the correct pivot character from the randomly replaced character as in the original C&W model; (ii) in addition, it predicts the radical of the pivot character with the n-grams as its context. The trained Chinese character embeddings are evaluated on a Chinese character similarity judgement task and Chinese word segmentation. The radical-enhanced Chinese character embeddings outperform the conventional embeddings that are trained only with respect to the context according to the reported evaluation scores.

Wieting et al. (2016) introduce a character n-gram model to compose character sequences. The n-gram vectors of different orders are first summed and then passed to a non-linear function. The proposed composing technique is compared with character-level LSTMs and CNNs respectively. Despite its simplicity, the proposed approach is effective for both word similarity judgement and POS tagging.

Bojanowski et al. (2017) propose a sub-word approach to word embedding. The skip-gram model (Mikolov et al., 2013) is extended by utilising the bag of n-grams so that the internal structure of words is captured in the vector

Table 4.3. *Overview of related work on POS tagging and NER. Models are characterised by their fundamental categories, namely HMM, CRF and Neural Networks (NN), as well as whether they are supervised (SUP) and transition-based (TB).*

Model	SUP	TB	HMM	CRF	NN	POS	NER	Language
Merialdo (1994)	–	–	+	–	–	+	–	English
Smith and Eisner (2005)	–	–	–	+	–	+	–	English
Goldwater and Griffiths (2007)	–	–	–	+	–	+	–	English
Van Gael et al. (2009)	–	–	–	+	–	+	–	English
Brants (2000)	+	–	+	–	–	+	–	Multilingual
Collobert et al. (2011)	+	–	–	+	+	+	+	English
Søgaard and Goldberg (2016)	+	–	–	–	+	+	–	English
Plank et al. (2016)	+	–	–	–	+	+	–	Multilingual
Huang et al. (2015)	+	–	–	+	+	+	+	English
Ma and Hovy (2016)	+	–	–	+	+	+	+	English
Ma and Sun (2016)	+	–	–	+	+	+	–	Chinese
Misawa et al. (2017)	+	–	–	+	+	–	+	Japanese
Peng and Dredze (2016)	+	–	–	+	+	–	+	Chinese
Yang et al. (2017)	+	–	–	+	+	+	+	Multilingual
Dong et al. (2016)	+	–	–	+	+	–	+	Chinese
Lample et al. (2016)	+	+	–	–	+	+	+	Multilingual

representations, which is crucial for morphologically rich languages. Instead of incorporating n-gram vectors of different orders as in Wieting et al. (2016), the word embeddings are obtained by simply summing the associated n-gram vectors of a specific order. The empirical results on human similarity judgement and word analogy tasks show that the proposed sub-word embedding model is effective, especially for low frequent words and morphologically rich languages. It is also more effective than the word-based models when the training dataset is small.

4.7 POS Tagging and Named Entity Recognition

Most of the sequence labelling models that have been applied to POS tagging in general (not modelled jointly with word segmentation) and NER are applicable to segmentation and tagging as well. Table 4.3 summarises the POS tagging and NER models introduced in this section.

Unsupervised POS tagging is typically modelled with different variants of HMM using maximum-likelihood estimation (MLE) for parameter estimation. Merialdo (1994) uses MLE to train a trigram HMM for POS tagging. Smith and Eisner (2005) propose a discriminative log-linear model (CRF) instead of HMM using contrastive estimation. A fully Bayesian approach is introduced in Goldwater and Griffiths (2007). The Bayesian approach integrates all possible parameter values rather than estimating a single set of parameters, which leads to substantial improvement over the MLE-based approach and yields similar tagging accuracy to Smith and Eisner (2005). Van Gael et al. (2009) propose another Bayesian-based non-parametric version of HMM, known as the infinite HMM (iHMM). The model accommodates an arbitrary

number of hidden states. Apart from POS tagging, the iHMM model is also applied to shallow parsing.

The TnT POS tagger (Brants, 2000) is based on second-order HMM models. Unigram, bigram and trigram features are used for estimating the transition probabilities. Linear interpolation is applied for smoothing. In addition, suffixes are utilised for handling unknown words. The evaluation shows that the tagging accuracy of the proposed HMM-based model is comparable to the Maximum Entropy models.

Collobert et al. (2011) present a unified window-based feed-forward neural network model and apply it to a series of different yet correlated sequence labelling tasks, namely POS tagging, chunking, named entity recognition (NER) and semantic role labelling. The pre-trained word embeddings are obtained as the by-product of training a neural language model with massive amounts of plain text. Employing the pre-trained embeddings leads to substantial improvement on all the tasks. Maximising sentence-level likelihood, which incorporates the transition information between different labels, also achieves better performance than only using word-level loss for local greedy prediction. In addition, multi-task learning is applied to exploit the correlations between different tasks by sharing the embeddings and the hidden layer, which leads to further improvements. With some additional features and ensemble decoding, the proposed model obtains equal or better performance on the benchmark tasks when compared to statistical models with heavy feature engineering.

Søgaard and Goldberg (2016) experiment with deep BiRNNs in a multi-task learning setup with three subtasks: POS tagging, chunking and combinatory categorial grammar (CCG) parsing. POS tagging is considered as a lower-level task and therefore represented by the RNN layer closest to the input layer. The states of the inner RNN layer for POS tagging are passed to the outermost layers for chunking or CCG parsing. The proposed multi-task learning approach obtains substantial improvements over the baselines using regular single-task learning. It is also more effective when compared to the multi-task learning approach in Collobert et al. (2011). Based on some additional experiments that are not presented in the paper, the authors also point out that multi-task learning is beneficial only when the associated tasks are sufficiently similar.

Plank et al. (2016) employ BiLSTMs for multi-lingual POS tagging. In addition to the standard BiRNN, a hierarchical BiLSTM architecture (Ling et al., 2015) is applied to extract character-level information. An auxiliary word frequency loss is added to the regular label cross-entropy loss. The proposed model is extensively evaluated on 22 languages and performs particularly well for morphologically rich languages. Integrating the frequency based auxiliary loss is beneficial to OOV words for some languages, but does not lead to notable improvements in general.

Huang et al. (2015) apply BiLSTM-CRF to three sequence labelling tasks: POS tagging, chunking and NER. In addition to the conventional pre-trained

word embeddings, a list of hand-crafted spelling features that are associated with capitalisation, prefixes and suffixes as well as punctuation are integrated. Word-level bigrams and trigrams are used as contextual features as well. The experimental results indicate that it is effective to combine BiRNNs with a CRF layer for sequence labelling. The extra spelling features and context features are also beneficial. The proposed model outperforms the feed-forward neural CRF model in Collobert et al. (2011) on all three tasks.

A similar model named CNN-BiLSTM-CRF is presented by Ma and Hovy (2016). Instead of using hand-crafted features, the authors exploit character-level information from scratch by applying CNNs over the character sequence, which makes the framework truly end-to-end. The proposed sequence labelling model is evaluated on POS tagging and NER respectively. By using the GloVe embeddings as well as fine-tuning the hyper-parameters, higher evaluation scores are obtained when compared to the model of Huang et al. (2015).

Ma and Sun (2016) use edge embeddings for neighbouring characters/words in addition to the regular character/word representations in the BiRNN-CRF framework for noun phrase chunking, shallow parsing (generic chunking), POS tagging and Chinese word segmentation for social media. The edge embeddings can be obtained by concatenating the associated character representations, using the associated bigrams or computed by a feed-forward neural network. The transition information between the edges is also incorporated. The proposed model with non-linear edge representation improves the regular BiRNN-CRF by a small margin on the tasks at hand.

Lample et al. (2016) propose a transition-based chunking model, similarly to transition-based dependency parsing (Nivre, 2008; Dyer et al., 2015), for NER. The model uses an output-stack-buffer structure to incrementally construct complete named entities, instead of combining the NER tags with boundary tags and modelling NER as word-level sequence labelling. There are three transition actions: *SHIFT* transfers an input word from the buffer to the stack as a partial named entity to be completed; *REDUCE* forms a complete named entity by merging all the words in the stack and assigning an NER tag; *OUT* directly passes the words that are not associated with any named entities. BiLSTMs are employed to encode the input words. Character-level information is also incorporated by using hierarchical BiLSTMs. The states of the stack and the output queue are represented as Stack-LSTMs in the same way as in Dyer et al. (2015). In addition to the transition-based model, a BiRNN-CRF model similar to the model introduced in Ma and Hovy (2016) is also proposed. The only difference is that hierarchical BiLSTMs are used to extract character-level information instead of CNNs. The evaluation shows that the accuracy of the transition-based model is comparable to the BiRNN-CRF model on English NER but relatively worse on the German and Dutch datasets. As a general sequence transition framework, the proposed model

is also applicable to the other joint segmentation and tagging tasks, such as chunking and joint word segmentation and POS tagging for Chinese.

Misawa et al. (2017) extensively evaluate different variants of the BiRNN-CRF model on Japanese NER. Unlike the results of Ma and Hovy (2016) for English NER, the experimental results show that the CNN-based approach for character-level features is detrimental to Japanese NER whereas the RNN-based approach achieves improvements, because Japanese words are shorter on average compared to space-delimited languages and the interaction between the characters is difficult to capture by applying CNNs. In addition, pre-trained embeddings are shown to be very beneficial, which is consistent with the evaluation results on English, German and Dutch NER tasks (Lample et al., 2016; Ma and Hovy, 2016). The BiRNN-CRF-based model is nonetheless effective for Japanese NER as it drastically outperforms the linear CRF model.

Peng and Dredze (2016) adapt the BiRNN-CRF model to NER for Chinese social media with the feature representations derived from word segmentation. NER is considered as the target task in a multi-task learning scenario. The LSTM parameters are shared for both word segmentation and NER. They are first trained for word segmentation, and then fine-tuned in the NER task. The experimental results show that integrating word segmentation representations leads to notable improvements for NER. The proposed framework is later extended for multi-task domain adaptation (Peng and Dredze, 2017).

Three different transfer learning approaches for BiRNN-CRF-based sequence labelling frameworks are tested in Yang et al. (2017) on POS tagging and NER. Depending on the similarities between the source and target tasks, different components of the BiRNN-CRF model are shared. For domain adaptation of the same task on the same language (cross-domain transfer), character representations, word embeddings, RNN representations as well as CRF weights are all shared. For different tasks on the same language (cross-application transfer), all the components except the CRF interface are shared. For different languages on the same task (cross-lingual transfer), only character representations are shared. According to the evaluation results, the transfer learning approach achieves drastic improvements under multiple low-resource settings. For the standard benchmark tasks with rich training data, the improvements obtained by transfer learning are marginal.

Dong et al. (2016) apply BiRNN-CRF to Chinese NER. Different variants of LSTMs are evaluated. The LSTM cell with no peephole connection, no forget gate and coupled input gate yields the highest evaluation score for NER. In addition, radical information is integrated. The experimental results show that the radical feature contributes only if pre-trained character embeddings are not employed.

4.8 General Neural CRF Models in Other Tasks

There are some general neural CRF models and RNNs that are proposed for some other sequence processing tasks, such as optical character recognition, automatic speech recognition and handwriting recognition.

Peng et al. (2009) propose a simple neural CRF model. A regular feed-forward neural network with one hidden layer is employed as the feature extractor for the linear chain CRF interface. The L-BFGS algorithm (Marsden et al., 1993) is applied to optimise the parameters. Since the neural network model is relatively simple, the optimal hyper-parameters are obtained by iteratively updating the model parameters and hyper-parameters in a two-step procedure. The neural CRF is evaluated on protein secondary structure prediction and handwriting recognition tasks. It outperforms the linear chain CRF and regular feed-forward neural networks on both tasks.

A more general definition of a neural CRF is given in Do and Artières (2010). The neural network part is initialised layer by layer in an unsupervised manner using restricted Boltzmann machines (RBMs) (Hinton et al., 2006). A quadratic regularisation term is added to the training objective to mitigate overfitting. The feed-forward neural CRF model is evaluated on optical character recognition and automatic speech recognition tasks and shows its effectiveness in solving these tasks.

Graves (2012) applies RNNs to different types of sequence labelling tasks categorised in a taxonomy as sequence classification, segment classification and temporal classification. Temporal classification is the most general form, in which the fundamental units for labelling are not segmented. Segment classification is equivalent to sequence labelling in the common sense, where the boundaries of input units are clear and we only assign the most probable categorical labels over all input units, without considering segmentation. POS tagging is a typical segment classification task. However, a number of NLP tasks such as speech recognition, machine transliteration, joint word segmentation and POS tagging and NER, actually belong to the temporal classification category, where different forms of segmentation are required. In contrast to temporal classification, sequence classification is the most restrictive case, in which the label sequences are constrained to be length one. We assign single labels to the whole sequences and no segmentation is involved. Sequence classification applies to various text classification tasks in NLP. In addition, the gradient descent approach for training different types of RNNs, including LSTMs, is proposed in Graves (2012). Different variants of RNNs are tested on phoneme recognition and online/offline handwriting recognition tasks.

4.9 Summary

In this chapter, we have reviewed related work on all the tasks considered in the thesis, including both supervised and unsupervised approaches. In

addition, we have discussed models that are related to our main BiRNN-CRF architecture more generally. In the next chapter, we will conclude the thesis by returning to the research questions stated in the introduction and outlining directions for future research.

5. Conclusions

In this chapter, we summarise and give the main conclusions of the thesis by addressing the research questions in Chapter 1. In addition, research orientations for future work are discussed.

5.1 Main Conclusions

All the segmentation and tagging tasks for NLP that we study in this thesis are modelled as character-level sequence labelling. The general BiRNN-CRF framework is modified so that it is applicable to multiple tasks and languages with minimal task and language-specific adaptation. In addition, named entity transliteration is also modelled as a segmentation and tagging task, instead of as general sequence-to-sequence transduction. All the tasks are extensively evaluated on annotated corpora from multiple sources.

Applying a general framework to text segmentation and tagging
The BiRNN-CRF as a general sequence labelling model has been successfully applied to various tasks, such as NER, chunking and POS tagging. The experimental results in this thesis indicate that it is a very expressive sequence labelling framework and capable of effectively modelling different types of segmentation and tagging tasks.

In contrast to regular deep neural networks that make independent predictions, the CRF interface of BiRNN-CRF optimises the predicted tags over the entire sequence by incorporating transition information. Additionally, unlike the linear chain CRF that only utilises local features and heavily relies on feature engineering, the BiRNN component effectively extracts global features from the input character embeddings. The BiRNN-CRF model combines the advantages of both regular deep neural networks and linear CRFs, and it is therefore very suitable for structured prediction for sequences. It also drastically reduces the need for feature engineering, which makes the model very flexible and adaptive.

We apply the BiRNN-CRF model as a character-level sequence labelling model to different segmentation and tagging tasks. It is associated with a number of hyper-parameters, such as embedding sizes, RNN state size, RNN layer number, learning rate and dropout rate. Most of them have a significant impact on performance. We use a single set of core hyper-parameters regardless of the task, language and dataset variances. The evaluation results show

that the accuracies achieved by the system in all conditions with the adopted hyper-parameters are on par with state-of-the-art systems.

The adopted hyper-parameters may still be suboptimal with respect to individual tasks, languages or datasets, so additional improvements can be achieved by specific fine-tuning. However, fine-tuning the hyper-parameters to a particular dataset makes the model less adaptive. Additional development (tuning) sets are often required to avoid overfitting in this case (Andor et al., 2016). Besides, large-scale hyper-parameter tuning also requires very rich computational resources which may not be available in practice.

Representing characters in the neural network models

Apart from hyper-parameter tuning, another way of enhancing the neural network models is to improve the representations of the input characters. We explored several approaches to effectively represent the input characters as vectors. The character set size of most languages is relatively small. The correlations and similarities of the characters are usually captured directly by the neural network in specific contexts, as they are difficult to encode in the unigram character embeddings. For specific categories of characters like digits and punctuation, it is nonetheless helpful to initialise them with similar vector representations. In this thesis, we randomly initialise all the character embeddings for the experiments on sentence segmentation and word/morpheme segmentations. The character embeddings are fine-tuned while training the entire neural network by back-propagating the gradients. For languages with small character sets, most of the characters occur frequently and their representations can be tuned well.

The character embeddings are passed to the BiRNN layers to capture contextual information. However, the experimental results in Papers III and VI show that not all the information can be effectively integrated simply by using BiRNN for Chinese and Japanese. Applying concatenated n-grams encodes local information more directly and leads to notable improvements, but it is not beneficial for most space-delimited languages. The concatenated n-grams drastically enlarge the embedding space and increase the number of parameters to be trained.

In contrast to the alphabetic languages, Chinese and Japanese have very large character sets. Most characters have semantic meaning out of context. Thus, it is beneficial to employ pre-trained character embeddings obtained from large amounts of plain text for initialisation. The experimental results indicate that pre-trained character embeddings give additional improvements by capturing semantic similarities. The improvements are more substantial if the training set is small and concatenated n-grams are not used.

The applied embedding models for characters do not integrate any orthographical information. We therefore utilise radicals and graphical features as sub-character information for Chinese characters, considering that rich infor-

mation is encoded in the graphical components. The improvement induced by the orthographical features is however marginal, as shown in Paper III.

Adapting the general BiRNN-CRF model to different tasks and languages
Overall, the input character representation is task-independent but not completely language-independent. For languages with rich character sets, such as Chinese and Japanese, initialising the character representations with pre-trained character embeddings achieves better performance than the randomly initialised embeddings. In addition, applying concatenated n-grams leads to substantial improvements for some languages (Chinese and Japanese), but is not helpful or even detrimental for some others (Spanish and Catalan).

The selection of output labels is task-specific but relatively straightforward. For all the segmentation tasks, the labels are associated with the boundaries of the fundamental units to be identified. For word segmentation, the more fine-grained *B*, *I*, *E*, *S* tag set is usually employed instead of the basic binary tag to integrate more positional information for higher segmentation accuracy. However, binary tags are more efficient and favoured if tagging speed is crucial and computational resources are limited. To make the basic word segmenter universal and applicable to any language, a few extra tags namely *X,* $\overline{B}$*,* $\overline{I}$*,* $\overline{E}$*,* $\overline{S}$, are added for word segmentation schemes that require an additional transduction step. Furthermore, two more tags (*U, T*) can be added for joint sentence segmentation modelling. For joint word segmentation and tagging, the boundary labels are combined with the POS tags. The combinatorial tags can be pruned to increase tagging efficiency by reducing the search space. Overall, the tagging framework becomes more expressive and powerful but less efficient when the label set increases in size.

The joint word segmentation and POS tagging system in Paper III is primarily developed for Chinese, but it is then generalised to a multilingual universal system in Papers V, VI and VII, where we evaluate sentence segmentation and word/morpheme segmentation in a highly multilingual setup by applying the model to more than 70 datasets from UD and MLP 2017. We obtain state-of-the-art accuracies on most of the datasets.

Sentence segmentation is a challenging task in general. Despite being the step prior to word segmentation, we can nonetheless identify meaningful words from the incorrectly segmented sentences, which is different from word segmentation to POS tagging, where it is usually meaningless to assign POS tags to incorrectly segmented words.

Sentence segmentation accuracy largely depends on the genre of the datasets, in particular whether punctuation is consistently available, whereas it is less related to linguistic properties and writing systems. In contrast, word segmentation performance is closely correlated with the characteristics of languages. Space-delimited languages are fundamentally easier to segment, and very high accuracy can be obtained even with a very small amount of training data. The segmentation accuracies on languages with no space

delimiters as well as languages with productive non-segmental multi-word tokens are substantially lower.

For morpheme segmentation, the amount of training data has the greatest impact on accuracy. Very high accuracy can be achieved given sufficient amounts of training data as the unique prefixes and suffixes to be identified are very limited and less ambiguous. We do not find notable correlations between any typological factors and segmentation accuracy from the experimental results in Paper V.

Joint segmentation and POS tagging for Chinese is evaluated in Paper III. The proposed model is applicable to Japanese and Vietnamese as well. When compared to previous work, apart from being more accurate in general, the experimental results show that the neural CRF-based model performs well particularly on non-standard texts, such as weblogs, forum text and short messages. Moreover, it achieves better accuracy on small datasets like UD Chinese as it benefits from pre-trained character embeddings and sub-character-level features. In addition, the implemented tagger is very efficient when run on GPU devices and can therefore be applied to tagging very large amounts of texts.

Characterising the difficulty of word segmentation

The difficulty of word segmentation can be characterised with the help of six typological factors, namely character size, lexicon size, average word length, segmentation frequency (segments-space ratio), non-segmental multi-word token portion and non-segmental multi-word token set size (unique non-segmental multi-word tokens). The experimental results in Paper VI indicate that segmentation accuracy is mostly related to segmentation frequency and the prevalence of unique non-segmental multi-word tokens. From the perspective of word segmentation with respect to the proposed factors, the UD languages can be categorised into four groups:

1. Languages with frequent word-internal spaces (Vietnamese).
2. Languages without space-delimiters (Chinese, Japanese).
3. Languages with rich non-segmental multi-word tokens (Arabic, Hebrew).
4. Space-delimited languages without productive multi-word tokens (the other UD languages).

We apply several language-specific settings targeting different language groups to improve the performance of the segmentation model, such as using space-delimited syllables as the basic segmentation units (group 1), applying concatenated 3-grams (group 1, group 2) as well as employing an encoder-decoder as the transducer for non-segmental multi-word tokens (group 3).

Adequate evaluation metrics for segmentation
In Paper IV, we analyse the evaluation metrics for word segmentation. The conclusion applies to sentence segmentation as well. The conventionally employed evaluation metrics, precision and recall, are not adequate for all segmentation tasks. For sentence segmentation and word segmentation, precision and recall are correlated, unlike in general information retrieval tasks. In addition, precision favours under-segmenting system so integrating precision in the evaluation metrics can be misleading. Using recall alone is more adequate for sentence segmentation and word segmentation as it is constrained by the reference segmentation and therefore not biased towards either over-segmenting or under-segmenting systems.

For morpheme segmentation, it is still necessary to employ both precision and recall if the characters not associated with prefixes and suffixes are excluded for evaluation.

5.2 Future Work

The neural network-based segmentation and tagging models explored in this thesis achieve state-of-the-art accuracies on all tasks according to the empirical evaluation on the datasets of different sizes, genres and annotation schemes. The proposed models and released implementations are therefore applicable as practical preprocessing tools. However, as segmentation is a very low-level preprocessing step, errors inevitably propagate to higher-level tasks. As a result, it is still valuable to improve the accuracy of the segmentation and tagging systems in the future, especially for languages that are fundamentally more difficult to process. In addition, the proposed systems heavily rely on GPU devices to obtain high processing speed, but GPU devices are not always accessible in practice.

Under these circumstances, we expect that the current segmentation and tagging models can be improved by:

- Refining the neural network architectures and exploiting more information from character representations for higher accuracy.
- Modifying or substituting the RNN components to increase both training and decoding efficiency, without inducing drastic accuracy loss.
- Optimising the segmentation and tagging speed on CPU devices by reducing parameters.

Despite being completely data-driven and applicable to multiple languages, the proposed models require sufficient training data for supervision to obtain reasonable performance, which is not ideal for low-resource languages with only small or even no annotated training sets. To improve the performance of the model in this case, we propose applying transfer learning as a cross-lingual approach to utilise the models trained on large datasets for low-resource

languages. Similarly to previous work (Collobert et al., 2011; Zhang et al., 2017; Chen et al., 2017b), we expect to find components of neural network models that can be shared across languages. We will specifically focus on transferring between the languages whose character sets are not identical but related, such as Chinese to Japanese and Chinese to Vietnamese. Moreover, having the word segmentation models trained on UD languages, we aim to find an effective model selection criterion with respect to language-specific properties so that any language can be processed even without any annotated data.

In addition, effectively utilising orthographical features from logograms for segmentation and tagging remains an unsolved problem, as the current approaches of using radical embeddings and extracting information from rendered character pictures via CNNs only achieve marginal improvements. Instead of training the radical embeddings and the CNNs for orthographical features jointly with the main BiRNN-CRF model, we will explore ways of encoding the orthographical information into the character embeddings via rich pre-training with large plain corpora.

For named entity transliteration, the neural network-based model presented in Paper II can be improved with some major modifications. Unlike the current pipeline model with several independent components, alignment, segmentation and mapping can be integrated in a unified framework to completely avoid error propagation. Additionally, substantial enhancement can be expected for the neural network-based model by using the multilingual dictionary from Paper I.

Moreover, the proposed segmentation and tagging models can be evaluated extrinsically with higher-level NLP tasks. Sentence segmentation and word segmentation are evaluated within dependency parsing in Papers VI and VII. More systematic experiments can be conducted with additional dependency parsers to make our models fully comparable with similar systems like UDPipe. In addition, sentence segmentation and word segmentation can be evaluated in sentence alignment and word alignment respectively. Various information retrieval tasks can be applied to evaluate word segmentation and transliteration as well.

5.3 Final Remarks

Overall, we hope that the segmentation and tagging models and tools that have been built in this thesis are of great use in the era of artificial intelligence and big data. By processing massive amounts of text effectively, we can utilise the cutting edge machine learning algorithms and information technologies for various applications that benefit our lives. We believe that the development of segmentation and tagging models along with NLP in general will lead us to a bright future.

References

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. TensorFlow: A system for large-scale machine learning. In *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*. Savannah, Georgia, USA, pages 265–283.

Daniel Andor, Chris Alberti, David Weiss, Aliaksei Severyn, Alessandro Presta, Kuzman Ganchev, Slav Petrov, and Michael Collins. 2016. Globally normalized transition-based neural networks. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany, pages 2442–2452.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations*. San Diego, USA.

Rafael E Banchs, Min Zhang, Xiangyu Duan, Haizhou Li, and A. Kumaran. 2015. Report of NEWS 2015 machine transliteration shared task. In *Proceedings of NEWS 2015 The Fifth Named Entities Workshop*. Beijing, China, pages 10–23.

Nan Bernstein-Ratner. 1987. The phonology of parent child speech. *Children's Language* 6(3).

J. Martin Bland and Douglas G. Altman. 1995. Multiple significance tests: the Bonferroni method. *British Medical Journal* 310(6973):170.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics* 5:135–146.

Thorsten Brants. 2000. TnT: a statistical part-of-speech tagger. In *Proceedings of the sixth conference on Applied natural language processing*. Seattle, Washington, USA, pages 224–231.

Michael R Brent. 1999. An efficient, probabilistically sound algorithm for segmentation and word discovery. *Machine Learning* 34(1):71–105.

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)*. Berlin, Germany, pages 409–420.

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 (Short Papers)*. Vancouver, Canada, pages 608–615.

Wanxiang Che, Jiang Guo, Yuxuan Wang, Bo Zheng, Huaipeng Zhao, Yang Liu, Dechuan Teng, and Ting Liu. 2017. The HIT-SCIR system for end-to-end parsing of universal dependencies. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Vancouver, Canada, pages 52–62.

Xinchi Chen, Xipeng Qiu, and Xuanjing Huang. 2017a. A feature-enriched neural model for joint Chinese word segmentation and part-of-speech tagging. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*. Melbourne, Australia, pages 3960–3966.

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*. Beijing, China, pages 1744–1753.

Xinchi Chen, Xipeng Qiu, Chenxi Zhu, Pengfei Liu, and Xuanjing Huang. 2015b. Long short-term memory neural networks for Chinese word segmentation. In *Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal, pages 1197–1206.

Xinchi Chen, Zhan Shi, Xipeng Qiu, and Xuanjing Huang. 2017b. Adversarial multi-criteria learning for Chinese word segmentation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*. Vancouver, Canada, pages 1193–1203.

Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. In *Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*. Doha, Qatar, pages 103–111.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555* v1.

Michael Collins. 2002. Discriminative training methods for hidden Markov models: theory and experiments with perceptron algorithms. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*. Philadephia, Pennsylvania, USA, pages 1–8.

Michael Collins and Brian Roark. 2004. Incremental parsing with the perceptron algorithm. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*. Barcelona, Spain, pages 111–119.

Ronan Collobert, Samy Bengio, and Johnny Marithoz. 2002. Torch: A modular machine learning software library. Technical report, Martigny, Switzerland. https://infoscience.epfl.ch/record/82802/files/rr02-46.pdf.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*. Lille, France, pages 160–167.

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

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning* 20(3):273–297.

Marta R. Costa-jussà. 2016. Moses-based official baseline for NEWS 2016. In *Proceedings of NEWS 2016 The Sixth Named Entities Workshop*. Berlin, Germany, pages 88–90.

Koby Crammer and Yoram Singer. 2003. Ultraconservative online algorithms for multiclass problems. *Journal of Machine Learning Research* 3(Jan):951–991.

Mathias Creutz and Krista Lagus. 2007. Unsupervised models for morpheme segmentation and morphology learning. *ACM Transactions on Speech and Language Processing (TSLP)* 4(1):1–34.

Hal Daumé III and Daniel Marcu. 2005. Learning as search optimization: Approximate large margin methods for structured prediction. In *Proceedings of the 22nd international conference on Machine learning*. ACM, Bonn, Germany, pages 169–176.

Trinh Minh Tri Do and Thierry Artières. 2010. Neural conditional random fields. In *International Conference on Artificial Intelligence and Statistics (AISTAT)*. La Palma, Canary Islands, pages 177–184.

Chuanhai Dong, Jiajun Zhang, Chengqing Zong, Masanori Hattori, and Hui Di. 2016. Character-based LSTM-CRF with radical-level features for Chinese named entity recognition. In *Natural Language Understanding and Intelligent Applications*, Springer, pages 239–250.

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

Nadir Durrani, Hieu Hoang, Philipp Koehn, and Hassan Sajjad. 2014. Integrating an unsupervised transliteration model into statistical machine translation. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*. Gothenburg, Sweden, pages 148–153.

Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A Smith. 2015. Transition-based dependency parsing with stack long short-term memory. 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)*. Beijing, China, pages 334–343.

Marcello Federico, Nicola Bertoldi, and Mauro Cettolo. 2008. IRSTLM: An open source toolkit for handling large scale language models. In *Ninth Annual Conference of the International Speech Communication Association*. Portland, Oregon, USA, pages 187–197.

Shai Fine, Yoram Singer, and Naftali Tishby. 1998. The hierarchical hidden Markov model: Analysis and applications. *Machine learning* 32(1):41–62.

G. David Forney. 1973. The Viterbi algorithm. *Proceedings of the Institute of Electrical and Electronics Engineers* 61(3):268–278.

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 *Thirteenth International Conference on Artificial Intelligence and Statistics*. Sardinia, Italy, pages 249–256.

Sharon Goldwater, Thomas L. Griffiths, and Mark Johnson. 2006. Contextual dependencies in unsupervised word segmentation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*. Sydney, Australia, pages 673–680.

Sharon Goldwater and Tom Griffiths. 2007. A fully Bayesian approach to unsupervised part-of-speech tagging. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*. Prague, Czech Republic, pages 744–751.

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*. Copenhagen, Denmark, pages 692–703.

Alex Graves. 2012. Supervised sequence labelling. In *Supervised sequence labelling with recurrent neural networks*, Springer, pages 5–13.

Jack Halpern. 2007. The challenges and pitfalls of Arabic romanization and Arabization. In *Proceedings of Workshop on Computational Approaches to Arabic Script-based Languages*.

Jun Hatori, Takuya Matsuzaki, Yusuke Miyao, and Jun'ichi Tsujii. 2012. Incremental joint approach to word segmentation, POS tagging, and dependency parsing in Chinese. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*. Stroudsburg, Pennsylvania, USA, pages 1045–1053.

Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. 2006. A fast learning algorithm for deep belief nets. *Neural computation* 18(7):1527–1554.

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

Chang-Ning Huang, Yumei Li, and Xiaodan Zhu. 2006. Tokenization guidelines of Chinese text (v5.0, in Chinese). *Microsoft Research Asia* .

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *arXiv preprint arXiv:1508.01991* v1.

Sittichai Jiampojamarn, Grzegorz Kondrak, and Tarek Sherif. 2007. Applying many-to-many alignments and hidden Markov models to letter-to-phoneme conversion. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*. Rochester, New York, USA, pages 372–379.

Wenbin Jiang, Liang Huang, Qun Liu, and Yajuan Lü. 2008. A cascaded linear model for joint Chinese word segmentation and part-of-speech tagging. In *Proceedings of ACL-08: HLT*. Columbus, Ohio, USA, pages 897–904.

Daniel Jurafsky and James H. Martin. 2000. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall PTR.

Bryan Jurish and Kay-Michael Würzner. 2013. Word and sentence tokenization with hidden Markov models. *The Journal of Language Technology and Computational Linguistics* 28(2):61–83.

Allen Kent, Madeline M. Berry, Fred U. Luehrs, and James W. Perry. 1955. Machine literature searching VIII. Operational criteria for designing information retrieval systems. *Journal of the Association for Information Science and Technology* 6(2):93–101.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *The 3rd International Conference for Learning Representations*. San Diego, California, USA.

Tibor Kiss and Jan Strunk. 2006. Unsupervised multilingual sentence boundary detection. *Computational Linguistics* 32(4):485–525.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. 2007. Moses: Open source toolkit for statistical machine translation. In

Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions. Prague, Czech Republic, pages 177–180.

Canasai Kruengkrai, Kiyotaka Uchimoto, Jun'ichi Kazama, Wang Yiou, Kentaro Torisawa, and Hitoshi Isahara. 2009. Joint Chinese word segmentation and POS tagging using an error-driven word-character hybrid model. *IEICE transactions on information and systems* 92(12):2298–2305.

Anoop Kunchukuttan, Pushpak Bhattacharyya, and Mitesh M. Khapra. 2017. Substring-based unsupervised transliteration with phonetic and contextual knowledge. In *Proceedings of the 20st Conference on Computational Natural Language Learning*. Berlin, Germany, pages 270–279.

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*. San Francisco, California, USA, pages 282–289.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of NAACL-HLT*. San Diego, California, USA, pages 260–270.

Haizhou Li, A Kumaran, Vladimir Pervouchine, and Min Zhang. 2009. Report of news 2009 machine transliteration shared task. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration*. Singapore, pages 1–18.

Haizhou Li, Min Zhang, and Jian Su. 2004. A joint source-channel model for machine transliteration. In *Proceedings of the 42nd Annual Meeting on association for Computational Linguistics*. Barcelona, Spain, pages 159–167.

Percy Liang. 2005. *Semi-supervised learning for natural language*. Master's thesis, Massachusetts Institute of Technology.

Wang Ling, Chris Dyer, Alan W. Black, Isabel Trancoso, Ramon Fermandez, Silvio Amir, Luis Marujo, and Tiago Luis. 2015. Finding function in form: Compositional character models for open vocabulary word representation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal, pages 1520–1530.

Chao-Hong Liu and Qun Liu. 2017. Introduction to the shared tasks on cross-lingual word segmentation and morpheme segmentation. In *Proceedings of MLP 2017: The First Workshop on Multi-Language Processing in a Globalising World*. Dublin, Ireland, pages 71–74.

Andrew Logothetis and Vikram Krishnamurthy. 1999. Expectation maximization algorithms for MAP estimation of jump Markov linear systems. *IEEE Transactions on Signal Processing* 47(8):2139–2156.

Chen Lyu, Yue Zhang, and Donghong Ji. 2016. Joint word segmentation, POS-tagging and syntactic chunking. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*. Phoenix, Arizona USA, pages 3007–3014.

Jianqiang Ma and Erhard Hinrichs. 2015. Accurate linear-time Chinese word segmentation via embedding matching. 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)*. Beijing, China, pages 1733–1743.

Shuming Ma and Xu Sun. 2016. A new recurrent neural CRF for learning non-linear edge features. *arXiv preprint arXiv:1611.04233* .

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. Berlin, Germany, pages 1064–1074.

S. S. Antman J. E. Marsden, L. Sirovich S. Wiggins, L. Glass, R. V. Kohn, and S. S. Sastry. 1993. *Interdisciplinary Applied Mathematics*. Springer.

Sven Martin, Jörg Liermann, and Hermann Ney. 1998. Algorithms for bigram and trigram word clustering. *Speech communication* 24(1):19–37.

Bernard Merialdo. 1994. Tagging English text with a probabilistic model. *Computational linguistics* 20(2):155–171.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* v3.

Shotaro Misawa, Motoki Taniguchi, Yasuhide Miura, and Tomoko Ohkuma. 2017. Character-based bidirectional LSTM-CRF with words and characters for Japanese named entity recognition. In *Proceedings of the First Workshop on Subword and Character Level Models in NLP*. Copenhagen, Denmark, pages 97–102.

Daichi Mochihashi, Takeshi Yamada, and Naonori Ueda. 2009. Bayesian unsupervised word segmentation with nested Pitman-Yor language modeling. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*. Singapore, pages 100–108.

Graham Neubig, Chris Dyer, Yoav Goldberg, Austin Matthews, Waleed Ammar, Antonios Anastasopoulos, Miguel Ballesteros, David Chiang, Daniel Clothiaux, Trevor Cohn, Kevin Duh, Manaal Faruqui, Cynthia Gan, Dan Garrette, Yangfeng Ji, Lingpeng Kong, Adhiguna Kuncoro, Gaurav Kumar, Chaitanya Malaviya, Paul Michel, Yusuke Oda, Matthew Richardson, Naomi Saphra, Swabha Swayamdipta, and Pengcheng Yin. 2017. DyNet: The dynamic neural network toolkit. *arXiv preprint arXiv:1701.03980* .

Hwee Tou Ng and Jin Kiat Low. 2004. Chinese part-of-speech tagging: One-at-a-time or all-at-once? word-based or character-based? In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*. Barcelona, Spain, pages 277–284.

Cam-Tu Nguyen, Trung-Kien Nguyen, Xuan Hieu Phan, Le-Minh Nguyen, and Quang-Thuy Ha. 2006. Vietnamese word segmentation with CRFs and SVMs: An investigation. In *Proceedings of the 20th Pacific Asia Conference on Language, Information and Computation*. Bali, pages 215–222.

Joakim Nivre. 2008. Algorithms for deterministic incremental dependency parsing. *Computational Linguistics* 34(4):513–553.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal dependencies v1: A multilingual treebank collection. In *Proceedings of the 10th International Conference on Language Resources and Evaluation*. Portoroz, Slovenia, pages 1659–1666.

Christopher Olah. 2015. Understanding LSTM networks. *GITHUB blog, posted on August* 27. http://colah.github.io/posts/2015-08-Understanding-LSTMs/.

Constantine P. Papageorgiou. 1994. Japanese word segmentation by hidden Markov model. In *Proceedings of the Workshop on Human Language Technology*. Strouds-

burg, Pennsylvania, USA, pages 283–288.

Wenzhe Pei, Tao Ge, and Baobao Chang. 2014. Max-margin tensor neural network for Chinese word segmentation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*. Baltimore, Maryland, USA, pages 293–303.

Jian Peng, Liefeng Bo, and Jinbo Xu. 2009. Conditional neural fields. In *Advances in neural information processing systems*. Vancouver, Canada, pages 1419–1427.

Nanyun Peng and Mark Dredze. 2016. Improving named entity recognition for chinese social media with word segmentation representation learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Berlin, Germany, pages 149–155.

Nanyun Peng and Mark Dredze. 2017. Multi-task multi-domain representation learning for sequence tagging. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*. Vancouver, Canada, pages 91–100.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar, pages 1532–1543.

Barbara Plank, Anders Søgaard, and Yoav Goldberg. 2016. Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. In *The 54th Annual Meeting of the Association for Computational Linguistics*. Berlin, Germany, pages 412–419.

Hoifung Poon, Colin Cherry, and Kristina Toutanova. 2009. Unsupervised morphological segmentation with log-linear models. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Stroudsburg, Pennsylvania, USA, pages 209–217.

Tao Qian, Yue Zhang, Meishan Zhang, Yafeng Ren, and Donghong Ji. 2015. A transition-based model for joint segmentation, POS-tagging and normalization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Beijing, China, pages 1837–1846.

Lawrence R. Rabiner. 1989. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE* 77(2):257–286.

Sujith Ravi and Kevin Knight. 2009. Learning phoneme mappings for transliteration without parallel data. In *Proceedings of human language technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Boulder, Colorado, USA, pages 37–45.

Nils Reimers and Iryna Gurevych. 2017a. Optimal hyperparameters for deep LSTM-networks for sequence labeling tasks. *arXiv preprint arXiv:1707.06799* .

Nils Reimers and Iryna Gurevych. 2017b. Reporting score distributions makes a difference: Performance study of LSTM-networks for sequence tagging. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Copenhagen, Denmark, pages 338–348.

Eric Sven Ristad and Peter N. Yianilos. 1998. Learning string edit distance. *IEEE Transactions on Pattern Recognition and Machine Intelligence* 20(5):522–532.

Teemu Ruokolainen, Oskar Kohonen, Sami Virpioja, and Mikko Kurimo. 2013. Supervised morphological segmentation in a low-resource learning setting using conditional random fields. In *Proceedings of the Seventeenth Conference on*

Computational Natural Language Learning. Sofia, Bulgaria, pages 29–37.

Hassan Sajjad, Nadir Durrani, Helmut Schmid, and Alexander Fraser. 2011. Comparing two techniques for learning transliteration models using a parallel corpus. In *Proceedings of 5th International Joint Conference on Natural Language Processing*. Chiang Mai, Thailand, pages 129–137.

K. Saravanan, Raghavendra Udupa, and A. Kumaran. 2013. Improving cross-language information retrieval by transliteration mining and generation. In *Multilingual Information Access in South Asian Languages*, Springer, pages 310–333.

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

Noah A. Smith and Jason Eisner. 2005. Contrastive estimation: Training log-linear models on unlabeled data. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*. Ann Arbor, Michigan, USA, pages 354–362.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*. Seattle, Washington, USA, pages 1631–1642.

Anders Søgaard and Yoav Goldberg. 2016. Deep multi-task learning with low level tasks supervised at lower layers. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Berlin, Germany, pages 231–235.

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

Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Highway networks. *arXiv preprint arXiv:1505.00387* v2.

Milan Straka, Jan Hajic, and Jana Straková. 2016. UDPipe: Trainable pipeline for processing CoNLL-U files performing tokenization, morphological analysis, POS tagging and parsing. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*. Portoroz, Slovenia, pages 4290–4298.

Milan Straka and Jana Straková. 2017. Tokenizing, POS tagging, lemmatizing and parsing UD 2.0 with UDPipe. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Vancouver, Canada, pages 88–99.

Yaming Sun, Lei Lin, Nan Yang, Zhenzhou Ji, and Xiaolong Wang. 2014. Radical-enhanced chinese character embedding. In *International Conference on Neural Information Processing*. Kuching, Sarawak, Malaysia, pages 279–286.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. Lake Tahoe, United States, pages 3104–3112.

Yee Whye Teh. 2006. A hierarchical Bayesian language model based on Pitman-Yor processes. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*. Sydney, Australia, pages 985–992.

Yee Whye Teh. 2011. Dirichlet process. In *Encyclopedia of machine learning*, Springer, pages 280–287.

Huihsin Tseng, Pichuan Chang, Galen Andrew, Daniel Jurafsky, and Christopher Manning. 2005. A conditional random field word segmenter for SIGHAN bakeoff 2005. In *Proceedings of the second SIGHAN workshop on Chinese language processing-Volume 171*. Jeju Island, Korea, pages 168–171.

Jurgen Van Gael, Andreas Vlachos, and Zoubin Ghahramani. 2009. The infinite HMM for unsupervised PoS tagging. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2*. Singapore, pages 678–687.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*. Montreal, Canada, pages 6000–6010.

Anand Venkataraman. 2001. A statistical model for word discovery in transcribed speech. *Computational Linguistics* 27(3):351–372.

Chunqi Wang and Bo Xu. 2017. Convolutional neural network with word embeddings for chinese word segmentation. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Taipei, Taiwan, pages 163–172.

Linlin Wang, Zhu Cao, Yu Xia, and Gerard de Melo. 2016. Morphological segmentation with window LSTM neural networks. In *AAAI Conference on Artificial Intelligence*. Phoenix, Arizona, USA, pages 2842–2849.

Yiou Wang, Jun'ichi Kazama, Yoshimasa Tsuruoka, Wenliang Chen, Yujie Zhang, and Kentaro Torisawa. 2011. Improving Chinese word segmentation and POS tagging with semi-supervised methods using large auto-analyzed data. In *Proceedings of 5th International Joint Conference on Natural Language Processing*. Chiang Mai, Thailand, pages 309–317.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2016. Charagram: Embedding words and sentences via character n-grams. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas, pages 1504–1515.

Defu Xia. 1993. *Translation Dictionary for Foreign Names (in Chinese)*. China Translation and Publishing Corporation, Beijing, China.

Fei Xia. 2000. The segmentation guidelines for the Penn Chinese treebank (3.0). *IRCS Technical Reports Series* 37:1–33.

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(02):207–238.

Nianwen Xue. 2003. Chinese word segmentation as character tagging. *Computational Linguistics and Chinese Language Processing* 8(1):29–48.

Nianwen Xue and Libin Shen. 2003. Chinese word segmentation as LMR tagging. In *Proceedings of the second SIGHAN workshop on Chinese language processing-Volume 17*. Sapporo, Japan, pages 176–179.

Dong Yang, Paul Dixon, Yi-Cheng Pan, Tasuku Oonishi, Masanobu Nakamura, and Sadaoki Furui. 2009. Combining a two-step conditional random field model and a joint source channel model for machine transliteration. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration*. Stroudsburg, PA, USA,

pages 72–75.

Zhilin Yang, Ruslan Salakhutdinov, and William W. Cohen. 2017. Transfer learning for sequence tagging with hierarchical recurrent networks. In *Proceedings of the International Conference on Learning Representations*. Toulon, France.

Shiwen Yu, Huiming Duan, Xuefeng Zhu, Bin Swen, and Baobao Chang. 2003. Specication for corpus processing at Peking University: Word segmentation, POS tagging and phonetic notation (in Chinese). *Journal of Chinese Language and Computing* 13(2):121–158.

Daniel Zeman, Martin Popel, Milan Straka, Jan Hajic, Joakim Nivre, Filip Ginter, Juhani Luotolahti, Sampo Pyysalo, Slav Petrov, Martin Potthast, Francis Tyers, Elena Badmaeva, Memduh Gokirmak, Anna Nedoluzhko, Silvie Cinkova, Jan Hajic jr., Jaroslava Hlavacova, Václava Kettnerová, Zdenka Uresova, Jenna Kanerva, Stina Ojala, Anna Missilä, Christopher D. Manning, Sebastian Schuster, Siva Reddy, Dima Taji, Nizar Habash, Herman Leung, Marie-Catherine de Marneffe, Manuela Sanguinetti, Maria Simi, Hiroshi Kanayama, Valeria de Paiva, Kira Droganova, Héctor Martínez Alonso, Çağri Çöltekin, Umut Sulubacak, Hans Uszkoreit, Vivien Macketanz, Aljoscha Burchardt, Kim Harris, Katrin Marheinecke, Georg Rehm, Tolga Kayadelen, Mohammed Attia, Ali Elkahky, Zhuoran Yu, Emily Pitler, Saran Lertpradit, Michael Mandl, Jesse Kirchner, Hector Fernandez Alcalde, Jana Strnadová, Esha Banerjee, Ruli Manurung, Antonio Stella, Atsuko Shimada, Sookyoung Kwak, Gustavo Mendonca, Tatiana Lando, Rattima Nitisaroj, and Josie Li. 2017. CoNLL 2017 shared task: Multilingual parsing from raw text to Universal Dependencies. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Vancouver, Canada, pages 1–19.

Hua-Ping Zhang, Hong-Kui Yu, De-Yi Xiong, and Qun Liu. 2003. HHMM-based Chinese lexical analyzer ICTCLAS. In *Proceedings of the second SIGHAN workshop on Chinese language processing-Volume 17*. Sapporo, Japan, pages 184–187.

Longkai Zhang, Houfeng Wang, Xu Sun, and Mairgup Mansur. 2013. Exploring representations from unlabeled data with co-training for Chinese word segmentation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Seattle, Washington, USA, pages 311–321.

Meishan Zhang, Guohong Fu, and Nan Yu. 2017. Segmenting Chinese microtext: Joint informal-word detection and segmentation with neural networks. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*. Melbourne, Australia, pages 4228–4234.

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)*. Berlin, Germany, pages 421–431.

Yue Zhang and Stephen Clark. 2007. Chinese segmentation with a word-based perceptron algorithm. In *Proceedings of the 45th Annual Meeting on association for Computational Linguistics*. Prague, Czech Republic, pages 840–847.

Yue Zhang and Stephen Clark. 2008. Joint word segmentation and POS tagging using a single perceptron. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*. Columbus, Ohio, pages 888–896.

Yue Zhang and Stephen Clark. 2010. A fast decoder for joint word segmentation and POS-tagging using a single discriminative model. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*. Massachusetts, USA, pages 843–852.

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

Hai Zhao, Chang-Ning Huang, Mu Li, and Bao-Liang Lu. 2006. Effective tag set selection in Chinese word segmentation via conditional random field modeling. In *Proceedings of the 20th Pacific Asia Conference on Language, Information and Computation*. Bali, pages 87–94.

Xiaoqing Zheng, Hanyang Chen, and Tianyu Xu. 2013. Deep learning for Chinese word segmentation and POS tagging. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Seattle, Washington, USA, pages 647–657.