NiuParser: A Chinese Syntactic and Semantic Parsing Toolkit

Jingbo Zhu Muhua Zhu * Qiang Wang Tong Xiao

Natural Language Processing Lab.
Northeastern University

zhujingbo@mail.neu.edu.cn zhumuhua@gmail.com
wangqiangneu@gmail.com xiaotong@mail.neu.edu.cn

Abstract

We present a new toolkit - NiuParser - for Chinese syntactic and semantic analysis. It can handle a wide range of Natural Language Processing (NLP) tasks in Chinese, including word segmentation, part-of-speech tagging, named entity recognition, chunking, constituent parsing, dependency parsing, and semantic role labeling. The NiuParser system runs fast and shows state-of-the-art performance on several benchmarks. Moreover, it is very easy to use for both research and industrial purposes. Advanced features include the Software Development Kit (SDK) interfaces and a multi-thread implementation for system speed-up.

1 Introduction

Chinese has been one of the most popular world languages for years. Due to its complexity and diverse underlying structures, processing this language is a challenging issue and has been clearly an important part of Natural Language Processing (NLP). Many tasks are proposed to analyze and understand Chinese, ranging from word segmentation to syntactic and/or semantic parsing, which can benefit a wide range of natural language applications. To date, several systems have been developed for Chinese word segmentation, part-of-speech tagging and syntactic parsing (examples include Stanford CoreNLP[1], FudanNLP[2], LTP[3] and etc.) though some of them are not optimized for Chinese.

In this paper we present a new toolkit for Chinese syntactic and semantic analysis (call it *NiuParser*[4]). Unlike previous systems, the NiuParser toolkit can handle most of Chinese parsing-related tasks, including word segmentation, part-of-speech tagging, named entity recognition, chunking, constituent parsing, dependency parsing, and semantic role labeling. To the best of our knowledge we are the first to report that all seven of these functions are supported in a single NLP package.

All subsystems in NiuParser are based on statistical models and are learned automatically from data. Also, we optimize these systems for Chinese in several ways, including handcrafted rules used in pre/post-processing, heuristics used in various algorithms, and a number of tuned features. The systems are implemented with C++ and run fast. On several benchmarks, we demonstrate state-of-the-art performance in both accuracy/F1 score and speed.

In addition, NiuParser can be fit into large-scale tasks which are common in both research-oriented experiments and industrial applications. Several useful utilities are distributed with NiuParser, such as the Software Development Kit (SDK) interfaces and a multi-thread implementation for system speed-up.

The rest of the demonstration is organized as follows. Section 2 describes the implementation details of each subsystem, including statistical approaches and some enhancements with handcrafted rules and dictionaries. Section 3 represents the ways to use the toolkit. We also show the performance of the system in Section 4 and finally we conclude the demonstration and point out the future work of NiuParser in Section 5.

* This work was done during his Ph.D. study in Northeastern University.

[1] http://nlp.stanford.edu/software/corenlp.shtml

[2] http://fudannlp.googlecode.com

[3] http://www.ltp-cloud.com/intro/en/

[4] http://www.niuparser.com/index.en.html

Proceedings of ACL-IJCNLP 2015 System Demonstrations, pages 145–150,
Beijing, China, July 26-31, 2015.

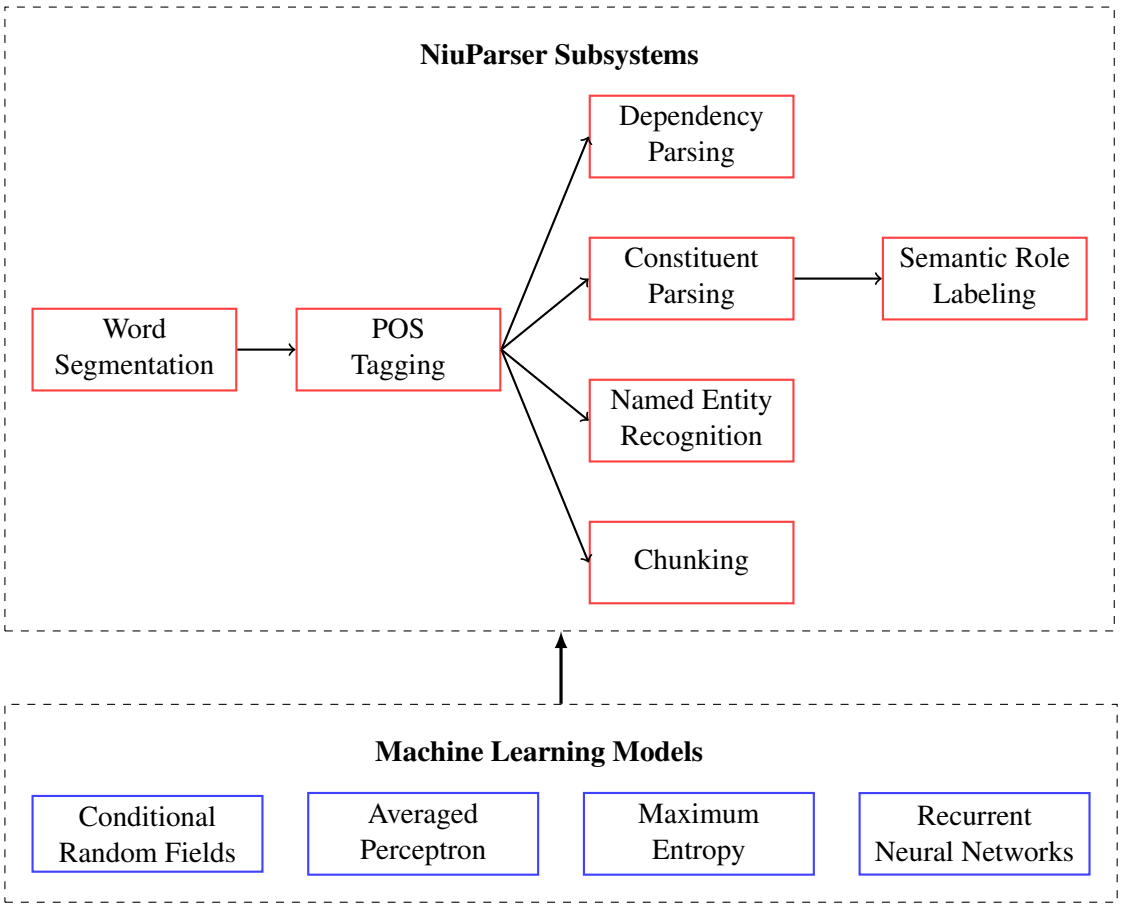

Figure 1: The system architecture of NiuParser.

2 The NiuParser System

2.1 What is NiuParser

The NiuParser system is a sentence-level syntactic and semantic parsing toolkit developed by Natural Language Processing Laboratory in Northeastern University of China. The system is designed specifically to process the Chinese language. Subsystems of NiuParser include word segmentation, POS tagging, named entity recognition, shallow syntactic parsing (chunking), constituent parsing, dependency parsing, and constituent parse-based semantic role labeling. Figure 1 shows the architecture of the NiuParser system. As we can see from the figure, subsystems in NiuParser are organized in a pipeline structure. A given sentence is first segmented into a word sequence, each word in which is assigned a POS tag by the POS tagging subsystem. Based on the POS tagging result, we can choose to do named entity recognition or syntactic parsing. Finally, shallow semantic structures are generated by semantic role labeling on the base of constituent parsing.

2.2 Statistical Approaches to Subsystems

2.2.1 Sequence Labeling

The subsystems of word segmentation, POS tagging, named entity recognition, and chunking in NiuParser are based on statistical sequence labeling models. Specifically, we adopt linear-chain Conditional Random Fields (CRF) (Lafferty et al., 2001) as the method for sequence labeling. Given an input sample $X = x_1, x_2, \ldots, x_L$ and its corresponding sequence $Y = y_1, y_2, \ldots, y_L$, Conditional Random Fields are defined as follows.

$$P_w(Y|X) = \frac{1}{Z_w(X)} exp(W^T \Phi(X, Y))) \quad (1)$$

where $Z_w(X)$ denotes the normalization constant and $\Phi(X, Y)$ are manually defined feature functions. In the testing phase, the Viterbi algorithm is applied to find an optimal label sequence or a k-best list for a testing instance.

With Conditional Random Fields, **Chinese word segmentation** is regarded as a character-based sequence labeling problem. We adopt the scheme of six tags (*B, B2, B3, I, E, O*) to translate

between a segmented sentence and its corresponding label sequence (Zhao et al., 2005). Specifically, *B, B2, B3* denotes the first, the second, and the third character in a word, respectively. *I* means that the character is inside in a word, and *E* means that the character is at the end of a word. Finally, *O* denotes a single-character word. Features include the characters (and their combinations) in a sliding window.

As mentioned above, the NiuParser system utilizes the pipeline method to integrate all the subsystems. That is, **POS tagging**, **named entity recognition**, and **chunking** take the output of the preceding subsystem as input. For POS tagging, we obtain training data from Penn Chinese Treebank (CTB) (Xue et al., 2005), which has 32 POS tags. The named entity recognition subsystem takes the guideline of OntoNotes (Pradhan et al., 2007). Named entities annotated in OntoNotes have 18 entity types in total, including *person names*, *organization names*, and *events*, etc. Table 1 presents a complete list of the entity types in OntoNotes. Chunking uses training data derived from constituent parse trees in CTB. In NiuParser, we consider phrase types including *noun phrase (NP)*, *verbal phrase (VP)*, *quantifier phrase (QP)*, *prepositional phrase (PP)*, *adjective phrase (ADJP)*, and *classifier phrase (CLP)*, etc. Features for the three subsystems are words (and their combinations) in a sliding window. Prefix and suffix of words are also used as features for better system generalization.

2.2.2 Transition-based Parsing

Syntactic parsers can be grouped into two categories according to decoding algorithms: dynamic programming-based and transition-based. For the purpose of efficiency, we implement the constituent and two versions of dependency parsers in the NiuParser system with transition-based methods (Zhu et al., 2013; Zhang and Nivre, 2011; Chen and Manning, 2014). Specifically, parsers are variants of shift-reduce parsers, which start from an initial state and reach a final state by performing an action in each stage transition. Figure 2 and Figure 3 present an example parse of the two parsers, respectively.

One version of the dependency parsers follows the work in (Chen and Manning, 2014), regarding the state transition process as a sequence of classification decisions. In each transition, a best action is chosen by a Neural Network classifier. The other parses (the constituent parser and the other version of dependency parser) utilize exactly the same framework, where both training and decoding phases are formalized as a beam search process. In the decoding phase, the candidate parse with the highest score in the beam will be picked as the parsing result once the beam search process terminates. In the training phase, a beam search-based global online training method is adopted. The training process iterates through the whole training data by decoding the sentences sequently. On each sentence, parameters will be updated immediately once the gold parse is pruned off the beam. In the NiuParser system, we utilize averaged perceptron to learn parameters.

2.2.3 Two-Stage Classification

Researchers in semantic role labeling have explored diverse syntactic structures (chunks, constituent parses, and dependency parses) as input. The semantic role labeling subsystem in NiuParser considers constituent parse trees as input. The subsystem can recognize constituents in a parse tree as arguments with respect to a specified predicate (See Figure 4). Here, semantic role labeling is formalized as a two-stage classification problem. The first stage (called *identification*) conducts a binary classification to decide whether a constituent in a parse tree is an argument. After the first stage, a set of constituents is fed to the second stage (called *classification*) classifier which is a multi-class classifier, used for assigning each argument an appropriate semantic label.

The statistical model used in the semantic role labeling subsystem is Maximum Entropy (Berger et al., 1996), which provides classification decisions with corresponding probabilities. With such probabilities, the identification stage applies the algorithm of enforcing non-overlapping arguments (Jiang and Ng, 2006) to maximize the log-probability of the entire labeled parse tree. In the classification stage, the classifier assigns labels to arguments independently.

2.3 Improvements and Advanced Features

2.3.1 Word Segmentation

In Chinese sentences, words like dates, email addresses, and web page URLs are pervasive but training data for statistical methods is limited in size to cover enough such words. A purely statistical approach often fails to recognize such words once the words do not appear in the training

PERSON	peopel, including fictional	NORP	nationalities or religious or political groups
FACILITY	building, airports, highways, etc.	ORGANIZATION	companies, agencies, etc.
GPE	countries, cities, states	LOCATION	non-GPE, mountain ranges, bodies of water
PRODUCT	vehicles, weapons, foods, etc.	EVENT	named hurricanes, battles, wars, sports events
WORD OF ART	titles or books, songs, etc.	LAW	named documents made into laws
LANGUAGE	named language	DATE	absolute or relative dates or periods
TIME	times smaller than a day	PERCENT	percentage *including ”%”
MONEY	monetary values, including unit	QUANTITY	measurements, as of weight or distances
ORDINAL	”first”, ”second”	CARDINAL	numerals that do not fall under another type

Table 1: Named entity types in OntoNotes

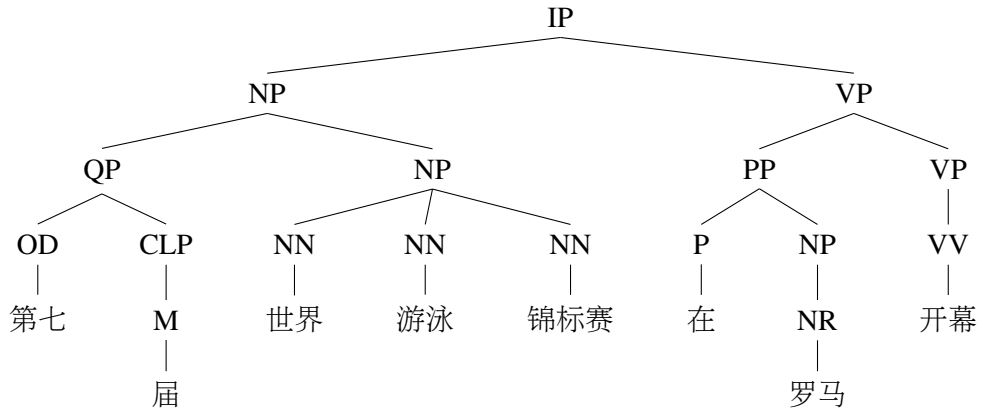

Figure 2: Example of constituent parsing in NiuParser.

data. Fortunately, such words generally have some regular patterns and can be recognized by regular expressions. The NiuParser system provides a regular expression engine to do preprocessing for the CRF-based segmenter.

Post-processing: Besides the word types handled in the preprocessing step, a CRF-based segmenter has a low accuracy in recognizing out-of-vocabulary words. The NiuParser system implements a double-array trie for post-processing. Users can add entries (each entry is a string of characters and its corresponding segments) into a dictionary. String of characters in the dictionary will be assured to be segmented according to its corresponding segments.

2.3.2 Named Entity Recognition

In academics, named entity recognition often suffers from limited training data. In contrast, practitioners generally seek to mine a large-vocabulary entity dictionary from the Web, and then use the entity dictionary to recognize entities as a maximum matching problem. This approach, however, fails to resolve ambiguities. The improvement here is to combine dictionary-based methods and statistical methods.

We first use the forward maximum matching approach to recognize entities in an input sentence by using an entity dictionary. The recognition result is then sent to a CRF-based recognizer. Here each word is assigned a label (start of an entity, inside an entity, or end of an entity) according to the maximum matching result. The labels are used as additional features in the CRF-based recognizer. This approach is similar to the stacking method.

2.3.3 System Speed-up

In addition to fast algorithms (e.g., shift-reduce parsing), NiuParser also supports a multithreading mode to make full advantage of computers with more than one CPU or core. In general, the speed can be improved when multiple threads are involved. However, it does not run faster when too many threads are used (e.g., run with more than 8 threads) due to the increased cost of scheduling.

2.4 Usage

The NiuParser system supports three ways to use the functionalities in the toolkit.

First, users can use the toolkit as an executable file in the command lines. Model files and configuration of the system are specified in a configuration file. Input-output files and the functionality to

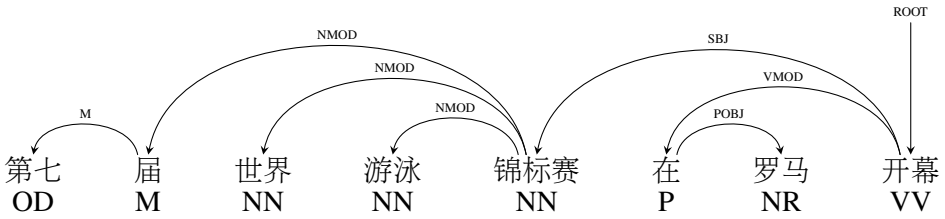

Figure 3: Example of dependency parsing in NiuParser.

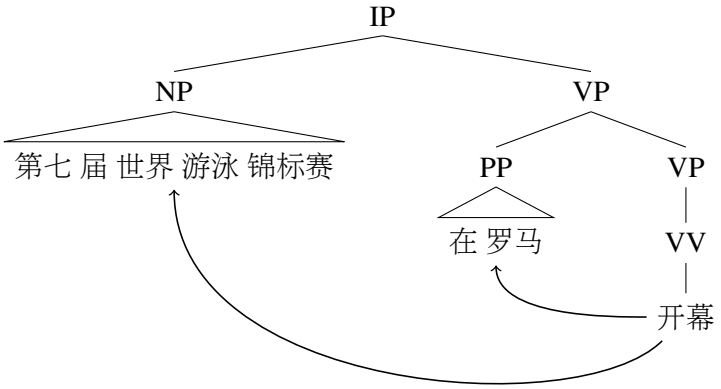

Figure 4: Example of semantic role labeling in NiuParser.

be used are specified as command line arguments.

Second, all the functionalities in NiuParser can be integrated into users' own applications or business process by using the toolkit's SDK interfaces. The SDK supports both Windows and Linux platforms. In contrast to web services, SDK is more suitable to be deployed in the server side.

Third, a demo web page is provided for users to view the analysis results intuitively.[5] All the analysis results are presented graphically.

3 Experiments

We ran our system on several benchmarks. Specifically, we trained and tested word segmentation, POS tagging, chunking, and constituent parsing on CTB5.1: articles 001-270 and 440-1151 were used for training and articles 271-300 were used for testing. The performance of named entity recognition was reported on OntoNotes, where 49,011 sentences were used for training and 1,340 sentences were used for testing. For semantic role labeling, we adopted the same data set and splitting as in (Xue, 2008). Finally, the data set and splitting in (Zhang and Clark, 2011) were used to evaluate the performance of dependency parsing.

All results were reported on a machine with a 800MHz CPU and 4GB memory. See Table 2 for results of acurracy/F1 scores, memory use, model sizes and speed. Note that we evaluated the speed with a single thread and the accuracies were achieved with statistical models only.

From the results we can see that most of the subsystems achieve state-of-the-art performance, (the chunking subsystem is an exception, whose accuracy still have some room left for further improvements.). In addition, the memory use of dependency parsing is extremely heavy. We will optimize the implementation of dependency parsing in our future work.

4 Conclusions and Future Work

We have presented the NiuParser Chinese syntactic and semantic analysis toolkit. It can handle several parsing tasks for Chinese, including word segmentation, part-of-speech tagging, named entity recognition, chunking, constituent parsing, dependency parsing, and constituent parser-based semantic role labeling. The NiuParser system is fast and shows state-of-the-art performance on several benchmarks. Moreover, it supports several advanced features, such as the Software Development Kit (SDK) interfaces and the multi-thread implementation for system speed-up.

In our future work, we will add more function-

[5] http://demo.niuparser.com/index.en.html

Task	Acurrary/F1	Memory Used	Model Size	Speed*
word segmentation	97.3%	68M	57M	45K
POS tagging	93.5%	93M	185M	38.8K
named entity recognition	88.1%	687M	708M	1.87K
chunking	81.1%	71.9MG	90M	18.8K
constituent parsing	83.2%	0.98G	243M	583.3
dependency parsing†	82.4%	2.9G	116M	402.4
dependency parsing‡	82.1%	597M	22M	13.5K
semantic role labeling	68.4%	1.2M/0.9M	30M	494*

Table 2: Evaluation of NiuParser on various tasks. †beam search-based global training method. ‡classification-based method with Neural Networks. *characters per second. * predicates per second.

alities to NiuParser. First of all, we will integrate a new subsystem which conducts dependency-based semantic role labeling. In addition, we will develop a faster constituent parsers by using Recurrent Neural Network. According to the previous work (Chen and Manning, 2014) (and its clone in the NiuParser system), this method reduces the cost of feature extraction and thus shows the advantage in speed. We expect the same approach can be adapted to constituent parsing.

Acknowledges

This work was supported in part by the National Science Foundation of China (Grants 61272376, 61300097, and 61432013).

References

Adam L. Berger, Stephen A. Della Pietra, and Vincent J. Dealla Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguics*, 22:39–71.

Danqi Chen and Christopher D. Manning. 2014. A fast and accurate dependency parser using neural networks. *Proc. of EMNLP 2014*, pages 740–750.

Zheng Ping Jiang and Hwee Tou Ng. 2006. Semantic role labeling of nombank: a maximum entropy approach. *Proc. of EMNLP 2006*, pages 138–145.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *Proc. of ICML 2001*.

Sameer S. Pradhan, Hovy Eduard, Mitch Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2007. Ontonotes: A unified relational semantic representation. *Proc. of ICSC 2007*.

Nianwen Xue, Fei Xia, Chiou Fu-Dong, and Palmer Martha. 2005. The penn chinese treebank: Phrase structure annotation of a large corpus. *Natural Language Engineering*, 11:207–238.

Nianwen Xue. 2008. Labeling chinese predicates with semantic roles. *Computational Linguistics*, 32:225–255.

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

Yue Zhang and Joakim Nivre. 2011. Transition-based dependency parsing with rich non-local features. *Proc. of ACL 2011*, pages 188–193.

Hai. Zhao, Chang-Ning Huang, and Mu Li. 2005. An improved chinese word segmentation system with conditional randome fileds. *Proc. of SIGHAN 2006*, pages 162–165.

Muhua Zhu, Yue Zhang, Wenliang Chen, Min Zhang, and Jingbo Zhu. 2013. A fast and accurate constituent parsing. *Proc. of ACL 2013*.