Language-independent Approach to High Quality Dependency Selection from Automatic Parses

Gongye Jin†, Daisuke Kawahara† and Sadao Kurohashi†

Many knowledge acquisition tasks are tightly dependent on fundamental analysis technologies, such as part of speech (POS) tagging and parsing. Dependency parsing, in particular, has been widely employed for the acquisition of knowledge related to predicate-argument structures. For such tasks, the dependency parsing performance can determine quality of acquired knowledge, regardless of target languages. Therefore, reducing dependency parsing errors and selecting high quality dependencies is of primary importance. In this study, we present a language-independent approach for automatically selecting high quality dependencies from automatic parses. By considering several aspects that affect the accuracy of dependency parsing, we created a set of features for supervised classification of reliable dependencies. Experimental results on seven languages show that our approach can effectively select high quality dependencies from dependency parses.

Key Words: Dependency Parsing, Dependency Selection, Knowledge Acquisition

1 Introduction

Knowledge acquisition from a large corpus has been actively studied in recent years. Fundamental analysis techniques have been applied to a corpus and knowledge is acquired from the analysis. In particular, dependency parsing has been used for tasks like case frame compilation (Kawahara and Kurohashi 2006), relation extraction (Saeger, Torisawa, Tsuchida, Kazama, Hashimoto, Yamada, Oh, Varga, and Yan 2011), and paraphrase acquisition (Hashimoto, Torisawa, Saeger, Kazama, and Kurohashi 2011). For these tasks, the accuracy of dependency parsing is vital. Although the accuracy of state-of-the-art dependency parsers for some languages like English and Japanese is over 90%, it is still not high enough to acquire accurate knowledge. Furthermore, if one tries to apply this method of knowledge acquisition to difficult-to-analyze languages like Chinese and Arabic, the quality of the resulting knowledge worsens.

During the dependency parsing process, even if a parser’s average performance is high, certain types of dependency relations are judged with high accuracy but other types with very low accuracy. In addition, some types of dependency structures are relatively difficult for any parser to correctly analyze. As a result, a parser will tend to produce automatic parses of varying levels

† Graduate School of Informatics, Kyoto University
of quality, depending on the properties of dependency. Using full structures of automatic parses for subsequent tasks, such as the extraction of predicate-argument structures, will inevitably lead to noisy results. In practice, however, several tasks do not require full parse structures but only partial ones (Flannery, Miayo, Neubig, and Mori 2011). To avoid the propagation of errors from automatic analyses, it is preferable to use only high quality dependencies for knowledge acquisition rather than automatic parses. In this study, we present a supervised language-independent approach for selecting high quality dependencies from automatic parses. This method considers linguistic features that are related to the level of difficulty inherent in dependency parsing. We use a single set of dependency labeled data, such as Treebank, part of which is used to train a dependency parser. We conduct experiments on seven languages, five of which are Indo–European languages and two non-Indo–European languages (Chinese and Japanese). The experimental results show that for all the languages, our proposed method can select high quality dependencies than baseline methods.

This study is organized as follows. Section 2 reviews some research relevant to our approach. Section 3 describes the high quality dependency selection process. Section 4 presents a detailed description of our research, conducted using three languages, along with the results. Section 5 describes several additional experiments conducted using other languages. Section 6 presents a discussion of our results. Finally, Section 7 presents a conclusion of our approach and proposes future work.

2 Related Work

There have been several approaches devoted to automatic selection of high quality parses or dependencies. According to selection algorithms, they can be categorized as supervised and unsupervised methods.

Supervised methods primarily focus on the construction of a machine learning classifier and predict the reliability of parses or dependencies on the basis of various syntactic and semantic features. Yates, Schoenmackers, and Etzioni (2006) created WOODWARD, which is a Web-based semantic filtering system. They first mapped the parses to a logic-based representation called relational conjunction (RC). Thereafter, four different methods were employed to analyze whether a conjunct in the RC representation was likely to be reasonable. Kawahara and Uchimoto (2008) built a binary classifier that determines the reliability of each parse. The linguistic features they used for the classification, such as sentence length, the number of unknown words, and the number of commas, are based on the idea that the reliability of parses is determined by the degree
of sentence difficulty. The work most related to ours is that of Yu, Kawahara, and Kurohashi (2008). They proposed a framework that selects high quality parses in the first stage and then high quality dependencies from those parses. In comparison with their work, we consider that even some low quality sentences can contain high quality dependencies. In addition, we take into account other characteristics that can directly affect high quality dependency selection, such as context information and tree features.

Among supervised methods, ensemble approaches were also proposed. For example, Reichart and Rappoport (2007) judged parse quality using a Sample Ensemble Parse Assessment (SEPA) algorithm. They trained several different parsers by using samples from the training data. Thereafter, the level of agreement among those parsers was used to predict the quality of the parse. Another similar approach proposed by Sagae and Tsujii (2007) also selected high quality parses by computing the level of agreement among different parser outputs. However, different from the research previously mentioned, which used several constituency parsers trained with different training data sets, they used a single data set to train different dependency parsing algorithms. Different from the abovementioned methods, our method judges the reliability of each dependency produced by a parser.

Unsupervised algorithms for detecting reliable dependency parses have been proposed. Observing the score of each edge is a basic method for judging the reliability of each dependency (Iwatate 2012). Goldberg and Elhadad (2012) proposed a method of assigning each edge a risk score as the inverse of the confidence score defined by a risk function. However, selecting dependencies purely on the basis of such scores is not sufficient for judging high quality dependencies. We reached this same conclusion in our comparative evaluations of dependency selection approaches on different languages. Reichart and Rappoport (2009) proposed an unsupervised method for high quality parse selection, based on the idea that syntactic structures frequently produced by a parser are more likely to be accurate than those produced less frequently. They developed POS-based Unsupervised Parse Assessment Algorithm (PUPA) to calculate the statistics summarizing the POS tag sequences of parses produced by an unsupervised constituency parser.

Dell’Orletta, Venturi, and Montemagni (2011) proposed ULISSE (Unsupervised LInguiStically driven Selection of dEpendency parses), which is also an unsupervised system. Different from PUPA, this method addressed the reliable parse selection task using an unsupervised method in a supervised parsing scenario. Rather than using constituency-related features, such as ordered POS tag sequences, they used dependency-motivated features, such as parse tree depth and dependency link length. Although unsupervised methods may solve the domain adaption
issue and do not use costly annotated data, the accuracy of selected parses, which is less than 95%, still needs to be improved for knowledge acquisition tasks.

3 High Quality Dependency Selection

In this section, we present a framework for highly reliable dependency selection from automatic parses. Figure 1 shows the overview of our approach. We use a part of a treebank dataset to train a parser and another part to train a binary classifier that judges the reliability of a dependency. We use support vector machines (SVMs) for this classification.

3.1 Training Data for Dependency Selection

Supervised methods always require manually annotated training data that are usually very expensive to obtain. Owing to limited existing resources, we train a classifier for selecting highly reliable dependencies from parsing outputs using training data from the same treebank that is used in the first stage dependency parsing. In particular, the standard training section of the treebank is used to train a dependency parser and then the development section is used to apply dependency parsing using the previously trained parser. From the output parses of the development section, we acquired training data for dependency selection by collecting each dependency. We then judge the success of each dependency according to the gold standard data. All correct dependencies were used as positive training examples for dependency selection and vice versa.

![Fig. 1 Overview of high quality dependency selection](image-url)
3.2 Dependency Selection

We judge the dependency in each parse and retain only high quality output for knowledge acquisition. There are many factors that affect the parsing performance, such as distance between dependencies and complexity of tree structures. By taking these factors into consideration, we create sets of features for selecting high quality dependencies. Tables 1, 2, and 3 present the details of these features.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>POS&lt;sub&gt;head&lt;/sub&gt;, POS&lt;sub&gt;mod&lt;/sub&gt;</td>
<td>Part of speech pair of head and modifier</td>
</tr>
<tr>
<td>Word&lt;sub&gt;head&lt;/sub&gt;, Word&lt;sub&gt;mod&lt;/sub&gt;</td>
<td>Word pair of head and modifier</td>
</tr>
<tr>
<td>Distance</td>
<td>Distance between the head and its modifier</td>
</tr>
<tr>
<td>HasComma</td>
<td>If there exists a comma between head and modifier, set as 1; otherwise set as 0</td>
</tr>
<tr>
<td>HasColon</td>
<td>If there exists a colon between head and modifier, set as 1; otherwise set as 0</td>
</tr>
<tr>
<td>HasSemiColon</td>
<td>If there exists a semi-colon between head and modifier, set as 1; otherwise set as 0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HasVerb</td>
<td>If there exists a verb between head and modifier, set as 1; otherwise set as 0</td>
</tr>
<tr>
<td>POS&lt;sub&gt;head&lt;/sub&gt;−1/mod−1</td>
<td>Part of speech tag of the preceding word of head and modifier</td>
</tr>
<tr>
<td>POS&lt;sub&gt;head&lt;/sub&gt;+1/mod+1</td>
<td>Part of speech tag of the following word of head and modifier</td>
</tr>
<tr>
<td>Word&lt;sub&gt;head&lt;/sub&gt;−1/mod−1</td>
<td>The preceding word of head and modifier</td>
</tr>
<tr>
<td>Word&lt;sub&gt;head&lt;/sub&gt;+1/mod+1</td>
<td>The following word of head and modifier</td>
</tr>
<tr>
<td>POS&lt;sub&gt;head&lt;/sub&gt;−2/mod−2</td>
<td>Part of speech tag of the preceding second word of head and modifier</td>
</tr>
<tr>
<td>POS&lt;sub&gt;head&lt;/sub&gt;+2/mod+2</td>
<td>Part of speech tag of the following second word of head and modifier</td>
</tr>
<tr>
<td>Word&lt;sub&gt;head&lt;/sub&gt;−2/mod−2</td>
<td>The preceding second word of head and modifier</td>
</tr>
<tr>
<td>Word&lt;sub&gt;head&lt;/sub&gt;+2/mod+2</td>
<td>The following second word of head and modifier</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>POS&lt;sub&gt;Grand&lt;/sub&gt;</td>
<td>Part of speech tag of grandparent node of a modifier</td>
</tr>
<tr>
<td>Word&lt;sub&gt;Grand&lt;/sub&gt;</td>
<td>Word of grandparent node of a modifier</td>
</tr>
<tr>
<td>POS&lt;sub&gt;Luncle/Runcle&lt;/sub&gt;</td>
<td>Part of speech tag of the leftmost/rightmost uncle node of a modifier</td>
</tr>
<tr>
<td>Word&lt;sub&gt;Luncle/Runcle&lt;/sub&gt;</td>
<td>Word of the leftmost/rightmost uncle node of a modifier</td>
</tr>
<tr>
<td>POS&lt;sub&gt;Lchild/Rchild&lt;/sub&gt;</td>
<td>Part of speech tag of the leftmost/rightmost child node of a modifier</td>
</tr>
<tr>
<td>Word&lt;sub&gt;Lchild/Rchild&lt;/sub&gt;</td>
<td>Word of the leftmost/rightmost child node of a modifier</td>
</tr>
</tbody>
</table>
3.2.1 Basic Features

If there is a comma, colon or semi-colon between two words, they are much less likely to have a dependency relation than those pairs that do not contain any punctuation. It is much more difficult for parsers to correctly analyze dependencies that contain punctuation than those without punctuation. We use the most common punctuations as features for dependency selection. A dependency relation between words is much more likely when arguments are nearby (McDonald and Nivre 2007). Therefore, distance between words is also an important factor that reflects the difficulty of judging dependency relations. Yu et al. (2008) used the abovementioned features but did not use Word\textsubscript{head}, Word\textsubscript{mod}, or the context features, which are described in the next section.

3.2.2 Context Features

In addition to these basic features, we consider context features that are thought to affect parsing performance. Table 2 lists these context features. For example, the two sentences “they eat salad with a fork” and “they eat salad with sauce” contain the PP-attachment ambiguity problem, which is one of the most difficult problems encountered in parsing. The two prepositional phrases “with a fork” and “with sauce” depend on the verb “eat” and the noun “sauce,” respectively. However, a dependency parser can hardly resolve these two cases. Therefore, we tend to judge this type of structure as unreliable. Consider another similar sentence “they eat it with a fork.” Because the prepositional phrase “with a fork” cannot depend on the pronoun “it” but only on the verb “eat,” this case can be clearly judged as a highly reliable dependency. In some more complex cases, it is also necessary to observe a larger span of context. To learn such linguistic characteristics automatically, besides POS tags and the head and modifier in a dependency, we also use the preceding and following one and two words, respectively, along with the POS tags of the abovementioned linguistic characteristics.

Another important fact is that verb phrases in the dependency tree structure of a parse are normally the root node of the entire dependency tree or the parent node of a subtree. When a word pair contains a verb phrase between them, the two words are always on different sides of the parent node. Thus, these kinds of word pairs normally have no dependency link between them. For example, in SVO (subject-verb-object) languages such as English and Chinese, the subject appears first, the verb second, and then the object third. The most common example of this is when the subjects and objects located on both sides of the verb are the modifiers of the verb. Therefore, argument pairs that have a verb between them rarely have a dependency relation. Observing whether there are verb phrases between head-modifier pairs can help judge
dependency reliability.

3.2.3 Tree Features

The input for our high quality dependency selection method is a dependency tree. It is natural to use tree features to identify dependency quality. On the basis of a head-modifier dependency, we observe modifiers of a modifier, i.e., children nodes, a head’s parent node—which are called grandparent nodes—and children nodes of the grandparent node, which we call uncle nodes.

3.2.4 Edge Score

Some dependency parsers output the score of each dependency (i.e., edge confidence value) during the parsing process. A high score indicates a high possibility that the dependency is correct. However, utilizing this score as the only feature is not sufficient for acquiring high quality dependencies, especially in low quality parses. We consider the real value of the score as an additional feature.

4 Main Experiments

4.1 Experimental Settings

We first conducted our experiment on three languages, including English, Japanese, and Chinese, using the data from the CoNLL-2009 shared task (Hajič, Ciaramita, Johansson, Kawahara, Martí, Márquez, Meyers, Nivre, Padó, Štěpánek, Straňák, Surdeanu, Xue, and Zhang 2009). For each language, we employed the MSTparser\(^1\) (version 0.5.0) as a base dependency parser and use the training data to train a dependency parsing model. KD-Fix (0.05 * 20) confidence score (Mejer and Crammer 2012) is one of the edge score calculation methods in the MSTparser, which was reported as the best method for the MSTparser to predict edge scores. We used the KD-Fix value as the edge score in all experiments. We applied the development data to the dependency parsing model to acquire the training data for dependency selection. We used automatic POS tags for the dependency selection approach (automatic segmentation for Japanese and Chinese). The MXPOST\(^2\) tagger was used for English automatic POS tagging, and for Chinese, we employed MMA (Kruengkrai, Uchimoto, Kazama, Wang, Torisawa, and Isahara 2009) to apply both segmentation and POS tagging. We used JUMAN\(^3\) for Japanese morphological analysis.

\(^1\) http://www.seas.upenn.edu/~strctlrn/MSTParser/MSTParser.html
\(^2\) http://www.inf.ed.ac.uk/resources/nlp/local_doc/MXPOST.html
\(^3\) http://nlp.ist.i.kyoto-u.ac.jp/EN/index.php?JUMAN
From the dependency parser output, we collected training data for high quality dependency selection. All correct dependencies, according to the gold standard data, were defined as positive examples and vice versa. We utilized SVMs to complete this binary classification task, specifically, we employed SVM-Light\(^4\) with a linear kernel. The option -j ratio was used to solve the positive and negative imbalance in the training data for the classifier, where ratio was calculated by dividing the number of negative samples by the number of positive samples.

In order to compare these results with the work done by Yu et al. (2008), we set the basic feature as a baseline. The evaluation data for each language was then used to evaluate the effectiveness of dependency selection.

### 4.2 Evaluation Metrics

On the basis of the output of the SVMs, we selected dependencies with output scores greater than a specific threshold, with a higher output score indicating a more reliable dependency. As a result, a high threshold meant a low recall. Thereafter, we evaluated the selected dependencies by calculating the percentage of correct head-modifier dependencies (excluding punctuations) according to the gold standard data. Precision and recall were calculated as follows.

\[
\text{precision} = \frac{\text{# of correctly retrieved dependencies}}{\text{# of dependencies retrieved}}
\]

\[
\text{recall} = \frac{\text{# of correctly retrieved dependencies}}{\text{# of correct dependencies in gold standard data}}
\]

In automatically tagged and parsed Chinese and Japanese data, there were segmentations that were incorrectly produced. These cases were treated as incorrect instances. Note that the maximum recall for each language was equal to the precision of the base parser.

### 4.3 Experimental Results

#### 4.3.1 Effectiveness of Dependency Selection

Figure 2 shows the precision-recall curves of the selected dependencies using SVMs. Different criteria were used for all languages. The curves labeled “basic feat.” indicates that basic features were used for the dependency selection. “Score” is the method that selects dependencies with the edge scores (MSTparser’s KD-Fix) higher than the threshold. Note that this method does not use SVMs. “Prop feat.” indicates that proposed features were used, in which the real values of the edge scores were used as a feature in the classifier. We can see from the results that our proposed features outperformed the other feature sets in most cases, especially for Chinese and

\(^4\) [http://svmlight.joachims.org/](http://svmlight.joachims.org/)
To investigate the feature that is most effective on the dependency selection, we plotted different precision-recall curves using different feature combinations (Figure 3). Note that “all” is equal to our proposed method in Figure 2, which uses all features. We can see that all features work differently for different languages. For example, edge scores can effectively help select high quality dependency for English but context and tree-based features perform better on Japanese. Those languages have different base parser’s performance and different dependency styles (e.g., head-final for Japanese). We speculate that features such as distance between arguments and comma have a significant influence on Japanese dependency selection.

4.3.2 Statistics of Selected Dependencies

In this section, we investigate the distribution of dependencies in order to determine the primary types selected. This investigation lets us observe whether selected dependencies are biased.
towards certain types of POS (e.g., meaningless patterns, such as “DT NN”). Each dependency type was represented by coarse-grained POS pairs (the first two characters of the POS names). Figures 4 and 5 summarize the statistics for selected dependency POS pairs using different methods for different languages. In each figure, the left and middle graphs represent the dependencies selected under different thresholds (i.e., 50% and 20% recall, respectively), and the rightmost graphs are plotted without the dependency selection.

For all languages, under both selection methods (i.e., proposed method and the selection method by edge score), dependencies with nouns were dominant across all types. For large-scale knowledge acquisition, however, dependencies with verbs are most important because verb phrases always contain most of the information about a specific event. Therefore, verb phrase extraction is key in recognizing predicate-argument structures in knowledge acquisition. The pattern “NN VB” also was prominent, which indicates that predicate-argument dependencies were still selected quite frequently among the high quality dependencies.

Selecting dependencies using different features will inevitably lead to the loss of some informa-

![Fig. 4](image_url) Statistics summarizing dependency POS tags that use the proposed method: dependencies without selection (right), dependencies with 50% recall (middle), dependencies with 20% recall (left)
tive patterns along with those that are considered noisy. From the results, both selection methods have similar tendencies in selecting various types of dependencies. The key point here is that our proposed method, which used different types of features, still produced a high proportion of informative dependencies.

5 Additional Experiments

5.1 Experiments on Other Languages

We applied additional multilingual experiments on Catalan, Czech, Spanish, and German using the CoNLL-2009 shared task data. The MSTparser again was used for dependency parsing. Note that these languages possess the non-projective property, which indicates crossing edges in a dependency tree. They allow for more constructions than the projective constraint. As a result, the non-projective option in MSTparser was triggered. Figure 6 shows the precision-recall curves of dependency selection for these four languages. We directly used the 6th column in the CoNLL-2009 shared task data as automatic POS tags.
For each language, we exhibited the precision at various recall values in order to highlight the quality of selected dependencies under different selection thresholds. Table 4 shows the precision of dependencies selected by our method under 20% and 50% recall for all languages. From the results, we can see that the dependency selection method that used our proposed features outperformed the method that used the basic features for all languages. For Japanese, Chinese,

![Precision-recall curves of dependency selection in four different Indo-European languages: Catalan (top left), Czech (top right), Spanish (bottom left), and German (bottom right)](image)

**Table 4**  Precision of selected dependencies under different criteria

<table>
<thead>
<tr>
<th></th>
<th>English</th>
<th>Japanese</th>
<th>Chinese</th>
<th>Catalan</th>
<th>Czech</th>
<th>Spanish</th>
<th>German</th>
</tr>
</thead>
<tbody>
<tr>
<td>base parser</td>
<td>0.906</td>
<td>0.877</td>
<td>0.825</td>
<td>0.845</td>
<td>0.836</td>
<td>0.855</td>
<td>0.861</td>
</tr>
<tr>
<td>recall = 20%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>basic feat.</td>
<td>0.976</td>
<td>0.980</td>
<td>0.948</td>
<td>0.945</td>
<td>0.938</td>
<td>0.966</td>
<td>0.972</td>
</tr>
<tr>
<td>score</td>
<td>0.996</td>
<td>0.972</td>
<td>0.966</td>
<td>0.985</td>
<td>0.994</td>
<td>0.995</td>
<td>0.994</td>
</tr>
<tr>
<td>prop feat.</td>
<td>0.996</td>
<td>0.988</td>
<td>0.978</td>
<td>0.995</td>
<td>0.994</td>
<td>0.996</td>
<td>0.994</td>
</tr>
<tr>
<td>recall = 50%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>basic feat.</td>
<td>0.965</td>
<td>0.967</td>
<td>0.900</td>
<td>0.935</td>
<td>0.921</td>
<td>0.945</td>
<td>0.943</td>
</tr>
<tr>
<td>score</td>
<td>0.990</td>
<td>0.966</td>
<td>0.950</td>
<td>0.968</td>
<td>0.971</td>
<td>0.985</td>
<td>0.983</td>
</tr>
<tr>
<td>prop feat.</td>
<td>0.990</td>
<td>0.981</td>
<td>0.955</td>
<td>0.985</td>
<td>0.976</td>
<td>0.987</td>
<td>0.985</td>
</tr>
</tbody>
</table>
and Catalan, using the proposed features had a greater effect on high quality dependency selection than other languages, compared to the method only using the edge scores.

5.2 Experiment on Different Domain

One of the biggest problems that most data-driven parsers face is the domain adaptation problem in that their accuracy decreases significantly owing to the lack of domain-specific knowledge. We applied the dependency parsing model trained on section 2 to section 21 of the Penn Treebank (PTB) to the Brown corpus, and obtained an unlabeled attachment score of 0.832, which is significantly lower than the in-domain score by 7.4%.

We applied the same dependency selection model trained on the PTB training sections to the pBrown corpus. Figure 7 shows the precision-recall curves for dependency selection on the PTB test section (section 23) and the Brown corpus using different selecting methods. Similarly to previous figures, “brown basic” indicates selection using basic features. “Brown score” indicates only edge scores were used for classification. “Brown prop.” and “ptb prop.” represent proposed features were used for dependency selection on brown corpus and PTB, respectively. From the results, we can see that our proposed method outperforms the one that used basic features, and also has an advantage over the method that used edge scores with low recall. For examples, when the recall was 20%, high quality dependencies with a precision greater than 98% could be acquired. This shows that our method works well on data from different domains and that there is a way to acquire knowledge from a large raw corpus in different domains (e.g., the Web).

![Fig. 7 Precision-recall curves of dependency selection from the Brown corpus](image-url)
5.3 Experiment using Different Proportions of Training Data

To determine the amount of training data required to achieve a reasonable dependency selection performance, we used different proportions of the PTB development section to train different classifiers. Thereafter, we used the same test set to evaluate each classifier using the same metric described in Section 4. Note that the method that uses only the edge scores does not change with different data proportions (i.e., one should get similar classification performance using the edge score as a feature regardless of the size of the training data set). Thus, we only used features including “basic”, “context”, and “tree” in this experiment. Figure 8 shows the precision-recall curves of dependency selection obtained using different sized training data sets. The performance decreases slightly when the training data set decreases in size. However, a precision greater than 98% is still achievable when the recall is 20%, even when using only 25% of the training data. From the results, we can see that training data sets of various sizes can be used to train effective classifiers.

5.4 Experiment using a Different Parser

As MSTparser is a graph-based dependency parser, we wanted to further test our proposed framework on different types of syntactic parsers. Therefore, we choose the Stanford parser\(^5\), a probabilistic context-free grammar (PCFG) parser. Although the output of the Stanford parser can be converted into dependency style, it is unable to output an edge score. As a result, we used features including “basic”, “context” and “tree” in this experiment. We conducted the experiment on English and Chinese. We directly used English raw text and Chinese segmented text by MMA as input, and employed the Stanford parser for POS tagging and parsing. Figure 9

---

\(^5\) http://nlp.stanford.edu/software/lex-parser.shtml

---

**Fig. 8** Precision-recall curves of dependency selection obtained using training data sets of various sizes
shows the precision-recall curves for English and Chinese. English can achieve high precision around 98.5% when the recall is 25%. Even though the base parser for Chinese achieved only 67.5%, we can still extract dependencies with over 92% precision.

5.5 Using Dependency Selection in Other Tasks

We applied our proposed dependency selection approach to predicate-argument structure extraction and distributional similarity calculation. First, we applied the dependency selection to automatic parses and used only high quality edges to extract predicate-argument structures. With a central focus on verbs in each dependency tree, we used only verb-dependent arguments and represented each argument by its syntactic surface case (i.e., subject, object, prepositional phrase, etc.) with sets of heuristic conversion rules. Distributional similarity is a method that determines word similarity on the basis of a metric derived from the distribution of the verb and its arguments in a large text corpus (Lin 1998).

For English, we used a large-scale Web corpus that contained 200 million sentences. We employed the Wordsim353 data set for evaluation, which contains human-assigned similarities between each word pair. For Chinese, five million sentences from the Chinese Gigaword were used for distributional similarity calculation. We also used a manually constructed gold-standard data set containing more than 500 word pairs for Chinese word similarity evaluation.

For each word pair in the evaluation set, we evaluated the word similarity using the distributional thesauri calculated using the acquired predicate-argument pairs. We used spearman’s correlation coefficient, calculated by comparing the ranks between the two sets of similarities.

---

6 http://alfonseca.org/eng/research/wordsim353.html
7 http://www.cs.york.ac.uk/semeval-2012/task4/
Data size is an important factor that can affect distributional similarity calculations. Therefore, to compare the thesauri calculated from predicate-argument pairs of various sizes, we randomly sampled different sized sets of predicate-argument pairs. Figure 10 shows the Spearman coefficient values for the distributional similarity calculations under the three criteria mentioned above. As we can see, performance of distributional similarity calculations can be improved by selecting high quality dependencies, for both Chinese and English. Using the dependency selection process, the negative effects that the noisy data can have on semantic representation of word similarity can be effectively reduced. The result shows that the quality of dependencies and data size are both important factors for distributional similarity calculation.

6 Discussion

We achieved dependency precisions of 99.6%, 97.8%, and 98.8% for English, Chinese, and Japanese, respectively, using automatically tagged data with a recall of 20%. All four European languages have around 99.5% precision under this recall. Our proposed features show a significant advantage over the original feature set proposed in the previous study of Yu et al. (2008). By taking into account context and tree information, we can effectively help the system learn automatic dependency parse reliability not only from the same domain but also from other domains. As shown in the experiments, the results are quite promising for different types of syntactic parsers, as well.

The statistics calculated for selected dependencies showed that even when we adopted a low
recall value to obtain a high precision they contained many dependencies related to nouns and verbs. The low recall (e.g., 20%) can be compensated for by using very large raw corpora, which are relatively easy to acquire. The applicability to different domains also allows us to acquire knowledge from large raw corpora of various domains. Moreover, our proposed approach can benefit subsequent NLP tasks, such as distributional similarity calculation.

Each feature type had a different influence on different languages. For example, English and Chinese performed similarly when using edge scores vs. proposed features. However, Japanese showed better performance using our proposed features. We speculate that features such as the distance between arguments and commas had a significant influence on Japanese dependency selection. Even though selection using edge scores can help acquire high quality dependencies, which sometimes have similar performance to our proposed features, it is very important to note that edge scores are not always available for different types of parsers (e.g., Stanford parser). Furthermore, producing edge scores (e.g., KD-Fix of MSTparser) demands a larger amount of computer memory, which normally makes it impractical to apply parallel analyses to large-scale corpora in practice.

7 Conclusion and Future Work

In this study, we proposed a classification approach for high quality dependency selection. We created a set of features that consider context and tree information in selecting highly reliable dependencies from parsed sentences. This approach can extract high quality dependencies even from low quality parses. The experiments showed that our method works for in-domain parses as well as out-of-domain parses.

We can extract high quality dependencies from a large corpus such as the Web, and can subsequently assist knowledge acquisition tasks, such as subcategorization frame acquisition (Korhonen, Krymolowski, and Briscoe 2006) and case frame compilation (Kawahara and Kurohashi 2010), which depend highly on the parse quality.

Since automatic parses can be used to improve the base parser itself (Chen, Kazama, Uchimoto, and Torisawa 2009), we plan to use a bootstrapping strategy to improve dependency parsers based on high quality knowledge acquired from large corpora.
Reference


Gongye Jin: Gongye Jin received his B.S. in Computer and Computing Science from Zhejiang University City College (ZUCC) in 2009. He obtained his M.S. in Informatics from Kyoto University in 2012. He is currently a Ph.D. student in Informatics at Kyoto University.

Daisuke Kawahara: Daisuke Kawahara received his B.S. and M.S. in Electronic Science and Engineering from Kyoto University in 1997 and 1999, respectively. He obtained his Ph.D. in Informatics from Kyoto University in 2005. He is currently an associate professor at the Graduate School of Informatics at Kyoto University. His research interests center on natural language processing, particularly knowledge acquisition and text understanding.

Sadao Kurohashi: Sadao Kurohashi received the B.S., M.S., and Ph.D. in Electrical Engineering from Kyoto University in 1989, 1991 and 1994, respectively. He has been a visiting researcher of IRCS, University of Pennsylvania in 1994.
He is currently a professor of the Graduate School of Informatics at Kyoto University. His research interests include natural language processing, knowledge acquisition/representation, and information retrieval. He received the 10th anniversary best paper award from journal of natural language processing in 2004, 2009 Funai IT promotion award, and 2009 IBM faculty award.

(Received February 20, 2014)
(Revised May 2, 2014)
(Rerevised July 6, 2014)
(Accepted August 29, 2014)