Expert Systems with Applications 41 (2014) 1716–1722

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

ReliAble dependency arc recognition

Wanxiang Che, Jiang Guo, Ting Liu *

School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China

ARTICLE INFO

Keywords:
Natural language processing
Syntactic parsing
Dependency parsing
RADAR
Binary classification

ABSTRACT

We propose a novel natural language processing task, ReliAble dependency arc recognition (RADAR), which helps high-level applications better utilize the dependency parse trees. We model RADAR as a binary classification problem with imbalanced data, which classifies each dependency parsing arc as correct or incorrect. A logistic regression classifier with appropriate features is trained to recognize reliable dependency arcs (correct with high precision). Experimental results show that the classification method can outperform a probabilistic baseline method, which is calculated by the original graph-based dependency parser.

© 2013 Elsevier Ltd. All rights reserved.

1. Introduction

As a fundamental task of natural language processing, dependency parsing has become increasingly popular in recent years. It aims to find a dependency parse tree among words for a sentence. Fig. 1 shows an example of dependency parse tree for a sentence, where `sbj` is a subject, `obj` is an object, etc. (Johansson & Nugues, 2007). Dependency parsing are widely used: in biomedical text mining (Kim, Ohta, Pyysalo, Kano, & Tsujii, 2009), as well as in textual entailment (Androutsopoulos & Malakasiotis, 2010), information extraction (Wu & Weld, 2010; Banko, Cafarella, Soderland, Broadhead, & Etzioni, 2007) and sentiment analysis (Meena & Prabhakar, 2007).

The performance of dependency parsing has increased recently (Kübler, McDonald, & Nivre, 2009). However, when we migrate dependency parsing systems from laboratory demonstrations to high-level applications, even the best parser available today still encounter some serious difficulties.

First of all, parsing performance usually dramatically degrades in real fields because of domain migration. Secondly, since every parser inevitably will make some mistakes during decoding, outputs from any dependency parser are always fraught with a variety of errors. Thus, in some high-level applications which expect to use correct parsing results, it is extremely important to be able to predict the reliability of the auto-parsed results. If these applications just use correct parsing results and ignore incorrect results, their performances may be improved further. For instance, if an entity relation extraction (a kind of information extraction) system, which depends on parsing results heavily (Zhang, Zhang, Su, & Zhou, 2006), only extracts relations from correct parsing sentences, then the system can extract more accurate relations and import less wrong relations through incorrect parsing results. Although some implied relations in those incorrect parsing sentences are missed, these missing relations may be extracted from other sentences that can be parsed correctly while zooming in the data to the whole Web.

Most large-margin based training algorithm for dependency parsing output models that predict a single parse tree of the input sentence, with no additional confidence information about the correctness of it. Therefore, an interesting problem is how to judge a parsing result as correct or not. However, it is difficult to obtain a parse tree in which all sub-structures are parsed correctly. CoNLL 2009 Shared Task results show that only about 40% English and 35% Chinese sentences can be parsed complete correctly (Hajič et al., 2009b). Some previous studies have addressed the problem to recognize reliable parsing results (Reichart & Rappoport, 2007; Dell'Orletta & Venturi, 2011; Kawahara & Kurohashi, 2010; Ravi, Knight, & Soricut, 2008). A parsing result is reliable when, the result is correct with high precision. However, all these studies focus on judging if the parsing results of a whole sentence are reliable or not, which can cause the following problems:

1. The reliable parsing results may still include some wrong parsing sub-structures. Different applications need different key sub-structures, such as backbone structures that are keys for semantic role labeling (Gildea & Jurafsky, 2002) and branch structures are important for multiword expression (Sag, Baldwin, Bond, Copestake, & Flickinger, 2002). If these key sub-structures are parsed incorrectly, even though the whole sentence is parsed with a high reliability, the tiny errors will be still harmful to these given applications. This problem results in a low precision.

* Corresponding author. Tel.: +86 13936137628.
E-mail address: tliu@ir.hit.edu.cn (T. Liu).

0957-4174/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.eswa.2013.08.070

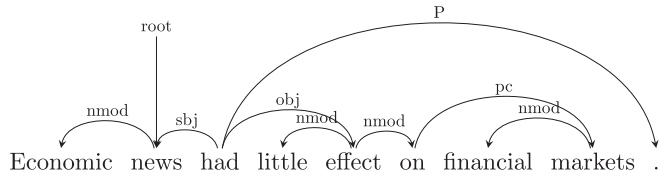

Fig. 1. An example of dependency parse tree.

2. The unreliable parsing results may include some useful sub-structures but should not be discarded totally. For instance, extracting entity relations is possible if the parse tree path is correct between two entities despite other parts of the sentence being incorrectly parsed. Discarding unreliable parsing sentences can result in a low recall.

Therefore, dependency arcs, novel reliability measuring objects for dependency parsing are proposed. A reliable dependency happens when a word can find its parent and label the dependency relation between them correctly with high precision. Once all reliable dependency arcs in a sentence are found, the corresponding parse paths or sub-trees from them can be mapped out. These reliable sub-structures can be used according to the needs of different applications. In paying attention to reliable parts and ignoring unreliable ones in a sentence, the precision of applications can be improved. Meanwhile, when the number of reliable sub-structures is more than that extracted from reliable whole sentences, higher recall can be obtained.

The problem of ReliAble Dependency Arc Recognition (RADAR) can be regarded a binary classification problem. The positive examples are the correctly predicted arcs and the others are the negative examples. Thus, the problem can be converted to find appropriate classifiers and proper features. Different from normal binary classification problems, the data are not balanced for RADAR. For the state-of-the-art dependency parser, the LAS (Labeled Attachment Score) can achieve about 80% in Chinese data set and 90% in English data set (Hajič et al., 2009b), which means that the ratio of the number of correct dependency arcs to the number of incorrect dependency arcs is 4:1 for Chinese and 9:1 for English. Aside from learning from the imbalanced data, how to evaluate RADAR is another issue. The normal evaluation methods based on accuracy are not suitable for the problem. If an incorrect dependency arc is recognized as a correct arc, the cost is larger than the opposite scenario. In addition, the classification accuracy would not be a suitable evaluation metric in an imbalanced scenario. Therefore, there is a need to find more appropriate evaluation criteria.

The rest of the present paper is organized as follows. Section 2 presents related work. Section 3 describes the proposed method. Section 4 discusses the present experimental setting and results. We conclude and set the direction of the future work in Sections 5 and 6 respectively.

2. Related work

To the best of our knowledge, Yates, Schoenmackers, and Etzioni (2006) was the first work to address explicitly the parsing reliability recognition problem. They detected erroneous parses using web-based semantics. In addition, an ensemble method based on different parsers trained on different data sampled from a training corpus to select high quality parsing results was proposed as well (Reichart & Rappoport, 2007). Dell'Orletta and Venturi (2011) was another study that detect reliable dependency parses with some heuristic features. Kawahara and Uchimoto (2008) classified sentences into two classes, reliable and unreliable, with a binary classifier. Ravi et al. (2008) predicted the accuracy of a parser on sets of sentence by fitting a real accuracy curve with linear regression algorithm.[1] However, all these works focused on recognizing reliable parsing results of whole sentences and caused corresponding problems for some applications as discussed in Section 1.

Although the parsing reliability recognition of whole sentences can be used in Active Learning (Settles, 2010) or Semi/Un-supervised Learning (Goldwasser, Reichart, Clarke, & Roth, 2011), recognizing sub-structures reliability is also useful. For instance, some studies (van Noord, 2007; Chen, Kawahara, Uchimoto, & Zhang, 2008, 2009) used sub-trees or word pairs extracted from a large auto-parsed corpus to help the dependency parser. However, the confidence of a sub-tree or a word pair is only expressed by its count that appears in the corpus. Therefore, their methods may be biased toward frequently appearing sub-trees or word pairs, which may be incorrect, and penalizes the sparse but correct ones.

The studies most relevant to ours are done by Atserias, Attardi, Simi, and Zaragoza (2010) and Avihai Mejer (2012). They both reported the similar problem with ours. Atserias et al. (2010) shows how to use the probability scores that a transition-based parser normally computes, in order to assigning a confidence score to parse trees. They assign such score to each arc and the active learning application uses the worst. Another independent work done by Avihai Mejer (2012) describes several methods for estimating the confidence in the per-edge correctness of a predicted dependency parse. The best method they confirmed in their study is based on model re-sampling, which is inefficient. Our work differs in that we proposed a novel supervised approach which makes use of additional information as the features for learning models.

3. Method description

This section introduces the dependency parsing model and a method to estimate the probability of each dependency arc. A binary classification method to recognize reliable arcs follows. Besides a classifier, the classification method includes three sorts of features and a process to construct training data.

3.1. Graph-based dependency parsing

Given an input sentence $\mathbf{x} = w_1 \dots w_n$, a dependency tree is denoted by $\mathbf{d} = \{(h,m,l) : 0 \leqslant h \leqslant n, 0 < m \leqslant n, l \in \mathcal{L}\}$, where (h,m,l) represents a dependency arc $w_h \rightarrow w_m$ whose *head* word (or *father*) is w_h and *modifier* (or *child*) is w_m with a dependency label l, and $\mathcal{L}$ is the set of all possible dependency relation labels. The artificial node w_0, which always points to the root of the sentence, is used to simplify the formalizations.

Then, an optimal dependency tree $\hat{\mathbf{d}}$ is determined based on $\mathbf{x}$:

$$\hat{\mathbf{d}} = \arg\max_{\mathbf{d}} Score(\mathbf{x}, \mathbf{d})$$

Recently, graph-based dependency parsing has gained interest due to its state-of-the-art performance (Kübler et al., 2009). Graph-based dependency parsing views the problem as finding the highest scoring tree from a directed graph. Based on dynamic programming decoding, it can find efficiently an optimal tree in a huge search space. In a graph-based model, the score of a dependency tree is factored into scores of small parts (sub-trees):

$$Score(\mathbf{x}, \mathbf{d}) = \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, \mathbf{d}) = \sum_{p \subseteq \mathbf{d}} Score(\mathbf{x}, p)$$

where $\mathbf{f}(\mathbf{x},\mathbf{d})$ refers to the feature vector and $\mathbf{w}$ is the corresponding weight vector, p is a scoring part that contains one or more dependence arcs in the dependency tree $\mathbf{d}$.

[1] When the size of a set is 1, the accuracy of a sentence can be predicted.

Download English Version:

https://daneshyari.com/en/article/382496

Download Persian Version:

https://daneshyari.com/article/382496

Daneshyari.com