Using Maximal Embedded Syntactic Subtrees for Textual Entailment Recognition

Sophia Katrenko
Human-Computer Studies Laboratory
University of Amsterdam
Kruislaan 419
katrenko@science.uva.nl

Pieter Adriaans
Human-Computer Studies Laboratory
University of Amsterdam
Kruislaan 419
pietera@science.uva.nl

Abstract

In this paper we address the textual entailment task by using tree mining and matching technique. Our results show that accuracy can be improved when using a combination of lexical entailment with syntactic matching. The best result we received by combining two components is the following: 70% of recall and 57,5% of precision on the test set.

1 Introduction

Textual entailment is a complex task which can contribute to variety of applications, such as question answering (QA) systems, information retrieval (IR) or information extraction (IE). Its main objective is to determine whether a meaning of one text (hypothesis) can be inferred from another one. An example of textual entailment is given below:

(h) About two weeks before the trial started, I was in Shapiro's office in Century City.

(t) Shapiro works in Century City.

As this example shows, the recognition of textual entailment can benefit from different NLP techniques and knowledge sources (e.g., WordNet). Among approaches proposed to the recognition of textual entailment the often used methods are lexical and syntactic based. There have also been developed hybrid approaches making use of information stored in WordNet. Most syntactic-oriented approaches have focused on the dependency structure (Maria Tereza Pazienza et al., 2005), (Jesus Herrera et al., 2005), (Milen Kouylekov et al., 2005). The extensive analysis of the contribution of syntactic information to the recognition task has been studied by (Lucy Vanderwende et al., 2005). In this paper we propose yet another approach employing embedded syntactic subtrees.

This paper is organized as follows. First, we discuss tree mining and matching methods. We will proceed to the experiment description and conclude with the directions for the future work.

2 Tree mining and matching

Tree mining and matching methods have been used for many tasks in different domains, including user navigation aiming at finding the most visited web pages, study of topological patterns in RNA and many others. Since the syntactic structure of a sentence can be represented as a dependency tree, it is of considerable interest to study whether tree matching can improve results of the textual entailment recognition.

Definition 1 (Labeled rooted tree) *A labeled rooted tree $T = (V, E, \Sigma, L)$ is a directed acyclic connected graph one vertex of which is distinguished. It consists of a set of vertices V, a set of edges E, an alphabet Σ for labels of both, vertices and edges, and a labeling function $L : V \cup E \rightarrow \Sigma$ assigning labels to vertices and edges.*

When comparing two trees, different types of subtrees can be extracted. These types are defined according to the restrictions on the tree nodes ordering. The most specific case is a subtree where for a given node all its descendants must appear in both trees (bottom-up tree). This restriction can be relaxed by allowing to remove some children (induced subtree) and requesting the ancestor-descendant ordering to be preserved even if some other descendants are missing (embedded subtree). The existence of different types of subtrees makes the use of tree mining more attractive since it is possible to define the granularity of the trees match.

Definition 2 (Embedded tree) *A rooted labeled tree $T' = (V', E', \Sigma', L')$ is an embedded subtree of a rooted labeled tree $T = (V, E, \Sigma, L)$, iff:*

(a) $V' \subset V$

(b) $v', w' \in V'$ and $(v', w') \in E'$ iff there exist $v, w \in V$ and v is an ancestor of w in T

(c) the labeling of V' is preserved in T'

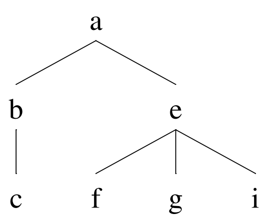

Figure 1: Tree A

An example of the embedded subtree for a tree depicted on Fig.1 is given below.

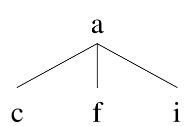

Figure 2: Tree B

Depending on the type of subtrees to be found, there are several algorithms which can be used for tree mining and matching. The excellent overview of different mining techniques is given in (Y. Chi, 2004). For the Textual Entailment challenge we decided to search for the most general, embedded subtrees between the hypothesis and the text fragment. In our view, embedded tree mining can be useful when applied to the dependency structures. It relaxes constraints by allowing some nodes(words) to be different. Such type of tree matching can also be referred to as fuzzy matching.

3 System description

Our system consists of two parts, lexical entailment and syntactic matching.

Given two sentences S_h (hypothesis) and S_t (text), the lexical overlap is calculated as a ratio of the number of lemmata two sentences share to the length of the shortest sentence (usually, hypothesis). $|S_h|$ and $|S_t|$ stand for the length of the hypothesis and the text fragment, respectively.

$$lex = \frac{overlap(S_h, S_t)}{min\{|S_h|, |S_t|\}}$$

For the syntactic matching we have constructed a module which includes parsing and matching steps. We have used Minipar[1] whose output can be presented either by dependency trees or by constituency structures. The output is also accompanied by lemmata of all words in a sentence. We have chosen to work on lemmata only. As a next step, the depth first search has been performed so all trees have been presented in a pre-order format.

Further, syntactic matching has been carried out on each pair of syntactic trees, T_h and T_t. The matching outputs a maximal rooted ordered embedded subtree, T_{emb}. To calculate the similarity, the following measure was used:

$$syn = \frac{|T_{emb}|}{min\{|T_h|, |T_t|\}}$$

Similarly to the previous example, $|T_h|$, $|T_{emb}|$ and $|T_t|$ stand for the size of a tree which equals to the number of nodes each tree contains.

For the syntactic matching we used a method proposed by (Mohammed J. Zaki, 2005). We set the support level to 100% (requesting all nodes in the resulting subtree to be present in two trees to be matched) and searched for the maximal subtree only. Since most methods in tree mining and matching work on the labeled trees where only vertices are labeled, we incorporated the labels of edges into labels of nodes.

Finally, the rule based classifier PART (Ian H. Witten et al., 2005) has been trained based on the scores received by lexical overlap and syntactic matching. This classifier is based on partial decision trees and adopts separate-and-conquere strategy. We trained the classifier using 10-cross fold validation.

[1]Minipar is available from http://www.cs.ualberta.ca/ lindek/minipar.htm

4 Results and Discussion

For the evaluation we submitted two runs. The first (*run1*) combines lexical overlap and syntactic matching, whereas the second (*run2*) is a simple lexical overlap only. As Table1 suggests, the overall accuracy is higher for the *run1*.

Table 1: Accuracy on the test set (official results)

TASK	ACCURACY	
	RUN1	RUN2
TOTAL	**59,00%**	57,13%
QA	**60,50%**	58,00%
SUM	**69,50%**	67,00%
IR	**62,00%**	56,50%
IE	44,00%	**47,00%**

When training classifier on the training set we also noticed that the thresholds for *run2* and *run1* for the lexical overlap were different. This is due to the fact that in the latter case the classifier produced rules using information not only based on the lexical overlap score but also on the scores of syntactic matching.

We have carried out the analysis of how well the methods work on each topic in particular (Table.3). In general, *run1* provides higher precision, while *run2* gives better results on recall. The results of challenge also suggest that the system performs on *IE* topic at worst. Our analysis shows that although recall on *IE* topic is one of the highest for two runs, precision is low. For instance, for the *run2*, only 24 positive instances out of 100 have been missclassified. Similarly to (Roy Bar-Heim et al., 2005), we have analyzed the missclassified examples looking for the type of the information which might improve the classification (Table 2).

Table 2: Missclassified examples in IE

TYPE	**OCCURRENCE**
PARAPHRASES	**11**
SEMANTIC INFORMATION AND BACKGROUND KNOWLEDGE	**18**
ANAPHORA RESOLUTION	**1**

We have noticed that in most cases a combination of several information sources will be needed. In particular, in the mentioned fragments different types of paraphrasing apart from pure structural are involved. We assume therefore that most missclassified fragments would benefit from both, paraphrasing and using additional knowledge sources.

Moreover, some of the examples on IE topic clearly reflect patterns often used in the information extraction task. One of such examples is the snippet 358 with the organization-location relation presented below.

(h) The declaration was the first from PepsiCo to damp speculation, after two weeks of press reports and mounting concern from politicians, including French President Jacques Chirac, that Paris-based Danone would be acquired.

(t) Danone headquarters are located in Paris.

Since the topic annotations were missing in a test set, we trained the classifier on the whole training corpus making no distinction between topics the text snippets belong to. After we have received the annotated test data, we also carried out additional test on the information extraction topic. When trained on IE topic only, the accuracy for IE on the test data increases but the overall accuracy decreases.

Table 3: Precision and recall on the test set (for TRUE category)

TASK	RUN1		RUN2	
	PRECISION	RECALL	PRECISION	RECALL
TOTAL	59,5%	58,91%	55,51%	71,75%
QA	**57,89%**	77%	55,19%	**85%**
SUM	**76,71%**	56%	65,74%	**71%**
IR	**69,35%**	43%	56,70%	**55%**
IE	45,59%	62%	**48,10%**	**76%**

As mentioned in Section 3, we incorporated the labels of edges into the node labels. Consequently, such nodes as $Botswana\_subj$ and $Botswana\_pcomp - n$[2] have been considered to be different and they were not matched by our method. One way to overcome it is to discard syntactic functions and to consider the labels of vertices only. We have conducted this experiment similarly to *run1*, combining the results of lexical overlap and syntactic matching (we refer to this experiment as to *run3*). We also trained different classifiers but the

[2]where *subj* and *pcomp-n* are syntactic functions

best result has been received by using Naïve Bayes approach. In comparison to *run2*, both, recall and precision are higher. Recall equals 70% and precision increases to 57,5%. This run can be considered as a trade-off between two first runs, where recall was high but precision low (*run2*) and in an opposite way (*run1*). The overall accuracy in *run3* is slightly higher than in *run1*. Further investigation shows that, in comparison to *run1*, *run3* provides the same accuracy for *QA* and *SUM* topics, higher accuracy for *IE* topic (46,50%) and lower accuracy for the *IR* topic (60,00%).The precision and recall plots are given on Fig.3 and Fig.4, respectively.

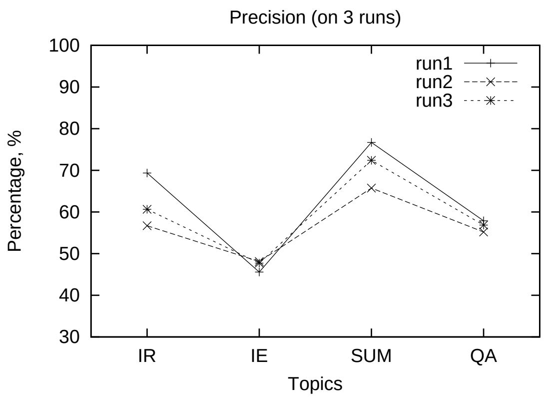

Figure 3: Precision

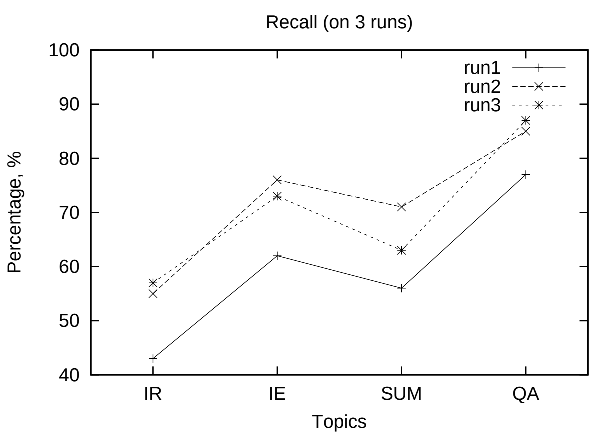

Figure 4: Recall

Besides this, the results obtained on the test set correspond to the 10-cross fold validation results on the training data (precision: 60,3%, recall: 69,1%). Interestingly, no matter how the classifiers have been trained and how two components (lexical and syntactic) have been combined, the highest recall reaches 72% only. This suggests that lexical and syntactic components have limitations and should possibly be bridged with a deeper semantic analysis.

As reported by (Milen Kouylekov et al., 2005) who also have used dependency parsing, the results can be affected by accuracy of parsing. In some cases a sentence with a complex structure leads not to one rooted tree but more. We encountered two such cases in the training set and none of them in the test data.

5 Conclusions

For the Textual Entailment challenge we have used embedded tree mining to solve textual entailment task. There are several ways to extend this approach in future. First, it is possible to modify syntactic matching component. Although the method we proposed performs a relaxed matching of trees, it is in some cases too restrictive since it does not make use of the additional semantic information (e.g., synonyms).

Our analysis shows that, when using syntactic matching, precision and accuracy increase but recall drops. According to our assumptions, one reason for this is the use of the method producing *ordered* embedded trees. Indeed, such pairs of sentences as below do not receive high similarity scores and can be sometimes misclassified.

(h) The currency used in China is the Renminbi Yuan.

(t) The Renminbi Yuan is the currency used in China.

We have used the most general (embedded) subtrees. Although in most cases it allows to filter out possible false positives and to increase precision, there are still negative cases (e.g., id 636) when sentences have very complex structure and the text fragment and hypothesis are matched. In order to avoid matching, additional constraints based on the linguistic information need to be introduced. For instance, the syntactic matching component can take into account the importance of words and syntactic functions (weighted matching).

On the other hand, the system can be expanded by including other components. While examining training and test examples we also noticed that many of them include paraphrases. It has already been shown by (Roy Bar-Heim et al., 2005) that adding paraphrases contributes to recall. Yet another module can make use of the existing resources, such as WordNet. We did not include the latter into our system since our preliminary experiments on the training set suggested that naïve use of WordNet does not contribute to the entailment recognition. However, it might be possible to include it into syntactic matching component.

In our view, it can also be useful to analyze the whole corpus with respect to the linguistic phenomena (e.g. paraphrasing) according to the topics. It can provide insights on the performance of the system on each of the topics in particular. In addition, it is of certain interest to use known information extraction patterns and to analyze whether they can contribute to the system performance on IE topic.

Acknowledgment

This work has been carried out within the Adaptive Information Disclosure project, which is part of the Dutch "Virtual Laboratory for e-science" project (www.vl-e.nl). Authors would like to thank their colleagues from the AID project and from the HCS Lab for valuable discussions. Special thanks to Mohammed Zaki who has made his tree mining tool *TreeMinerD* available.

References

Roy Bar-Heim, Idan Szpektor, and Oren Glickman. 2005. Definition and Analysis of Intermediate Entailment Levels. *Empirical Modeling of Semantic Equivalence and Entailment Workshop at the Annual Meeting of the Association of Computational Linguistics (ACL 2005)*. Ann Arbor, Michigan, USA.

Jesus Herrera, Anselmo Penas, and Felisa Verdejo. 2005. Textual Entailment Recognition Based on Dependency Analysis. *In Proceedings of the PASCAL Challenges Workshop on Recognising Textual Entailment*. Southampton, U.K.

Y. Chi, S. Nijssen, R. Muntz, J. N. Kok. 2004. Frequent Tree Mining - An Overview. *In Fundamenta Informaticae*

Milen Kouylekov, and Bernardo Magnini. 2005. Recognizing Textual Entailment with Tree Edit Distance Algorithms. *In Proceedings of the PASCAL Challenges Workshop on Recognising Textual Entailment*. Southampton, U.K.

Maria Tereza Pazienza, Marco Pennacchiotti, and Fabio Massimo Zanzotto. 2005. Textual Entailment as Syntactic Graph Distance: a rule based and a SVM based approach. *In Proceedings of the PASCAL Challenges Workshop on Recognising Textual Entailment*. Southampton, U.K.

Lucy Vanderwende, Deborah Coughlin, and Bill Dolan. 2005. What Syntax can Contribute in Entailment Task. *In Proceedings of the PASCAL Challenges Workshop on Recognising Textual Entailment*. Southampton, U.K.

Ian H. Witten and Eibe Frank. 2005 Data Mining: Practical machine learning tools and techniques. *Morgan Kaufmann*. San Francisco. 2nd Edition.

Mohammed J. Zaki. 2005. Efficiently Mining Frequent Trees in a Forest: Algorithms and Applications. *In IEEE Transaction on Knowledge and Data Engineering, special issue on Mining Biological Data*