The contribution of the University of Alicante to AVE 2007

Óscar Ferrández, Daniel Micol, Rafael Muñoz, and Manuel Palomar
Natural Language Processing and Information Systems Group
Department of Computing Languages and Systems
University of Alicante
San Vicente del Raspeig, Alicante 03690, Spain
{ofe, dmicol, rafael, mpalomar}@dlsi.ua.es

Abstract

In this paper we discuss a system used to recognize entailment relations within the AVE framework. This system creates representations of text snippets by means of a variety of lexical measures and syntactic structures. Once these representations have been created, we compare the corresponding to the text and to the hypothesis and we try to determine if there is an entailment relation between the text and the hypothesis. The hypotheses have been generated by merging the answers with their corresponding questions, applying a set of regular expression aimed at this issue. In the performed experiments our system obtained a maximum F-measure score of 0.40 and 0.39 for the development and test English corpora, respectively.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: H.3.3 Information Search and Retrieval

General Terms

Algorithms, Semantic Similarity, Experimentation, Measurement, Performance

Keywords

Question Answering, Answer Validation, Recognizing Textual Entailment, Lexical Similarity, Syntactic Trees

1 Introduction

The Answer Validation Exercise (AVE) is a two-year-old track within the Cross-Language Evaluation Forum (CLEF) 2007. AVE provides an evaluation framework for answer validations in Question Answering (QA) systems. This automatic answer validation would be useful for improving the performance of QA systems, helping humans in the assessment of QA systems output and developing better criteria for collaborative ones.

Systems must emulate human assessment of QA responses and decide if an answer to a question is correct or not according to a given text. This year, the participants receive a set of triplets (Question, Answer, Supporting Text) and they must return a boolean value for each triplet showing whether the answer is supported by the text. This shows that the AVE task is very related to the recognition of textual entailments, since it can be considered as a kind of such relations.

With our participation, we want to evaluate our system within the very realistic environment that AVE provides. In addition, AVE boots the direct applicability of our system in the field of QA,

which is very appealing. Our system is designed to recognize textual entailment relations. In fact, we have participated in ACL-PASCAL Third Recognising Textual Entailment (RTE) Challenge [3] this year [2]. To apply our system to the AVE competition we had to do some adjustments that will be explained in detail later.

The remainder of this paper is structured as follows. The following section presents our approach for our participation in AVE. Third section illustrates the experiments carried out and the results obtained. Finally, fourth section shows the conclusions and proposes future work based on our actual research.

2 The AVE Approach

The proposed approach attempts to detect when the text, which could be consider as a passage returned by a QA system, entails or implies the answer given and, if this occurs, the answer is then justified. To determine if this relation appears, our approach will detect lexical and syntactic implications between two text snippets (the text or the passage and the hypothesis that will be created by both the question and the answer). We propose several methods that mainly rely on lexical and syntactic inferences in order to address the recognition task. Next subsections summarize the procedure followed to apply our approach to AVE.

2.1 Corpora Processing

The corpora provided by the AVE organizers has the following format:

```
<q id=1 lang=EN>
  <q_str>Who was Yasser Arafat?</q_str>
  <a id=1 value=XXX>
    <a_str>Palestine Liberation Organization Chairman</a_str>
    <t_str doc=XXX>President Clinton appealed ... </t_str>
  </a>
  <a id=2 value=XXX> ... </a>
  ....
</q>
```

where each question (tag `q`) contains a string (`q_str`), which is the question formulated in natural language. In addition, `q` can have one or more answers and each answer (`a_str`) is associated with a text (`t_str`) that will be required to determine if the answer is entailed to the question.

Since our system is designed to determine implications between two text snippets, the best way to adapt the AVE corpus to our system seems to be the following: for each answer and question, convert them into an affirmative sentence and detect if there is entailment with its associated text.

Therefore, we generated a set of regular expressions to manage these situations. Table 1 shows such regular expressions together with the number and percentage of solved question-answer pairs. AVE organizers provide a set of patterns intended for this purpose, although we used our own due to when these patterns were published we have already adapted our system to the aforementioned regular expressions.

Our system was applied to the English corpora from AVE. These corpora contain 1121 question-answer pairs for the development corpus and 202 for the test one.

Finally, to complete the explanation of the corpora processing, we would like to mention what occurs when a pair does not match any of the generated regular expressions. We propose two solutions: in the first one, called *automatic*, the tokens of the answer are linked together with the tokens of the corresponding question, while for the second solution, called *semi-automatic*, we have done a review of these pairs manually creating the affirmative sentences.

Regular expression	Number of Q-A pairs solved (%)	
	English development corpus	English test corpus
(What) (\S+) (.+)	350 (31.22%)	92 (45.54%)
(Which) (\S+) (\S+) (.+)	165 (14.72%)	10 (4.95%)
(Who) (\S+) (.+)	179 (15.97%)	36 (17.82%)
(Where) (\S+) (.+)	76 (6.78%)	8 (3.96%)
(How many) (\S+) (.+)	96 (8.56%)	14 (6.93%)
(How much) (.+)	12 (1.07%)	0 (0.0%)
Total	878 (78.32%)	160 (79.21%)

Table 1: Regular expressions used to convert questions and answers into affirmative sentences.

2.2 The Core of the System

The core of our system is composed of two modules, each of which attempts to recognize the textual entailment relation from different perspectives, which are the lexical one and the syntactic. In this section we will describe both of them in little detail. For further information, please refer to [1] and [6].

2.2.1 Lexical module

The performance of this method relies on the computation of a wide variety of lexical measures, which basically consists of overlap metrics. Some researchers have already used this kind of metrics [7]. However, our approach does not use any semantic knowledge.

Prior to the calculation of the measures, all texts and the hypotheses created merging the question-answer pairs by means of regular expressions are tokenized and lemmatized. Later on, a morphological analysis is performed as well as a stemmization. Once these steps are completed, we create several data structures that contain the tokens, stems, lemmas, functional[1] words and the most relevant[2] ones corresponding to the text and the hypothesis. The lexical measures will be applied over these structures and this will allow us to know which of them are more suitable to recognize entailment relations. The followings paragraphs describe the lexical measures implemented in our system.

- **Simple matching**: word overlap between text and hypothesis is initialized to zero. If a word in the hypothesis appears also in the text, an increment of one unit is added. The final weight is normalized dividing it by the length of the hypothesis.

- **Levenshtein distance**: it is similar to simple matching. However, in this case we use the mentioned distance as the similarity measure between words. When the distance is zero, the increment value is one. On the other hand, if such value is equal to one, the increment is 0.9. Otherwise, it will be the inverse of the obtained distance.

- **Consecutive subsequence matching**: this measure assigns the highest relevance to the appearance of consecutive subsequences. In order to perform this, we have generated all possible sets of consecutive subsequences, from length two until the length in words, from the text and the hypothesis. If we proceed as mentioned, the sets of length two extracted from the hypothesis will be compared to the sets of the same length from the text. If the same element is present in both the text and the hypothesis set, then a unit is added to the accumulated weight. This procedure is applied to all sets of different length extracted from the hypothesis. Finally, the sum of the weight obtained from each set of a specific length is normalized by the number of sets corresponding to this length, and the final accumulated

[1] As functional words we consider nouns, verbs, adjectives, adverbs and figures (number, dates, etc).
[2] Considering only nouns and verbs.

weight is also normalized by the length of the hypothesis in words minus one. One should note that this measure does not consider non-consecutive subsequences. In addition, it assigns the same relevance to all consecutive subsequences with the same length. Furthermore, the more length the subsequence has, the more relevant it will be considered.

- **Tri-grams**: two sets containing tri-grams of letters belonging to the text and the hypothesis were created. All the occurrences in the hypothesis' tri-grams set that also appear in the text's will increase the accumulated weight in a factor of one unit. The calculated weight is then normalized dividing it by the total number of tri-grams within the hypothesis.
- **ROUGE measures**: ROUGE measures have already been tested for automatic evaluation of summaries and machine translation [4]. For this reason, and considering the impact of n-gram overlap metrics in textual entailment, we believe that the idea of integrating these measures[3] in our system is very appealing. We have implemented these measures as defined in [4].

In order to detect entailment relations, several machine learning classifiers were considered, being the Bayesian Network the best for our needs. We have used the Bayesian Network implementation from Weka [9], considering each lexical measure as a feature for the training and test stages of our system.

2.2.2 Syntactic module

This module aims to provide a good accuracy rate by using few syntactic modules that behave collaboratively. These include tree construction, filtering and graph node matching.

- **Tree generation**: the first module constructs the corresponding syntactic dependency trees. For this purpose, *MINIPAR* [5] output is generated and afterwards parsed for each text and hypothesis of our corpus. Phrase tokens, along with their grammatical information, are stored in an on-memory data structure that represents a tree, which is equivalent to the mentioned syntactic dependency tree.
- **Tree filtering**: once the tree has been constructed, we may want to discard irrelevant data in order to reduce our system's response time and noise. For this purpose we have generated a database of relevant grammatical categories (see Table 2) that will allow us to remove from the tree all those tokens whose category does not belong to such list. The resulting tree will have the same structure as the original, but will not contain any stop words nor irrelevant tokens, such as determinants or auxiliary verbs.
- **Graph node matching**: in this stage we proceed to perform a graph node matching process, termed alignment, between both the text and the hypothesis[4]. This operation consists in finding pairs of tokens in both trees whose lemmas are identical, no matter whether they are in the same position within the tree. Some authors have already designed similar matching techniques, such as the ones described in [8]. However, these include semantic constraints that we have decided not to consider. The reason of this decision is that we desired to overcome the recognition task from an exclusively syntactic perspective.

  Let τ and λ represent the text's and hypothesis' syntactic dependency trees, respectively. We assume we have found a word, namely β, present in both τ and λ. Now let γ be the weight assigned to β's grammatical category (Table 2), σ the weight of β's grammatical relationship (Table 3), μ an empirically calculated value that represents the weight difference between tree levels, and δ_β the depth of the node that contains the word β in λ. We define the function $\phi(\beta) = \gamma \cdot \sigma \cdot \mu^{-\delta_\beta}$ as the one that calculates the relevance of a word in our system. The experiments performed reveal that the optimal value for μ is 1.1.

[3] The considered measures were ROUGE-N with n=2 and n=3, ROUGE-L, ROUGE-W and ROUGE-S with s=2 and s=3.

[4] One should remember that the hypothesis has been created from the pair question-answer by means of regular expressions (see section 2.1)

Grammatical category	Weight
Verbs, verbs with one argument, verbs with two arguments, verbs taking clause as complement	1.0
Nouns, numbers	0.75
Be used as a linking verb	0.7
Adjectives, adverbs, noun-noun modifiers	0.5
Verbs *Have* and *Be*	0.3

Table 2: Weights assigned to the relevant grammatical categories.

Grammatical relationship	Weight
Subject of verbs, surface subject, object of verbs, second object of ditransitive verbs	1.0
The rest	0.5

Table 3: Weights assigned to the grammatical relationships.

For a given pair (τ, λ), we define the set ξ as the one that contains all words present in both trees, being $\xi = \tau \cap \lambda \ \ \forall \alpha \in \tau, \beta \in \lambda$. Therefore, the similarity rate between τ and λ, denoted by the symbol ψ, would be $\psi(\tau, \lambda) = \sum_{\nu \in \xi} \phi(\nu)$. One should note that a requirement of our system's similarity measure would be to be independent of the hypothesis length. Thus, we must define the normalized similarity rate, as $\overline{\psi(\tau, \lambda)} = \frac{\sum_{\nu \in \xi} \phi(\nu)}{\sum_{\beta \in \lambda} \phi(\beta)}$. Once the similarity value has been calculated, it will be provided to the user together with the corresponding text-hypothesis pair identifier. It will be his responsibility to choose an appropriate threshold that will represent the minimum similarity rate to be considered as entailment between text and hypothesis. All values that are under such a threshold will be marked as not entailed. The development corpus will help us to establish this threshold properly.

3 Experiments and Results

In AVE, all pairs must be tagged with one of the following values:

- VALIDATED indicates that the answer is correct and supported although not the one selected.
- SELECTED indicates that the answer is VALIDATED and it is the one chosen as the output of an hypothetical QA system. One of the VALIDATED answers per question should be marked as SELECTED.
- REJECTED indicates that the answer is incorrect or there is not enough evidence of its correctness.

Since our system returns a numeric value to determine the entailment, we decided to mark as SELECTED the pair with the highest true entailment score among all pairs that belong to the same question. If it is the case that two or more pairs have the highest score, then one of them is randomly chosen.

Regarding the framework that the AVE organizers propose to evaluate the systems, apart from the well-known measures of Precision, Recall and F over the YES pairs[5], we would like to point

[5]The YES pairs are those which are considered as VALIDATED or SELECTED.

out a new measure, called Q-A accuracy. This measure only considers the accuracy obtained from correct SELECTED values and attempts to simulate the decision that could be made by a QA system. However, for our system it is quite difficult to establish one of the VALIDATED values as a SELECTED since differences between true entailment scores are usually minimal. This happens due to the fact that no semantic knowledge is considered. Therefore, although the system is able to determine lexical and syntactic implications, in the case of SELECTED values this does not seem to be enough.

Table 4 shows the different experiments carried out and the results obtained for our system. The proposed baseline was generated setting all pairs as VALIDATED, which was useful to evaluate the gain of the remainder experiments.

Corpus	**Run**	**Prec. YES**	**Rec. YES**	**F-measure**	**Q-A acc.**
Development	baseline	0.12	1.0	0.21	–
	lex automatic	0.26	0.78	0.39	–
	lex semi-automatic	0.27	0.78	0.40	–
	syn automatic	0.31	0.03	0.06	–
	syn semi-automatic	0.17	0.17	0.17	–
Test	baseline	0.11	1.0	0.19	–
	lex semi-automatic	0.25	0.81	0.39	0.18
	syn semi-automatic	0.18	0.81	0.29	0.19

Table 4: Results obtained for the AVE 2007 track.

Two main experiments were carried out for our participation in AVE. The first one applies the lexical module to detect VALIDATED and SELECTED pairs, whereas the second one only uses syntactic information (obtained by the syntactic module) to solve implications. These runs were named *lex* and *syn* respectively. A simple combination of both modules, for instance to decide the judgment depending on the accuracy of each one for true and false implications, does not improve the results. Therefore we believe that subsequent work could be the combination of these modules in a collaborative way rather than by means of other simpler techniques.

Moreover, each run (*lex* or *syn*) was processed with the two types of corpus, *automatic* and *semi-automatic*, created from the original AVE corpora (see section 2.1). Table 4 reveals that, although the semi-automatic experiments obtain better results, the effort needed to generate this corpus is not worth in comparison with the gain of accuracy obtained.

The approach that achieved better results is *lex*. This is due to the fact that there are some cases where the hypothesis' construction does not make sense and consequently the syntactic tree is incorrectly generated. These situations occur when the answer has a grammatical category inconsistent with the one expected by the question (for instance, if the answer is a quantity or date when the question asks for a person name).

4 Conclusions and Future Work

This paper presents two independent approaches considering mainly lexical and syntactic information. Throughout this paper we expose and analyze a wide variety of lexical measures as well as syntactic structure comparisons that attempt to recognize the textual implications required for the AVE task.

The approach that obtained the best results was the lexical one, being the optimal for our participation, and obtaining an F-measure score of 0.40 and 0.39 for the development and test corpus, respectively. However, we would like to point out that the results obtained in challenges or competitions about recognizing entailment relations depend on the idiosyncrasies of the corpora used. For instance, whereas AVE generates its corpora directly from the output of several QA systems, the RTE challenge constructs the corpora by means of a review process of several anno-

tators and from different sources (see RTE-3 overview [3] and our participation in this challenge [2]).

Future work can be related to the development of a semantic module. This module will be able to construct characterized representations based on the text using named entities and role labeling in order to extract semantic information from the pair of text snippets. In addition, once the semantic module was implemented, subsequent work will be to combine these modules in an efficient way. Each module should perform the recognition individually as well as support it together with the rest of the modules.

Acknowledgments

This research has been partially funded by the QALL-ME consortium, which is a 6^{th} Framework Research Programme of the European Union (EU), contract number FP6-IST-033860 and by the Spanish Government under the project CICyT number TIN2006-1526-C06-01. It has also been supported by the undergraduate research fellowships financed by the Spanish Ministry of Education and Science, and the project ACOM06/90 financed by the Spanish Generalitat Valenciana.

References

[1] Óscar Ferrández, Daniel Micol, Rafael Muñoz, and Manuel Palomar. DLSITE-1: Lexical analysis for solving textual entailment recognition. In *Proceedings of the 12th International Conference on Applications of Natural Language to Information Systems*, Paris, France, June 2007. Springer.

[2] Óscar Ferrández, Daniel Micol, Rafael Muñoz, and Manuel Palomar. A perspective-based approach for solving textual entailment recognition. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 66–71, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[3] Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. The third pascal recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 1–9, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[4] Chin-Yew Lin. ROUGE: A Package for Automatic Evaluation of Summaries. In Stan Szpakowicz Marie-Francine Moens, editor, *Text Summarization Branches Out: Proceedings of the Association for Computational Linguistics Workshop*, pages 74–81, Barcelona, Spain, July 2004.

[5] Dekang Lin. Dependency-based Evaluation of MINIPAR. In *Workshop on the Evaluation of Parsing Systems*, Granada, Spain, 1998.

[6] Daniel Micol, Óscar Ferrández, Rafael Muñoz, and Manuel Palomar. DLSITE-2: Semantic similarity based on syntactic dependency trees applied to textual entailment. In *Proceedings of the TextGraphs-2 Workshop*, pages 73–80, Rochester, New York, United States of America, April 2007. The North American Chapter of the Association for Computational Linguistics.

[7] Jeremy Nicholson, Nicola Stokes, and Timonthy Baldwin. Detecting Entailment Using an Extended Implementation of the Basic Elements Overlap Metrics. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, pages 122–127, Venice, Italy, April 2006.

[8] Rion Snow, Lucy Vanderwende, and Arul Menezes. Effectively using syntax for recognizing false entailment. In *Proceedings of the North American Association of Computational Linguistics*, pages 33–40, New York City, New York, United States of America, June 2006.

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