The Contribution of FaMAF at QA@CLEF 2008.Answer Validation Exercise

Julio J. Castillo
Faculty of Mathematics Astronomy and Physics
National University of Cordoba, Argentina
cj@famaf.unc.edu.ar

Abstract. The system utilizes a Recognizing Textual Entailment (RTE) approach. The system uses the question string (q_str) as a Text (T) and one answer (t_str) as a Hypothesis (H). We use two different approaches using machine learning, specifically Support Vector Machine as classifier. The results show an increment over the baselines, however enhanced is needed. The features used are unigram, bigram, and trigram overlap of lexemes and stems, Levenshtein distance, tf-idf measure, and semantic similarity using wordnet.
Experimental results show that the best run of our initial system achieved a 0.21 of F-measure and 0.17 of QA-accuracy. This shows an increment of 23.53% over the QA accuracy baseline.

Categories and Subject Descriptors
H.3 [**Information Storage and Retrieval**]: H.3.3. Information Search and Retrieval; H.3.4 Systems and Software; H.3.7 Digital Libraries

General Terms
Algorithms, Measurement, Experimentation

Keywords
Question Answering, Answer Validation, Recognizing Textual Entailment, QA@CLEF

1. Introduction

The objective of the Answer Validation Exercise (AVE) 2008 is to develop systems able to decide if the answer to a question is correct or not [5]. This is a three years old track and is part of Cross Language Evaluation Forum (CLEF) 2008.

AVE challenge is an evaluation framework for Question Answering (QA) systems. The inputs for the AVE systems are a set of triplets (Question, Answer, Supporting Text) and the results are a boolean value indicating whether the answer is supported by the text [6]. Answer Validation task must select the best answer for the final output. Therefore, AVE task is very similar to RTE (Recognition of Textual Entailments).

We have developed a system that performs morphological analysis (stemming, and POS tagging), and extract lexical (uni-bi and trigram overlap, Levenshtein, and tf-idf), and semantic measures using Wordnet[1] to build a model using a Support Vector Machine (SVM [4]), to determinate whether the implication holds.

2. System description

We have developed a system and carried out two runs. Both are based on a supervised learning approach using a SVM. One of this consists of twelve features, and the other takes in account only three features. Figure 1 shows the general architecture of our system. Similar to [7].

[1] wordnet.princeton.edu/

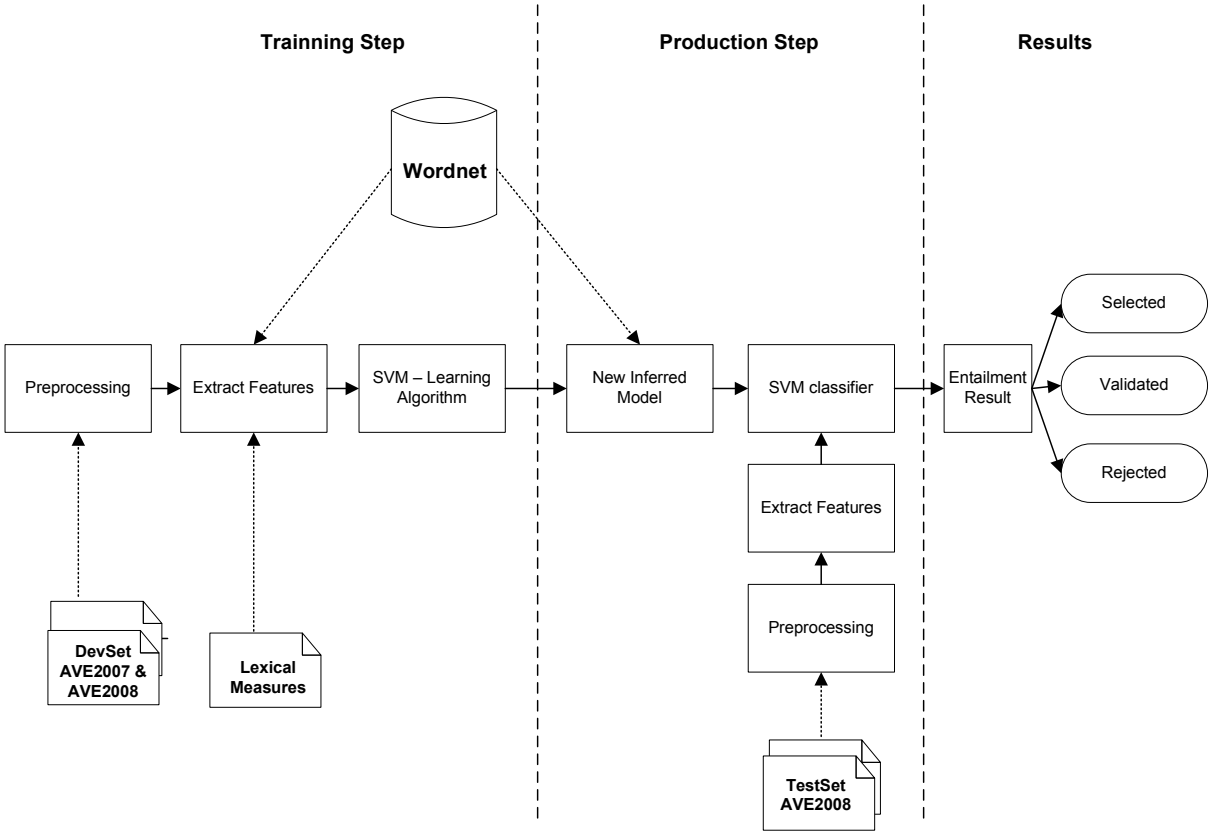

Figure 1. Our System for English Answer Validation

Two different runs were submitted to AVE 2008 Challenge, and the experiments shows that our second run achieved a 0.21 F-measure, and that outperformed the standard baseline by 0.7 points. The first run is based on 12 feature vectors, and our second run is based on three features.

In order to extract all of these features, the system employs different tools such as a stemmer, a POS tagger, and Wordnet. The run one, is based on lexical overlaps and is quite simple, and his results are mayor that baselines, however we think that is not very competitive. This system approach is based on determinate if H (Hypothesis) is entailed by T (Text) considering only lexical similarities. The procedure for both runs will be described in sections 2.1 and 2.2.

2.1 Run 1: Lexical similarity

The main idea is to extract a set of lexical measures to determinate the similarity between (Hypothesis, Text) pairs. Our lexical approach is similar to [1] and [2].

We don't preprocess the pairs using English stopwords list, because this process could remove important information of question, such as "the type" of question.

We have used an implementation of Porter stemmer[2] [3] to obtain the stems of tokens. The bag of stems is used for process the last six features.

Follow the first run features are depicted:

1. Percentage of word of the hypothesis in the text (treated as bags of words).

2. Percentage of word of the text in the hypothesis (treated as bags of words).

3. Percentage of bigrams of the hypothesis in the text (treated as bags of words).

[2] http://tartarus.org/~martin/PorterStemmer/

4. Percentage of trigrams of the hypothesis in the text (treated as bags of words).

5. TF-IDF + Cosine measure between the text and hypothesis.

6. Levenshtein[3] distance between T and H (over words).

7. Percentage of word of the hypothesis in the text (treated as bags of stems).

8. Percentage of word of the text in the hypothesis (treated as bags of stems).

9. Percentage of bigrams of the hypothesis in the text (treated as bags of stems).

10. Percentage of trigrams of the hypothesis in the text (treated as bags of stems).

11. TF-IDF + Cosine measure between the text and hypothesis (treated as bags of stems).

12. Levenshtein distance between T and H (over stems).

TF-IDF measure is calculated over T and H's that are thought as little documents. One of this is q_str and the others are t_str. The value of this feature is between 0 and 1.The similarity of both documents is captured by using the cosine similarity measure. This measure is performed measuring the cosine of the angle for the two documents vectors.

Levenshtein distance between T and H is the minimum numbers of operations (edit distance) that are needed to transform one string T, into the other string H, using only insertion, deletion, or substitution of a single character. We think that the system should be able of insert, delete or substitute entire word or stems, instead of characters. It will be our next step.

2.2 Run 2: Lexical and Semantic similarity

The second run that we have submitted uses only three features. The first step to preprocess these pairs is to obtain the stems of tokens. We have used OpenNLP[4] to obtain pos-tagging. Additionally we have used Wordnet[5] as tool for determinate synonym and hypernonym relationship.
The features used for the second run are the followings:

1. Levenshtein distance between T and H (over stems). Identical to feature 6 in Run1.

2. Lexical similarity using Levenshtein distance:

3. Semantic similarity using Wordnet

The steps necessary to obtain lexical similarity using Levenshtein distance are:
- Each string T and H are divided in a list of tokens.
- The similarity between two different tokens is performed using the "string edit-distance matching"(Levenshtein distance). This is for all tokens.
- The string similarity between two list of tokens is reduced to the problem of "bipartite graph matching".

Therefore, it is performed using the Hungarian algorithm over this bipartite graph.
After, we find the optimal assignment that maximizes the sum of ratings of each token. Note that each graph node is a token of the list.

Finally the final score is calculated by:

[3] http://www.let.rug.nl/~kleiweg/lev/levenshtein.html
[4] http://opennlp.sourceforge.net/
[5] wordnet.princeton.edu/

$$finalscore = \frac{TotalSim}{Max(Lenght(T), Lenght(H))}$$

Where:

TotalSim: is the sum of the similarities with the optimum assignment (using Hungarian algorithm).
The maximum value of TotalSim is equal to Max (Length (Text), Length (H)).

Length (T): is equal to the quantity of tokens that the graphs have in his origin component.
Length (H): is equal to the quantity of tokens that the graphs have in his destine component.

Wordnet is used to calculate the semantic similarity of two strings. Given two tokens, they are used only in synonym and hypernonym relationship (is a type of relation). After, BFS (Breadth First Search) algorithm is used over these tokens, and then if two words are found, his similarity is calculated using two factors: length of the path, and orientation of the path.

The required steps are the followings:
3.1. Tokenization
3.2. Stemming words
3.3. POS tagging (to compare the word senses using the same POS. Therefore only are compared words in the same taxonomy).
3.4. WSD is performed using the Lesk algorithm, based on Wordnet definitions.
3.5. A semantic similarity matrix is defined.
The semantic similarity is computed by:

$$Sim(s,t) = 2 \times \frac{Depth(LCS(s,t))}{Depth(s) + Depth(t)}$$

Where:
- s,t : are source and target words that we are comparing(s is the Hypothesis and t is the Text).
- Depth(s): Is the shortest distance between root node to current node (in the local taxonomy).
- LCS(s,t): is the least common sub-summer of “s” and “t”.

3.6. A bipartite graph is building and computed using Hungarian Algorithm.
3.7. To obtain the final score, we use “matching average”, i.e.:

$$MatchingAverage = 2 \times \frac{Match\ (X,Y)}{Length\ (X) + Length\ (Y)}$$

Where:
- X and Y are two sentences, particularly are T and H.

3. Experimental Evaluation

3.1. Training and Test Sets

The development set available for the AVE 2007 English task consists of 1121 answers, where 11,59% are validated answers and the rest 88.41% are rejected. On the other hand, the AVE 2008 development set consists of 195 pairs, only 21 are positives, it is 10,77% of the total.

Our system was sensitive to this unbalanced training set, so we built a balanced set, with approximately the same number of TRUE and FALSE pairs. We took all TRUE pairs from the training sets in AVE 2006 and AVE 2007 and then we incorporated a number of FALSE pairs totalling a 40% of the total. We have testing with 10%, 20%, 40%, 50%, and 60% of negatives of the total, and the best development set result was with 40% of FALSE.

3.2. Analysis of Results

The described system has been tested in English using training sets of AVE 2006 and AVE 2007. The tables 1 and 2 show the recall, precision and f-measure over correct answers that were obtained with Run 1 and Run 2. Two baseline systems return VALIDATED for 100% and 50% of answers in the test set.
The results were a relative high qa_accuracy and an acceptable f-measure.

Language: English

	F	**Precision**	**Recall**
RUN 2	**0.21**	**0.13**	0.56
RUN 1	0.17	0.09	**0.94**
100% VALIDATED	0.14	0.08	1
50% VALIDATED	0.13	0.08	0.5

Table 1. General evaluation of the Famaf's system

The results obtained by both systems have been better than the baselines, achieving the Run 1 a high recall.
The Run 1 represents a very positivist approach because the systems highly classify an (T,H) pair as a positive entailment.

Regarding the false positives, we can see that the Levenshtein distance is the best of the twelve features, because is the most differentiable among the others, and helps the classifier to classify correctly.
The first 6 features (over bag of words) are more descriptive that the last 6 features (over stems). However, his difference is not significant.

Table 2 shows the results obtained for the two runs, compared with the value obtained in a perfect selection, best QA system, and a baseline system that validates all answers and randomly select one of them. This measure is used for compare the AV systems with QA systems presented in QA@CLEF.

Baselines for comparing AV systems performance with QA systems in English.

	estimated_qa_performance	**qa_accuracy**
Perfect selection	0.56	0.34
Best QA system	0,21	0,21
RUN 2	**0.16**	**0.17**
RUN 1	0.16	0.16
random	0.09	0,09

Table 2. Evaluation results obtained by the qa-accuracy measure

4. Conclusion and Future Work

We presented to AVE 2008 our initial RTE system that is based on machine learning approach using SVM as classifier. We have used lexical features such as unigram, bigram, and trigram overlap of lexemes and stems, and also we have used Levenshtein distance, tf-idf measure, and semantic similarity with Wordnet.
Experimental results show that the best run of the system achieved a 0.21 of F-measure and 0.17 of QA-accuracy, an increment of 23.53% over the QA accuracy baseline. In spite of the simplicity of the approach, we have obtained a reasonable 0.17 of QA accuracy for the second run.

Future work is oriented to probe with different classifiers as Bayesian Binary Regression (BBR), and use different datasets RTE, and RTE+AVE. To enhance the system, we will work with lexical and semantic similarity, adding features and testing his improvement.
Additionally an NER module will be incorporated and combined with the rest of the system and his performance will be evaluated.

References

[1] Alvaro Rodrigo, Anselmo Peñas, Jesus Herrera, Felisa Verdejo. *Experiments of UNED at the Third Recognizing Textual Entailment Challenge*.Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing. 2007.

[2] Prodromos Malakasiotis and Ion Androutsopoulos. *Learning Textual Entailment using SVMs and String Similarity Measures*. ACL-PASCAL Workshop on Textual Entailment and Paraphrasing, 45th Annual Meeting of the Association for Computational Linguistics (ACL 2007), Prague, Czech Republic, 2007.

[3] Julie Beth Lovins. *Development of a stemming algorithm*. Mechanical Translation and Computational Linguistics, March 1968.

[4] Corinna Cortes and V. Vapnik, *Support-Vector Networks*, Machine Learning, *20, 1995.*

[5] Peñas A., Rodrigo A., Sama V., and Verdejo F. *Overview of the Answer Validation Exercise 2006*, In Working notes for the Cross Language Evaluation Forum Workshop (CLEF 2006), Alicante, España, September 2006.

[6] Peñas A., Rodrigo A., Sama V., and Verdejo F. *Overview of the Answer Validation Exercise 2007*, In Working notes for the Cross Language Evaluation Forum Workshop (CLEF 2007), Budapest, Hungary, September 2007.

[7] M.A. García-Cumbreras, J. M. Perea-Ortega, F. Martínez Santiago, L.A. Ureña-López. *SINAI at QA@CLEF2007. Answer Validation Exercise*, In Working notes for the Cross Language Evaluation Forum Workshop (CLEF 2007), Budapest, Hungary, September 2007.