Wild Devs' at SemEval-2017 Task 2: Using Neural Networks to Discover Word Similarity

Răzvan-Gabriel Rotari, Ionuț Hulub, Ștefan Oprea, Mihaela Plămadă-Onofrei, Alina Beatrice Lorenț, Raluca Preisler, Adrian Iftene, Diana Trandabăț

University Alexandru Ioan Cuza of Iași, Romania

{razvan.rotari, ionut.hulub, stefan.oprea, mihaela.onofrei, alina.lorent, raluca.preisler, adiftene, dtrandabat}@info.uaic.ro

Abstract

This paper presents Wild Devs' participation in the SemEval-2017 Task 2 "Multilingual and Cross-lingual Semantic Word Similarity", which tries to automatically measure the semantic similarity between two words. The system was build using neural networks, having as input a collection of word pairs, whereas the output consists of a list of scores, from 0 to 4, corresponding to the degree of similarity between the word pairs.

1 Introduction

The Wild Dev's team participated this year in SemEval 2017 Task 2, subtask 1, in the evaluation for the English language. The system is based on a neural network, trained on an enriched corpus of word pairs.

The paper is structured in 4 sections: this section discusses existing approaches to similarity using word embedding, before presenting the architecture of our system in Section 2. The next section briefly analyses the results, while Section 4 drafts some conclusions and further work.

In natural language processing, one of the most important challenges is to understand the meaning of words.

The organizers of Task 2 (Task2, 2017) state that this task "provides a reliable benchmark for the development, evaluation and analysis" of:

• Word embeddings, monolingual word embeddings, as well as bilingual and multilingual word embeddings which have a unified semantic space for the languages;

- Similarity measures that use lexical resources;
- Supervised systems that combine multiple measures.

Our initial option was word embedding. The most prominent word embedding software tools are:

1. Word2Vec (Mikolov et al., 20013) is an algorithm with the explicit goal of producing word embeddings that encode general semantic relations (Collobert et al., 2011).

2. GloVe (Pennington et al., 2014) has a similar aim as Word2Vec. Its authors present GloVe mainly as an unsupervised learning algorithm, also offering an implementation.

3. Deeplearning4j is an open source deep learning library for Java which implements both Word2Vec and GloVe, among other algorithms (Deeplearning4j, 2017).

4. Principal Component Analysis (Jolliffe, 2002) and T-Distributed Stochastic Neighbor Embedding (van der Maaten 2008) are two algorithms that reduce the dimensionality of already generated word embedding vectors.

After initial tests using the data provided by the task organizers, we realized that it would yield better results to aggregate multiple techniques, and thus resolved to use a supervised system which combines multiple techniques. Recent studies show that neural-network-inspired word embedding models exceed the traditional countbased distributional models on word similarity. The supervised system we propose is a neural network trained on a list of gold standard word pairs. Three individual models (word embedding, definition comparison, synonyms, detailed in the next section) are run on the collection of word pairs and different lists of scores are obtained. The neural network is trained by comparing the lists of scores to the gold standard.

In order to develop a gold standard, we collected a set of word pairs and designed a web site that allowed human annotators to rate each word pair on the scale of 0 to 4. The users were students from the Faculty of Letters, who thus had a proficient and professional knowledge of the English language.

2 Methodology

Nowadays, there is a huge mass of textual data in electronic format, and this fact increased the need for fast and accurate techniques for textual data processing. Despite the evolution of the field, evaluation still rely (most of the time) on a comparison between the output of a statistical system on the one hand, and a hand-crafted gold standard, on the other hand. Generally, a gold standard provides an interesting basis for the comparison of systems against the same set of data, or for the comparison of the evolution of the performance of the different versions of a system performing a certain task.

2.1 Increased Gold Standard

In order to train the neural network, we needed a set of word pairs and a gold standard. The task's website affirmed that a human-generated gold standard is used, therefore we decided to increase its size by building a collection of word pairs manually rated according to their similarity.

For building the list of word pairs, we used Daniel Defoe's novel "Robinson Crusoe"¹. We picked nouns at random from the novel and built pairs (for testing that a word is a noun, we used WordNet). A web interface (figure 1) was developed to allow user to validate the similarity between word pairs. Most word pairs obtained scores between 0 and 2. In order to obtain higher scores (3 and 4), meaning higher similarity, we built some pairs using a noun from the novel and one of the synonyms in its synset from WordNet.

¹ The novel is in the public domain and is available on Project Gutenberg The group of annotators was composed of students from the Faculty of Letters who had proficient and professional knowledge of the English language. To assist their voting, we built a web site (Figure 1) that offers word pairs and saves the votes in a database. We used a session cookie to avoid giving the same word pair to a session twice.



Figure 1. The website for rating word pairs

After the website was up, we added a username field to sessions, as well as a statistics page which collected the total number of votes for a student for all his sessions.

One of our priorities has been the maintenance of the website and quick fixing of bugs, in order to avoid losing data about sessions and the number of votes each student has done. As always when using volunteers, the danger exists that some users could cheat by scoring the word pairs at random. We took the precaution that we kept a record of all votes for a word pair, and thus could single out a suspect-looking vote. In order to insure inter-annotator agreement, each pair of words was evaluated by 5 different users.

The site also contained the original description of the rating scale, from the SemEval website, such that our gold standard would be similar to that used by the organizers.

In this way, we obtained a gold standard of 5747 word pairs, and the distribution of scores is satisfactory. The collection will be made publically available.

After having the corpus, we built our system using neural networks. The architecture of our systems is presented in figure 2.



Figure 2. Architecture

2.2 Word Embeddings

From word vectors, we can find out the similarity of two words using the dot product of their vectors. One drawback is that most existing implementations load all vectors into RAM, which requires gigabytes of RAM. Given word vector files which can be obtained from various word embedding algorithms, our supervised system can use each of those vectors to obtain a score file, and train the neural network.

2.3 Definition Comparison

This module calculates the similarity of two words by comparing their definitions using the Levenshtein distance as a String metric.

An initial check is performed to verify if the words are identical, in which case the score will be 4, or if they are pairs of antonyms formed by derivation from the same root (e.g. hopeful <-> hopeless, legal <-> illegal), in which case the score will be 0. If it is not the case, the program goes on.

For each of the two words, a call is made to the API offered by the Pearson Publishing House.

The result is a JSON which can contain one or more entries, depending on how many meanings the word has. Our program parses the JSON and extracts only the definitions, which are then stored in an array.

Thus we have two arrays of definitions, one for each word, we iterate through the first array and we compare every definition to all the definitions of the other word. For this purpose, we use the Levenshtein distance (the minimum number of single-character edits, i.e. insertions, deletions or substitutions, required to change one word into the other). To calculate the Levenshtein distance we use Java's StringUtils library.

This score is increased by our program if one word includes the other (e.g. flower-sunflower) or if at least one of the words can be found in the definition of the other. From all the scores we get from comparing the definitions, only the biggest will be kept. Because the scores obtained by applying the Levenshtein distance to dictionary entries have very small values (between 0 şi 1.5 out of 100), we process them to get one of the values: 0, 1, 2, 3, 4.

If one or both words cannot be found in the dictionary or if an error occurs during the execution, the returned score is 0.

The program can support a limited number of multi-word expressions and personal names, if they can be found in the Pearson dictionary.

2.4 Synonymy

This module calculates the similarity of two words using Wordnet, by comparing their synsets. For each word pair, the list of synsets is retrieved, and the sets of synonyms are compared two by two in order to count the common words. A score is thus obtained, to be compared to the score given by users (mainly to check the user's credibility). Additionally, the list of word pairs having higher scores has been increased by using synonyms of these words, extracted from Wordnet.

3 Evaluation

We performed an internal evaluation of our system on the training data and obtained a score of 0.372 (Pearson: 0.385, Spearman: 0.357). The results show that we have managed to accomplish the main objective of this project, to outperform the random strategy. The lower scores have been obtained for named entities and multiword expressions, instances which do not exist in our gold standard, for which we plan to add dedicated modules.

Our team participated in task 1 for English, and was officially evaluated with a Pearson score of 0.459 and a Spearman score of 0.477, giving a total of 0.468.

4 Conclusions

This paper explores word similarities by using a supervised system that aggregated corpus based techniques, as well as word embedding techniques. It also exposes the need for more experiments that should be done in this field, and we take into account the possibility to create such a solution for the Romanian language.

In the future, we will refine the components of the supervised system. Given more time, we could get an even larger gold standard using our site, which will allow us to even better train our neural network.

We could also implement word embedding software that efficiently uses hard disk space, ra-

ther than loading all vectors into RAM at once, or use a distributed computing approach.

There are some other aspects that we are interested to tackle in the future, such as named entity recognition and multiword expressions recognition.

References

- Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011). Natural Language Processing (almost) from Scratch. Journal of Machine Learning Research, 12 (Aug), 2493–2537. Retrieved from http://arxiv.org/abs/1103.0398
- Deeplearning4j Development Team. (2017) Deeplearning4j: Open-source distributed deep learning for the JVM, Apache Software Foundation License 2.0. http://deeplearning4j.org
- Jolliffe I.T. (2002) *Principal Component Analysis*, Series: Springer Series in Statistics, 2nd ed., Springer, NY, 2002, XXIX, 487 p. 28 illus. ISBN 978-0-387-95442-4
- Mikolov Tomas, Kai Chen, Greg Corrado, Jeffrey Dean (2013) Efficient Estimation of Word Representations in Vector Space, arXiv:1301.3781.
- Pennington, Jeffrey, (2014) Richard Socher, and Christopher D. Manning. "Glove: Global Vectors for Word Representation." EMNLP. Vol. 14. 2014.
- Jose Camacho-Collados, Mohammad Taher Pilehvar, Nigel Collier and Roberto Navigli (2017) SemEval-2017 Task 2: Multilingual and Crosslingual Semantic Word Similarity. In Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval 2017). Vancouver, Canada.
- van der Maaten, L.J.P.; Hinton, G.E. (2008). Visualizing High-Dimensional Data Using t-SNE. Journal of Machine Learning Research. 9: 2579–2605.