Cross-Lingual ad hoc Semantic Retrieval

Ahmad Altaweel and **Daria Sadova** and **Mohammad Deaa Barni** and **Rafik Takieddin**
University of Mannheim
Schloss
68131 Mannheim, Germany
(aaltwee, dsadova, mbarni, rtakiedd)@mail.uni-mannheim.de

Abstract

Cross-lingual information retrieval (CLIR) is a subfield of information retrieval dealing with retrieving information written in a language different from the language of the user's query. This allows users to search document collections in multiple languages and retrieve relevant information in a form that is useful to them, even when they have little or no linguistic competence in the target languages. In this work we investigate the whole algorithm for CLIR system. To do so, we separated our work for four parts: computation the translation matrix, measuring the similarity between documents, evaluation them and verification our final program with real query and collection of documents. We propose interactive part in our program. The model allows the user to express his information need via a query and the number of documents the user wish to retrieve. Besides, we made toy-collection of documents in both languages to test whether our system works right or not.

1 Introduction

The goal of the project was to implement the model for cross-lingual retrieval, to test it and to evaluate the performance of the model.
Semantic vector space models (VSM) of language represent each word with a real-valued vector. It is a non-supervised approach for computing words representations that gives state-of-the-art results in similarity and analogy tasks. We used the pre-trained 300-dimensional GloVe (Global Vectors) word embeddings, introduced by Pennington in (Pennington et al., 2014) for English[1] and the Spanish Billion Words Corpus and Embeddings provided by (Cardellino, 2016) [2]. The former is a model trained on the non-zero entries of a global word-word co-occurrence matrix, which tabulates how frequently words co-occur with one another in a given corpus. The latter consists of an unannotated corpus of the Spanish language of nearly 1.5 billion words, compiled from different corpora and resources from the web and a set of word embeddings, created from this corpus using the Word2Vec algorithm, provided by the Gensim package. These embeddings were evaluated by translating to Spanish Word2Vecs word relation test set.
We obtained the top 4000 most frequent words in English, translated them into Spanish using Google translator and made pairs between them.
We propose a quantitative evaluation that directly compares and finds precision automatically produced positions with the ground truth positions (i.e., given positions) for our dataset. Furthermore, we constructed a dataset (with documents in both languages), which we scaled using the corresponding queries. The stop-words for English texts included in Python library, the stop-words for Spanish texts provided by (Glavaš et al., 2017) [3]

2 Cross-Lingual Text Scaling

Our scaling approach consists of four components: (1) construction of the translation matrix; (2) calculation the aggregate embedding vectors of documents by computing a weighted sum of embeddings of words; (3) measurement the similarity of documents in different languages by computing the cosine of the angle between their aggregate

[1] https://nlp.stanford.edu/projects/glove/
[2] http://crscardellino.me/SBWCE/
[3] http://dws.informatik.uni-mannheim.de/fileadmin/lehrstuehle/ki/people/goran/cl-scaling.rar

dense vectors; (4) evaluating the performance of the cross-lingual embedding model using standard evaluation metrics: R-precision and mean average precision (MAP).

We start from independent monolingual word embeddings of both English and Spanish representing the words with 300-dimension VSM. In order to allow for semantic comparison of texts in different languages, we must construct a joint multilingual semantic vector space. To this end, we select the embedding space of English and map embeddings of Spanish in one embedding space, given a (relatively small) set of word translation pairs. Given a set of word translations pairs, we learn a translation matrix **M** that projects embedding vectors from one embedding space to the other. **S** and **T** are matrices with monolingual embeddings of source. We opt for a analytical solution for the matrix **M**. Given that we want to find the matrix that translates **S** to **T**, i.e., **S**· **M = T** and that the source matrix **S** is not a square matrix (i.e., it does not have an inverse), we compute the translation matrix **M** by multiplying the pseudo-inverse (inverse approximation for non-square matrices) of the source matrix **S** with the target matrix **T**:

$$M = S^{+} \cdot T \tag{1}$$

where **S+** is the Moore-Penrose pseudoinverse of the source matrix **S**, i.e.,

$$S^{+} = (S^{T}S)^{-1} \cdot S^{T}. \tag{2}$$

In the joint semantic space of words across two languages, the Spanish word $perro$ is expected to be close to its English translation dog. At the same time, when two words are not direct translations, their semantic proximity could be correctly quantified as well.

First part of application was created in Python using library $numpy$ to calculate all matrices.

2.1 Vectors of documents

To use word embeddings in retrieval, we need to derive dense document vectors from word embedding vectors. We first describe the dataset, then describe the algorithm to compute the aggregate embedding vectors of documents.

2.1.1 Dataset

This dataset consists of the term vectors extracted from 60,730 Wikipedia English articles and their comparable Spanish articles sampled in 2009 (Platt et al., 2010). The vocabulary is formed by first word-breaking all documents, removing the top 50 most frequent terms and keeping the next 20,000 most frequent terms. No stemming or folding is applied. The format of these files is a sparse matrix(Yih et al., 2011). The three numbers in the first line indicate the number of rows (i.e., the number of articles), the number of columns (i.e., the vocabulary size 20,000) and the number of non-zero entries in the matrix. For the remaining lines, the three numbers are the row index, column index and TF-IDF value, respectively.

2.1.2 Calculations

We assumed that every document **d** contains terms t_1, ... , t_n and **e(t)** is the word embedding of the term **t**. The aggregate embedding vector of the document **d**, to be used for retrieval, was computed as weighted average of word embeddings:

$$e(d) = \frac{\sum_{i=1}^{N} w_i \cdot e(t_i)}{\sum_{i=1}^{N} w_i} \tag{3}$$

Weight w_i determines how much the word embedding of term t_i contributes to the aggregate embeddings. We uses TF-IDF coefficients as weights. For the English documents we multiplied the word embeddings in each document by the translation matrix from previous part, multiplied by the weight, found a sum and divided by sum of all weights. For the Spanish document we did the same without multiplying by the translation matrix.

For this part we used $c\#$ and have got two files for English and Spanish aggregate embedding vectors, respectively as a result.

2.2 Measures of Semantic Similarity

Our next task was to find more relevant document for each by cosine similarity between their aggregate dense vectors given from previous part. We made two dictionaries where keys were numbers of each document, than calculated cosine similarity between all pairs. The only problem was that if we use prepared function from sklearn-library in Python, calculation takes a lot of time. But if we calculate cosine similarity manually, it works faster. Therefore, at first we calculated norm of vector for English document and then compared with each Spanish document using formula for cosine of the angel between them.

$$similarity = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|_2 \|\mathbf{B}\|_2} \tag{4}$$

The highest cosine means the most similar pair.
The first output file consists of each document and more relevant Spanish one. The second capability of our program is to output the exact number of relevant documents dependent on cosine similarity between them. E.g. program ask to input the number of document and the number of most relevant documents for it.

2.3 Evaluation

To measure retrieval effectiveness in the standard way, we need a test collection consisting of three things: (1) a document collection; (2) a test suite of information needs, expressible as queries; (3) a set of relevance judgments for each query-document pair.
Since we know that for each document in English there is just one relevant in Spanish therefore in our case precision for each document equal to average precision:

$$AP = \frac{1}{place \quad of \quad real \quad relevant \quad document}$$

The mean average precision (MAP) is the sum of precisions for all documents divided by their number.

$$MAP = \frac{1}{n} \cdot \sum_{i=1}^{n} AP(d_i) \tag{5}$$

We have got all measures, therefore we were able to calculate the MAP of 0.1565. It means that according to documents' embedding vector we got approximately only 15% of perfect matching. In the attached file "ranked.csv" every line consists of four numbers: the number of document, the number of most relevant for it and two last numbers are precision written on different ways.

3 Experiments

We provide two interactive experiments to summarize all our calculations and new knowledge.

3.1 Ranking according to query

For the main task, we are giving a list of documents in English and Spanish with the objective of mapping the VSM of both languages by exploiting the linear translation introduced by (Mikolov et al., 2013). We then enabled the user to execute a query of their choice and specify the number of the relevant document they wish to retrieve. The system will compute the weights for each word in the query and obtain the embedding vector and multiply it with the translation matrix **M**, next it will multiply it by the weights and aggregate the results into one embedding vector representing the query. By computing the cosine similarity between the query and each document in Spanish in our dataset, we are able to rank the relevant Spanish documents according to the user's query, keeping in mind their preferred number of relevant document.

3.2 Toy-collection

For main tasks we used given collection of documents. It consists of only numbers of documents, all words and corresponding TF-IDF to each. Testing our program would be a hard problem, because we were not able to read whole article from our collection and decide whether the most relevant document was found right or not. Therefore, we made a toy-collection consisted of chapters from different books in English and Spanish: "Harry Potter" by J. K. Rowling, "The lord of the rings" by J. R. R. Tolkien, "Pride and prejudice" by Jane Austen. In this case now we were manage to write query with most popular words for each novel and check the output document. At first we had to create files with TF-IDF coefficients for each word in each document. We used the sklearn-library in Python and got two files for English and Spanish respectively. Then we computed the aggregate embedding vectors of documents in Spanish. This task had already done earlier therefore we just changed out dataset to new toy-collection. For this moment we were ready with all prepared files to testing.
For each English document we multiplied the words' embeddings by the translation matrix. For each of word in the query we had to find average TF-IDF coefficient between all our collection. It gives to us weights of words and we can say which word is more important and which we can interpret as stop-words. Then we multiplied by these weights with respect to the (3). Comparing with Spanish embedding vectors by cosine similarity did take a good result. E.g. for query $frodo \quad sam \quad beast \quad orc \quad blue$ we had the most relevant documents exactly chapters from "The Lord of the ring" or for query $harry \quad Potter \quad quidditch \quad holidays$ we had chapters from "Harry potter and the chamber of secrets".

4 Conclusion

We provide all files with our source code, which can be summarized in following description:
$Rafik.py$ - create a translation matrix,
$SpanishDocs.CS$ - $EnglishDocs.CS$ - create documents' dense embedding vectors,
$ranking.py$ - rank all documents by cosine similarity and output some the most relevant,
$evaluation.py$ - calculate R-precision and mean average precision for our collection,
$finalworking.py$ - ranking according to query,
$preprocess.py$ - creation TF-IDF confections and embedding vectors for toy-collection,
$realtest.py$ - experiment with toy-collection.
We show good performance on English-Spanish semantic similarity with bilingual trained embeddings. Further, our experimental results show that we can use our project as a good cross-lingual search model.

5 Acknowledgments

We thank Dr. Goran Glavaš for helpful lectures and support.

References

Cristian Cardellino. 2016. Spanish Billion Words Corpus and Embeddings, March.

Goran Glavaš, Federico Nanni, and Simone Paolo Ponzetto. 2017. Unsupervised cross-lingual scaling of political texts. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 688–693, Valencia, Spain, April. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

John C. Platt, Kristina Toutanova, and Wen-tau Yih. 2010. Translingual document representations from discriminative projections. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 251–261, Stroudsburg, PA, USA. Association for Computational Linguistics.

Wen-tau Yih, Kristina Toutanova, John C. Platt, and Christopher Meek. 2011. Learning discriminative projections for text similarity measures. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*, CoNLL '11, pages 247–256, Stroudsburg, PA, USA. Association for Computational Linguistics.