Making History Count:
How Word Embeddings Impact Event Detection in Historical Corpora

Andrea Ferretti

University of Milan, Milan, Italy
andrea.ferretti1@studenti.unimi.it

Abstract. This work tries to analyze the impact that a word embedding has in the event mentions detection in historical text task. Firstly word embeddings generated from historical and contemporary corpora are compared to study the difference between them; special attention is given to terms that commonly convey the realization of an event. Then the embeddings are used separately for event detection on a set of annotated corpora, both historical and contemporary, and the performances are matched. The results suggest that there are slight performances improvements when using a word embedding created from a corpus of a similar period to the text where events are being detected.

Keywords: Event Detection · Historical Event Detection · Word Embeddings · Embeddings comparison.

1 Introduction

Word embeddings aim to map words of a vocabulary to vectors of numbers. This is done in order to have a more versatile, tractable, and mathematical representation of those words and improve the performances of several natural language processing tasks. The more intuitive way to do so would be to have a one-hot encoding representation of each word in the vocabulary. This method, however doesn't provide any information about the meaning of a word. To allow the vectors to retain semantic meaning the foundamental idea of distributional semantics is used: words have similar meaning if they appear in similar contexts. This translates to, given a corpus of documents, representing a word as a distribution, over the vocabulary, of the frequency of the words that appear in its context in the corpus. These vectors have the limit of being highly dimensional and extrimely sparse: every word would be represented by a vector of tens of thousands dimensions (the size of the vocabulary) the vast majority of which would have value zero.

To solve these problems several techniques, either based on neural networks or matrix factorization, have been proposed [1]. They both start from the sparse and high dimensional co-occurence matrix and obtain a fixed length, dense, and real valued vector for each word. The vectors are not interpretable when taken singularly, but when analyzed and compared to one another they show interesting properties: words that have similar meaning (in the distributional semantics sense) have vectors that are closed to each other according to the cosine or Euclidean distance, and pairs of vectors representing pairs of words analogy, such as man and woman, king and queen, also have the similar distances. Examples of embeddings algorithms with comparable performance based respectively on neural networks and matrix factorization techniques are Word2Vec [2] and GloVe [3].

Event detection is one of the numerous tasks that make use of word embeddings. Part of the complexity of the task stems from the intrinsic ambiguity of what can be defined as an event, the Oxford English Dictionary definition of event “anything that happens, or is contemplated as happening; an incident, occurrence”, can be taken as a good baseline. Event detection goes beyond just finding events in unstructured and unprocessed text, it is often concerned with classifying them according to their type and identifying participants and attributes [4].

More recently developed event detection systems make heavy use of deep learning techniques, in particular recurrent neural network, in all of its variants, appears to be the more promising model thanks to the possibility of analyzing arbitrarily long sequences of labled data [5]. Particulary bi-directional long short-term memory (BiLSTM) network appear to be the preferred model.

2 Research question and methodology

Most of the efforts in developing event detection systems are focused on contemporary text and on corpora of the biomedical field. The work in [6] tries to change this by addressing the task of event detection in historical text. It does so by providing a new annotated dataset of events in historical text, a new word embedding created from various historical documents, and a model to detect those events. Starting from these available results, the aim is to both analyze language variation in time and the extent to which this adresses the task of event detection in historical text.

To accomplish the first task the word embedding obtained from historical text of [6] is compared to a contemporary word embedding, and, in order to have a meaningful baseline, diferrences between two contemporary word embeddings are also taken. There isn’t a clear straight forward way to compare word embeddings, but inspired by [7] three methods of analysis are derived. First the distances between pairs of words that are supposed to represent a similar juxtaposition are compared for the embeddings. Then, after having rotated one embedding to best approximate the other [8], the cosine distance of two vectors representing the same words in the two embeddings is calculated. Finally, given a word, the set of k nearset words, according to the cosine distance, is taken for each embedding and the Jaccard distance between the two sets is computed. The rational for all three methods is that the closer are the vectors the more similar are the embeddings.

The set of words chosen for the analysis contains verbs which convey the realization of an event (as described in the event detection guidelines of [6]), some common words, and words whose meaning is supposed to have evolved or changed during time. Common words are taken as a baseline to measure the distance between other words in the different embeddings.

As for the second task an event detection model based on BiLSTM networks [9] is used. Given two annotated corpora, one historical and one contemporary, each one is split into training and test set, and different combinations of these splits are used to train and evaluate the model to detect event mentions in text (the event classification task is not considered). For example the model can be trained on the training set of the historical corpus and evaluated on the test set of the contemporary corpus. The various results are analyzed to see if there's a substantial difference in performance when changing from historical to contemporary text. The model also takes as a parameter a word embedding: the impact on performances of using a word embedding obtained from historical text opposed to using a word embedding obtained from contemporary text is also analyzed.

3 Experimental results

3.1 Word embeddings comparison

The word embedding based on historical text [1] (embHisto) is provided by [6] and is obtained by using the GloVe algorithm on a subset of the Corpus of Historical American English consisting of 36,856 texts published between 1860 and 1939 for a total of more than 198 million tokens. The contemporary embedding (embGlove) is also trained using GloVe on a corpus composed by Wikipedia 2014 and Gigaword 5 for a total of 6 billion tokens [2]. The other contemporary embedding (embCurrent) is also GloVe based and is trained on a subset of Common Crawl corpus for a total of 42 billion tokens [3]. The two contemporary embeddings are provided by [3] and all three embeddings have vectors of 300 dimensions.

The first method of analysis compares the cosine distance between pairs of words in the various embeddings. The distance function used is the cosine distance because of its invariance to the magnitude of the vectors, accounting only for the angle between them. The underline idea is that juxtapositions should mantain the same distance between each embedding if the relative meaning of the words in the pair is the same. The results are shonw in Table 1.

The second method, given a word, takes the two vectors representing that word in the two embeddings that are compared and computes the cosine distance between those vectors. In order to have a more meaningful comparison, the embeddings should be aligned to compensate for the random position of the vectors in the space [7], this translates to rotating one embedding to best aproximate the

[1] https://github.com/dhfbk/Histo

[2] http://nlp.stanford.edu/data/glove.6B.zip

[3] http://nlp.stanford.edu/data/glove.42B.300d.zip

Table 1: Cosine distances between a pair of words for the various embeddings.

Word pair	**Cosine distance**		
	embHisto	embGlove	embCurrent
(man, woman)	0.337	0.300	0.195
(he, she)	0.283	0.292	0.130
(king, queen)	0.359	0.366	0.240
(brother, sister)	0.344	0.405	0.250
(male, female)	0.198	0.105	0.062
(and, but)	0.405	0.418	0.243

other. This technique exploits the fact that the relative position of many common words is similar across various embeddings, as seen in the previous paragraph. Formalizing the problem we have a matrix A representing the embedding to be rotated, and the matrix B representing the embedding to be aproximate. The matrices, where each row corresponds to a vector of an embedded word, must have the same number of rows and columns, therefore the vocabularies of the embeddings need to be equalized. The solution to how to rotate the matrix is given by the rotation matrix Q obtained by solving the optimization problem know in literature as the orthogonal Procrustes problem:

$$\underset{\mathbf{Q}}{argmin}\ ||B - AQ|| \\ subject\ to\ QQ^t = I$$

After the rotation the cosine distance can be computed. It has been calculated, for the set of chosen words, the distance between the embHisto and embGlove, and the distance between dmbCurrent and embGlove. The results are shown in Table 2.

The third method compares neighbours, according to the cosine distance, of a word in one embedding to the neighbours of the same word in another embedding. According to the properties of word embeddings, the more similar the embeddings the more similar the neighbours of a word should be. To quantify this similarity, given a word, the set C of words composed of its 100 nearest neighbours in one embedding is used to compute the Jaccard distance with the set of 100 nearest neighbours of that word in another embedding. The Jaccard distance between set M and N is computed as:

$$d_J(M, N) = 1 - \frac{|M \cap N|}{|M \cup N|}$$

As before the comparison is done on the same list of words and between the pair of embeddings, after equalizing the vocabularies, embHisto and embGlove, and embCurrent and embGlove. Results are shown in Table 3

Table 2: Cosine distances between words in different embeddings after alignment.

Word	Cosine distance	
	embHisto	embCurrent
	embGlove	embGlove
did	0.413	0.462
does	0.438	0.413
do	0.450	0.319
be	0.462	0.425
is	0.473	0.425
make	0.473	0.361
made	0.496	0.507
have	0.496	0.413
had	0.529	0.507
but	0.529	0.413
and	0.551	0.450
done	0.571	0.485
get	0.571	0.438
were	0.582	0.561
got	0.582	0.473
was	0.592	0.540
has	0.592	0.540
makes	0.667	0.507
walk	0.701	0.630
gets	0.710	0.561
hotel	0.765	0.726
gun	0.773	0.693
airplane	0.788	0.529
sport	0.830	0.701
fame	0.837	0.639
gotten	0.919	0.582
sex	0.930	0.851
keyboard	0.985	0.684
computer	0.995	0.621

Table 3: Jaccard distance between sets of neighbours in different embeddings for a given word.

Word	Jaccard distance	
	embHisto	embCurrent
	embGlove	embGlove
and	0.155	0.199
had	0.166	0.173
was	0.187	0.219
have	0.192	0.169
were	0.194	0.199
be	0.197	0.160
is	0.204	0.184
but	0.233	0.188
has	0.241	0.192
made	0.260	0.183
do	0.262	0.125
got	0.263	0.201
does	0.265	0.158
did	0.265	0.177
make	0.269	0.153
get	0.270	0.183
hotel	0.289	0.307
gets	0.309	0.236
done	0.315	0.165
makes	0.329	0.224
walk	0.332	0.229
gun	0.421	0.226
airplane	0.452	0.224
fame	0.495	0.219
sport	0.556	0.210
sex	0.567	0.310
gotten	0.585	0.231
keyboard	0.764	0.255
computer	1.076	0.174

3.2 Event mentions detection

For the second task of detecting event mentions a BiLSTM network [4], whose detalied description is given in [9], is the model used. The complete structure is a bidirectional recurrent neural network that takes as input, encoding the categorical feature as one-hot encoded vectors when needed, the embedding for the word, the lemma of the word, the part of speech, and the casing. These input features are concatenated and fed to a LSTM network composed by two layer of 75 LSTM units each and having a dropout of 0.25 after each layer. The model is bidirectional, meaning there are two LSTM network, one running rom the beginning of the sentence to the end while the other runnig in reverse. The outputs of the two networks are concatenated and fed to a softmax classifier layer.

The datasets used to train the model are the Histo Corpus [5], composed of news and travel narratives documents published between the second half of the nineteenth century and the beginning of the twentieth century [6], and the NIST TAC KBP 2015 Event Nugget Detection corpus [6], composed of contemporary weblogs, broadcast news, newsgroups, broadcast conversation [10]. Event mentions have been manually annotated in the two corpora following different guidelines. Both datasets have been converted to a IOB2 compatible format and they have been divided into a training set and a test set.

The model was trained and evaluated on a subset of the possible combinations of training sets (trainHisto, trainTAC), test sets (testHisto, testTAC), and two embeddings (embHisto, embGlove). For example in some cases the model was evaluated on trainHisto dataset to exploit its bigger size compared to testHisto. The metrics used for evaluation are precision, recall, and F1-score. Results are shown in Table 4.

Table 4: Cosine distances between a pair of words for the various embeddings.

embedding	**training**	**evaluation**	**precision**	**recall**	**F1 score**
embGlove	trainHisto	testHisto	0.792	0.779	0.785
embHisto	trainHisto	testHisto	0.793	0.800	0.797
embGlove	trainHisto	testTAC	0.235	0.657	0.346
embHisto	trainHisto	testTAC	0.236	0.637	0.345
embGlove	trainTAC	testTAC	0.720	0.743	0.731
embHisto	trainTAC	testTAC	0.775	0.663	0.715
embGlove	trainTAC	trainHisto	0.729	0.184	0.294
embHisto	trainTAC	trainHisto	0.660	0.226	0.336

[4] https://github.com/UKPLab/emnlp2017-bilstm-cnn-crf

[5] https://github.com/dhfbk/Histo

[6] https://github.com/UKPLab/tac2015-event-detection/tree/master/tacdata

4 Concluding remarks

The analysis of the similarity between word embeddings shows that the magnitude of the differences between embHisto and embGlove doesn't look particulary out of order when compared to the difference between the two contemporary embeddings, embGlove and embCurrent. In the first method of analysis, the juxtapositions, embHisto displays a distance between pair of words very close to the embGlove one, whereas embCurrent seems to have very differenet values. In the second method, on avarage, embCurrent is a bit closer to embGlove than embHisto is, but, when looking at the common words and verbs that usually convey the happening of an event, the discrepancy reduces a lot. This holds true also for the third method, where the metric is the Jaccard distance. In order to analyze the evolution of the language it is interesting to see that terms like computer, keyboard, fame, sport, that have become more popular, used, or chagned their meaning in the recent period, also have the biggest gap when comparing embHisto to embGlove. This first analysis seems to suggest that while the historical embedding has differences to the contemporary one, they don't seem particulary relevant when only verbs conveying event are considered.

The results of the event detection task show that, when detecting events in a historical corpus, using a historical word embeddings improves the F1 score by around 0.01. Roughly the same gain in performances is obtained when employing a contemporary word embedding for event detection in a contemporary corpus. Mixing the two datasets, by training on the historical one and evaluating on the contemporary one (or viceversa), gives similar improvements but with a much lower F1 score due to the different labeling guidelines used when annotating the datasets. From the results, and when looking at the annotated corpora, it's pretty obvious that the definition of what constitutes an event in the contemporary corpus is stricter when compared to the historical corpus. This is expalins why the F1 is low score for all mixed combinations and why, when training on the historical corpus and evaluating on the contemporary one, the precision is low and the recall is high, and the opposite happens when inverting the corpora.

In conclusion these experiments showed a slight improvement in the event deteciton in historical text task when using a historical embedding. Obviously a word embedding depends greatly on the algorithm and the corpus from which it has been created, and the event detection task is strongly bound to how the events are annotated in the training set, therefore more experiments with different corpora and different embeddings algorithms would be quite usefull to see if the observerd results are a recurring occurence.

References

1. F. Almeida and G. Xexo, "Word embeddings: A survey," 2019.
2. T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems 26* (C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, eds.), pp. 3111–3119, Curran Associates, Inc., 2013.
3. J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, 2014.
4. J. Kalita, "Detecting and extracting events from text documents," 2016.
5. X. Wei and B. Wang, "A survey of event extraction from text," *IEEE Access*, vol. PP, pp. 1–1, 11 2019.
6. R. Sprugnoli and S. Tonelli, "Novel event detection and classification for historical texts," *Computational Linguistics*, vol. 45, no. 2, pp. 229–265, 2019.
7. J. Chen, Y. Tao, and H. Lin, "Visual exploration and comparison of word embeddings," *Journal of Visual Languages and Computing*, vol. 48, 08 2018.
8. W. L. Hamilton, J. Leskovec, and D. Jurafsky, "Diachronic word embeddings reveal statistical laws of semantic change," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, (Berlin, Germany), pp. 1489–1501, Association for Computational Linguistics, aug 2016.
9. N. Reimers and I. Gurevych, "Optimal hyperparameters for deep lstm-networks for sequence labeling tasks," 2017.
10. T. Mitamura, Z. Liu, and E. H. Hovy, "Overview of tac kbp 2015 event nugget track.," 2015.