Deep Learning for Social Sensing from Tweets

Giuseppe Attardi, Laura Gorrieri, Alessio Miaschi and Ruggero Petrolito

DOI: 10.4000/books.aaccademia.1291
Publisher: Accademia University Press
Place of publication: Torino
Year of publication: 2015
Published on OpenEdition Books: 11 November 2016
Serie: Collana dell'Associazione Italiana di Linguistica Computazionale
Electronic ISBN: 9788899200008

http://books.openedition.org

Electronic reference
Deep Learning for Social Sensing from Tweets

Giuseppe Attardi, Laura Gorrieri, Alessio Miaschi, Ruggero Petrolito
Dipartimento di Informatica
Università di Pisa
Largo B. Pontecorvo, 3
I-56127 Pisa, Italy
attardi@di.unipi.it

Abstract

English. Distributional Semantic Models (DSM) that represent words as vectors of weights over a high dimensional feature space have proved very effective in representing semantic or syntactic word similarity. For certain tasks however it is important to represent contrasting aspects such as polarity, opposite senses or idiomatic use of words. We present a method for computing discriminative word embeddings can be used in sentiment classification or any other task where one needs to discriminate between contrasting semantic aspects. We present an experiment in the identification of reports on natural disasters in tweets by means of these embeddings.

Italiano. I Distributional Semantic Model (DSM) rappresentano le parole come vettori di pesi in uno spazio di feature ad alte dimensioni, e si sono dimostrati molto efficaci nel rappresentare la similarità semantica o sintattica tra parole. Per certi compiti però è importante rappresentare aspetti contrastanti come la polarità, significati opposti o parole usate con significato idiomatico. Presentiamo un metodo per calcolare dei word embedding discriminativi che possono essere usati nella sentiment classification o per qualunque altro compito dove vi sia necessità di discriminare tra aspetti semantici contrastanti. Presentiamo un esperimento sull’identificazione di tweet relativi a calamità naturali utilizzando questi embedding.

1 Introduction

Distributional Semantic Models (DSM) that represent words as vectors of weights over a high dimensional feature space (Hinton et al., 1986), have proved very effective in representing semantic or syntactic aspects of lexicon. Incorporating such representations has allowed improving many natural language tasks. They also reduce the burden of feature selection since these models can be learned through unsupervised techniques from plain text.

Deep learning algorithms for NLP tasks exploit distributional representation of words. In tagging applications such as POS tagging, NER tagging and Semantic Role Labeling (SRL), this has proved quite effective in reaching state of art accuracy and reducing reliance on manually engineered feature selection (Collobert & Weston, 2008).

Word embeddings have been exploited also in constituency parsing (Collobert, 2011) and dependency parsing (Chen & Manning, 2014). Blanco et al. (2015) exploit word embeddings for identifying entities in web search queries.

Traditional embeddings are created from large collections of unannotated documents through unsupervised learning, for example building a neural language model (Collobert et al. 2011; Mikolov et al. 2013) or through Hellinger PCA (Lebrèt and Collobert, 2013). These embeddings are suitable to represent syntactic similarity, which can be measured through the Euclidean distance in the embeddings space. They are not appropriate though to represent semantic dissimilarity, since for example antonyms end up at close distance in the embeddings space.

In this paper we explore a technique for building discriminative word embeddings, which incorporate semantic aspects that are not directly
obtainable from textual collocations. In particular, such embedding can be useful in sentiment classification in order to learn vector representations where words of opposite polarity are distant from each other.

2 Building Word Embeddings

Word embeddings provide a low dimensional dense vector space representation for words, where values in each dimension may represent syntactic or semantic properties.

For creating the embeddings, we used DeepNL\(^1\), a library for building NLP applications based on a deep learning architecture. DeepNL provides two methods for building embeddings, one is based on the use of a neural language model, as proposed by Collobert et al. (2011) and one based on a spectral method as proposed by Lebret and Collobert (2013).

The neural language method can be hard to train and the process is often quite time consuming, since several iterations are required over the whole training set. Some researcher provide precomputed embeddings for English\(^2\).

Mikolov et al. (2013) developed an alternative solution for computing word embeddings, which significantly reduces the computational costs and can also exploit concurrency through the Asynchronous Stochastic Gradient Descent algorithm. An optimistic approach to matrix updates is also exploited to avoid synchronization costs.

The authors published single-machine multi-threaded C++ code for computing the word vectors\(^3\). A reimplementation of the algorithm in Python, but with core computations in C, is included in the Genism library (Řehůřek and Sojka, 2010)

Lebret and Collobert (2013) have shown that embeddings can be efficiently computed from word co-occurrence counts, applying Principal Component Analysis (PCA) to reduce dimensionality while optimizing the Hellinger similarity distance.

Levy and Goldberg (2014) have shown similarly that the skip-gram model by Mikolov et al. (2013) can be interpreted as implicitly factorizing a word-context matrix, whose values are the pointwise mutual information (PMI) of the respective word and context pairs, shifted by a global constant.

2.1 Discriminative Word Embeddings

For certain tasks, as for example sentiment analysis, semantic similarity is not appropriate, since antonyms end up at close distance in the embeddings space. One needs to learn a vector representation where words of opposite polarity are distant.

Tang et al. (2013) propose an approach for learning sentiment specific word embeddings, by incorporating supervised knowledge of polarity in the loss function of the learning algorithm. The original hinge loss function in the algorithm by Collobert et al. (2011) is:

\[ \mathcal{L}_{CW}(x, x') = \max(0, 1 - f(x) + f(x')) \]

where \( x \) is an ngram and \( x' \) is the same ngram corrupted by changing the target word with a randomly chosen one, \( f(\cdot) \) is the feature function computed by the neural network with parameters \( \theta \). The sentiment specific network outputs a vector of two dimensions, one for modeling the generic syntactic/semantic aspects of words and the second for modeling polarity.

A second loss function is introduced as objective for minimization:

\[ \mathcal{L}_{SS}(x, x') = \max(0, 1 - \delta_c(x)f(x) + \delta_d(x)f(x')) \]

where the subscript in \( f(\cdot) \) refers to the second element of the vector and \( \delta_c(\cdot) \) is an indicator function reflecting the sentiment polarity of a sentence, whose value is 1 if the sentiment polarity of \( x \) is positive and -1 if it is negative.

The overall hinge loss is a linear combination of the two:

\[ \mathcal{L}(x, x') = \alpha \mathcal{L}_{CW}(x, x') + (1 - \alpha) \mathcal{L}_{SS}(x, x') \]

Generalizing the approach to discriminative word embeddings entails replacing the loss function \( \mathcal{L}_{ss} \) with a one-vs-all hinge loss function:

\[ \mathcal{L}_{h}(x, t) = \max(0, 1 + \max_{y \neq t} f(x)_t - f(x)_y) \]

where \( t \) is the index of the correct class.

The DeepNL library provides a training algorithm for discriminative word embedding that performs gradient descent using an adaptive learning rate according to the AdaGrad method. The algorithm requires a training set consisting of documents annotated with their discriminative value, for example a corpus of tweets with their sentiment polarity, or in general documents with

---

\(^1\) https://github.com/attardi/deepnl
\(^2\) http://ronan.collobert.com/senna/
http://metaoptimizem.com/projects/wordreps/
http://www.fit.vutbr.cz/~imikolov/rnnlm/
http://ai.stanford.edu/~ehhuang/
\(^3\) https://code.google.com/p/word2vec
multiple class tags. The algorithm builds embeddings for both unigrams and ngrams at the same time, by performing variations on a training sentence replacing not just a single word, but a sequence of words with either another word or another ngram.

3 Deep Learning Architecture

The Deep Learning architecture used for training discriminative word embeddings consists of the following layers:

1. Lookup layer: extracts the embedding vector associated to each token
2. Linear layer
3. Activation layer: using the hardtanh function
4. Linear layer
5. Hinge loss layer

4 Experiments

We tested the use of discriminative word embeddings in the task of social sensing, i.e. of detecting specific signals from social media. In particular we explored the ability to monitor and alert about emergencies caused by natural disasters. We explored the corpus of Social Sensing\(^4\), which consists of 5,642 tweets about natural catastrophic events like earthquakes or floods. To obtain a balanced training set, we combined this corpus with a set of generic tweets, consisting of 23,507 tweets. The combined corpus, consisting of 29,149 tweets, was randomly split into a training, development and test set consisting respectively of 23,850, 2,649 and 2,650 tweets.

4.1 Lexicon

Most sentiment analysis systems exploit a specialized lexicon (Rosenthal et al, 2014; Rosenthal et al, 2015). We built a lexicon of words related or indicative of disasters, by using the Italian Word Embeddings interface\(^5\). Starting from a seed set of few specialized words we produced a lexicon of 292 words (including words with a hashtag).

4.2 Classifier

For detecting tweets reporting about natural disasters, we exploit an SVM classifier, which uses as continuous features the word embeddings created from the text of the Italian Wikipedia. Additionally a set of discrete features is used, similar to those used in the top scoring system in the task 10 of SemEval 2014 on Sentiment Analysis in Twitter (Mohammad et al., 2014). These features are summarized in the following table:

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>allcaps</td>
<td>feature telling whether a word is all in uppercase</td>
</tr>
<tr>
<td>EmoPos</td>
<td>Presence of a positive emoticon</td>
</tr>
<tr>
<td>EmoNeg</td>
<td>Presence of a negative emoticon</td>
</tr>
<tr>
<td>Elongated</td>
<td>Presence of an elongated word</td>
</tr>
<tr>
<td>Lexicon</td>
<td>Number of word present in a lexicon</td>
</tr>
<tr>
<td>count</td>
<td>Lowest score of word in lexicon</td>
</tr>
<tr>
<td>min</td>
<td>Score of the last word present in lexicon</td>
</tr>
<tr>
<td>last</td>
<td>Sum of the scores of words present in lexicon</td>
</tr>
<tr>
<td>sum</td>
<td>Count of negative words</td>
</tr>
<tr>
<td>Negation</td>
<td>Count of multiple punctuations (e.g. “!!!”)</td>
</tr>
<tr>
<td>Elongated</td>
<td></td>
</tr>
<tr>
<td>punct</td>
<td></td>
</tr>
<tr>
<td>Ngrams</td>
<td>Ngrams of length 2-4</td>
</tr>
</tbody>
</table>

4.3 Results

We created generic word embeddings on the corpus consisting of the plain text extracted from the Italian Wikipedia, for a total of 1,096,243,235 tokens, 4,456,972 distinct. We selected the 100,000 most frequent words and we created word embeddings for them, with a space dimension of 64.

The table below shows the results obtained with the discriminative word embeddings compared to a baseline obtained with the same classifier using the generic embeddings.

<table>
<thead>
<tr>
<th>Data</th>
<th>System</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Develop baseline</td>
<td>85.91</td>
<td>72.66</td>
<td>78.73</td>
<td></td>
</tr>
<tr>
<td>Develop DE</td>
<td>87.08</td>
<td>76.37</td>
<td>81.37</td>
<td></td>
</tr>
<tr>
<td>Test baseline</td>
<td>86.87</td>
<td>70.96</td>
<td>78.11</td>
<td></td>
</tr>
<tr>
<td>Test DE</td>
<td>85.94</td>
<td>75.05</td>
<td>80.12</td>
<td></td>
</tr>
</tbody>
</table>

The results show a significant improvement in recall with respect to the baseline, which leads to over a 2-point improvement in F1.

4.4 Related Work

Social sensing research is a rapidly growing field; however, it is difficult to compare our work with others since the data sets used are different.

The only experiment performed on the same data set, is described in (Cresci et al., 2015), which focuses on distinguishing whether damage

\(^4\) http://socialsensing.it/en/datasets
\(^5\) http://tanl.di.unipi.it/embeddings/
was reported, rather than just reporting a disaster. Sixteen experiments were carried out, using four subsets of the corpus for training, corresponding to four disaster events, and testing on either different events (cross-event) or same/different disaster types (in-domain, out-domain). F1 scores in detecting non-relevant tweets ranged between 19% and 28% for cross-event and out-domain and reached 73% for in-domain in one of the in-domain tests.

5 Conclusions
We have presented the notion of discriminative word embeddings that were designed to cope with semantic dissimilarity in tasks like sentiment analysis or multiclass classification.

As an example of the effectiveness of this type of embeddings in other applications, we have explored their use in detecting tweets reporting alerts or notices about natural disasters.

Our approach consisted in using a classifier trained on a corpus of annotated tweets, using discriminative embeddings as features, instead of the typical manually crafted features or dictionaries employed in tweet classification tasks as sentiment analysis.

In the future, we plan to explore the use of convolutional network classifier, also provided by DeepNL, without any additional features, as Severyn and Moschitti (2015) have done for the SemEval 2015 task on Sentiment Analysis in Twitter.

References
Speech Communication Association, Makuhari, Chiba, Japan.


