

Word Embeddings

19 Nisan 17

Algorithms / Tools	3
Hyperparameters	3
Text collection	3
Domains	4
Word embeddings in the literature	4
Pre-trained embeddings (English or multilingual)	5
Pre-trained embeddings (Turkish)	6
Problems	7
Future Directions	7
todo	8
References	8

Word embedding is a general term denoting a vector representation of a word in a context or a dataset (vector embedded in space). The simplest representation can be illustrated in a word-document relationship where a word vector will have in its cells the number of times that word occurs in the documents (if w is the word vector of size $1 \times N_{\text{docs}}$, then $w(j)$ is the number of times that word occurs in doc j). More elaborate weighting schemes other than counting have been proposed such as TF*IDF and PMI among the most widely used.

Since the number of documents can be very large, the dimension of these vectors is preferably decreased since dimensionality reduction in the vector space does not cause much loss information. The most widely known of such methods are PCA¹ and SVD². In these methods, the matrix containing the word vectors in its rows is factorized into its components and the most informative components / dimensions are selected to be representing it; then, the matrix is reconstructed out these selected components / dimensions, this time having smaller number of columns, thus smaller dimensionality. However, it is impossible to tell what these new columns represent or which features they are while it used to be documents (remember the simple case in the example) or context or neighbour words, depending on the task. These new unnamed columns can be said to be the latent features representing the words.

¹ https://en.wikipedia.org/wiki/Principal_component_analysis

² https://en.wikipedia.org/wiki/Singular_value_decomposition

Representations obtained by neural networks is similar to the vectors in the reduced space. Indeed, neural representations have a claim to overcome the sparsity problem occurring in the traditional methods. These representations are based on the hypothesis that the similar words occur in similar contexts (Jurafsky&Martin, 2017) so the network tries to learn these co-occurrence / related-occurrence history. This is why this representation scheme is called distributed representations in the continuous space (meaning is distributed to the neighbours / context in the continuous space).

The method is unsupervised; no labelled data is required. The vectors are produced by a neural network fed with a large set of documents. In other words, the tokens (meaning-affecting (like punctuation) or meaningful units (words) in the text) in the text, without disturbing their order in the sentences, are, one by one, input to the network and the network outputs their vectors after many iterations.

To produce the vector of one word, the network initially takes as input a random vector for the word along with the vectors of its context (neighbour words) and updates these vectors according to the contextual information; when the process is stopped, the network have produced a vector of a predefined size (like 50, 100, 300) for each word. Many variations of this representation have been proposed (Turian et al. 2010, Mikolov et al. 2013, Pennington et al. 2014), the most widely known algorithm being Skip-Gram (Mikolov et al. 2013), which has proven to be outperforming the others in some tasks (Schnabel et al. 2015, Chiu et al. 2016, Lai et al. 2016).

Neural embeddings have been proven to perform in tasks similarity and analogy detection better than the classical count-based methods (remember the simple example and the other weighting schemes) (Baroni et al. 2014) but the questions as to whether or not they always outperform non-neural representations and as to why they perform better are still not certainly answered (Levy et al. 2014). For now we can say that, these representations not only capture semantic similarity (the distance between the vectors of girl and boy is approximately the distance between the vectors of woman and man) but also syntactic similarity ($|\text{good}| - |\text{bad}| \sim |\text{better}| - |\text{worse}|$) and this provides with many openings and performance improvements for other more complex tasks in natural language understanding. Typically, the neural networks designed for different linguistic tasks such as NER, POS tagging, textual entailment, takes as input the vectors of the words in its training set with maybe some additional task-specific features. Thus, linguistic feature engineering for such tasks is reduced to a minimum level while the neural word embeddings encompass many of the possible linguistic features, provenly or supposedly.

To develop more accurate NLP applications; to bypass the data and system scarcity problems in the NLP technologies for Turkish, we set out to focus on neural networks and word embeddings as their basic requirement. We first aim to produce Turkish word embeddings. The reason why we would like to produce our own embeddings is that the existing few vector sets (for turkish) are produced from domains not properly corresponding to our needs - we would like to add our in-domain data as much as possible and whenever we need.

Once the corpus is ready, producing the vectors takes a few hours using off-the-shelf algorithms. The important stage is 1) collecting, blending, preprocessing the texts to make a corpus 2) tuning the hyperparameters of the algorithm. These two, in particular the hyperparameters (Levy et al. 2014), are indicated to be the primary factors affecting the future performance of the word vectors (Schnabel et al. 2015). After obtaining a number of sets of embeddings from different settings, we will evaluate them on some tasks to see their performance and will decide on the set that performs the best.

We would like to first see the alternatives of corpora, algorithms, hyperparameters and optimization settings in the literature and what has been done on the Turkish side.

Algorithms / Tools

- The most widely used algorithms are word2vec³ (with two popular architectures, CBOW and Skip-Gram) and GloVe⁴. Skip-Gram is said to be outperforming others in a number of tasks (Baroni et al. 2014).
- In the light of previous studies in the literature, we are going to use the word2vec algorithm with Skip-Gram.
- The package gensim (Rehurek&Sojka, 2010) has an implementation of word2vec, easy to use ⁵ and control. We are going to use that package.
 - Gensim has Igpl licence, allowing commercial use.

Hyperparameters

- Vector length
- Window size - context window size
- Min-count - minimum number of occurrences of the words in the final set
- Negative sampling - number of words to consider in the corpus to normalize current word's probability
- Sub-sampling - what portion of frequent words should be ignored
- Alpha - learning rate in the neural network

Text collection

We need a large set of documents, which will have at least 600M tokens, corresponding to 6M documents. It is not that this set will be a random collection of sentences or texts from various texts. We want it to represent the language itself and the language in use in the domains of the

³ <https://code.google.com/archive/p/word2vec/>

⁴ <https://nlp.stanford.edu/projects/glove/>

⁵ <https://rare-technologies.com/word2vec-tutorial/>

target work as much as possible. For this, we will sample documents from some domains under some contexts (proportional to the domain's importance in the representation).

There are two very large corpora in Turkish, BOUN corpus (Sak et al. 2008) and METU corpus (Say et al. 2002) but they are not available for commercial use.

Domains

- News
 - Rich in word variety; one the best available sources that depicts a language.
 - Selection:
 - Should be from diverse sources
 - Should be from various time intervals
 - 50% of the final corpus.
- Wikipedia
 - Adds various words with definitions, contributing to make a good context.
 - Selection:
 - Size of March, 2016 dump: 2GB⁶
 - 10%?
- Forums
 - For brands
 - Which ones?
 - Which forums?
- Social media
 - For informal language, neologies.
 - Selection:
 - Twitter, facebook
- Literary works
 - For the proper use of language
 - Selection:
 - ??

Word embeddings in the literature

- What are the contents of their corpora?
- What are their hyperparameters and method?

⁶ <https://archive.org/details/trwiki-20160305>

Pre-trained embeddings (English or multilingual)

(The first two (word2vec and GloVe) pre-trained embeddings are of course applicable for other languages but these available English word vectors have been the standard so we introduce them separately).

- Known as word2vec embeddings based on (Mikolov et al. 2013) and (Mikolov et al. 2013a)
 - For English
 - Method: Word2vec with CBOW
 - Raw corpus:
 - Source: Google news
 - Size: 100B tokens
 - Vocabulary: 3M words as well as phrases
 - Evaluation: (Mikolov et al. 2013a) and many other works, for example (Kim 2014, Santos et al. 2014, with some modifications Lample et al. 2016).
 - Availability:
 - [https://drive.google.com/file/d/0B7XkCwpI5KDYNINUTTISS21pQmM/edit
?usp=sharing](https://drive.google.com/file/d/0B7XkCwpI5KDYNINUTTISS21pQmM/edit?usp=sharing)
 - Open to commercial use
- Known as GloVe embeddings based on (Pennington et al. 2014)
 - For English
 - Method: GloVe
 - Raw corpus:
 - Sources: Wikipedia, commoncrawl collections, twitter
 - Vectors per source - size depends on the source
 - Evaluation: (Pennington??) and many other
 - Availability
 - <https://nlp.stanford.edu/projects/glove/>
 - Licence ambiguous
- A number of vectors for English, produced with different architectures like word2vec (CBOW or Skip-Gram), GloVe and dependency-based (Levy&Goldberg, 2014)
 - Details on the page: <https://github.com/3Top/word2vec-api>
- Kyubyong
 - Pre-trained vectors for 30+ languages including turkish
 - Method: both word2vec and fasttext (vectors per method)
 - Raw corpus:
 - Source: wikipedia dumps
 - Size: 370M tokens
 - Vocabulary: 30K words

- $|w| = 300$
- Evaluation: None
- Availability
 - <https://github.com/Kyubyong/wordvectors>
 - MIT licence, allows commercial use.
- Polyglot (Al-Rfou et al. 2013)
 - For 117 languages including turkish
 - Method: similar to (Bengio et al. 2009), window-based and noise-additive
 - Raw corpus:
 - Wikipedia
 - Size: 23M - 1B tokens
 - Vocabulary: 100K
 - $|w| = 64$
 - Evaluation: Qualitative. On NER (polyglot-NER??), seems not very successful.
 - Availability
 - <https://sites.google.com/site/rmyeid/projects/polyglot>
 - Licence requires contact

Pre-trained embeddings (Turkish)

- BOUN tweetNER (Okur et al. 2016)
 - Trained for the task of NER in twitter
 - Method: word2vec with Skip-Gram (Mikolov et al. 2013)
 - Raw corpus:
 - BOUN web corpus (news and texts from some web sites): 491M tokens; 21M tweets: 293M tokens
 - Preprocessing:
 - Lowercased
 - URLs, mentions, smileys, hashtags, mentions normalised
 - Vocabulary: 350K words
 - $|w| = 200$
 - Evaluation: Used for NER. No comparative results. Only, NER trained with embeddings performs better than NER trained on normalized text. (Their previous NER (Demir&Özgür2014) that uses word embeddings as additional features outperforms classical ones.)
 - Availability:
 - <http://tabilab.cmpe.boun.edu.tr/projects/ttner/>
 - Allows commercial use
- METU tweets (Önal&Karagöz, 2015)

- Trained for NER general purpose
- Method: word2vec with Skip-Gram (Mikolov et al. 2013)
- Raw corpus:
 - BOUN web corpus (news and texts from some web sites): 491M tokens; a dump of turkish wikipedia: ~400M tokens
 - Vocabulary: 954K words
 - $|w| = 50$
- Evaluation: Qualitative. For NER trained on the same dataset, this system does not outperform (Demir&Özgür, 2014) who also make use of word embeddings but with this result we cannot directly compare the two sets of vectors.
- Availability:
 - No.

- Sabancı embeddings (Sen&Erdoğan, 2014)
 - Method: word2vec with Skip-Gram (Mikolov et al. 2013)
 - Raw corpus:
 - BOUN web corpus (news and texts from some web sites): ~470M tokens; a dump of turkish wikipedia: 52M tokens
 - Preprocessing:
 - Non-textual tokens removed
 - Stemmed - morphemes also included
 - Deasciified
 - Lowercased
 - Vocabulary: 380K (including 25K morphemes)
 - $|w| = 100-700$ (depending on the task, the best size is either 300 or 400).
 - Evaluation: Qualitative. (Only quantitative method and hyperparameter comparison).
 - Availability:
 - <http://myweb.sabanciuniv.edu/umutsen/research/>
 - No explicit licence restriction

Problems

- Out of vocabulary words
- Generalization

Future Directions

- Char-based models
- Morpheme embeddings

todo

- Collect sets with different content (rates) for different purposes
 - At least as possible
- Produce embeddings with different hyperparameters
- Evaluate the embeddings
 - On a word similarity task
 - On a general purpose task like sentiment analysis or NER, whose (baseline) performance is already known.

References

- (Jurafsky&Martin, 2017) Jurafsky, Dan, and James H. Martin. "Vector Semantics". *Speech and language processing*. 3rd Ed - Draft. Pearson, 2017.
- (Mikolov et al. 2013) Mikolov, Tomas, et al. "Efficient estimation of word representations in vector space." *arXiv preprint arXiv:1301.3781* (2013).
- (Pennington et al. 2014) Pennington, Jeffrey, Richard Socher, and Christopher D. Manning. "Glove: Global Vectors for Word Representation." *EMNLP*. Vol. 14. 2014.
- (Turian et al. 2010) Turian, Joseph, Lev Ratinov, and Yoshua Bengio. "Word representations: a simple and general method for semi-supervised learning." *Proceedings of the 48th annual meeting of the association for computational linguistics*. Association for Computational Linguistics, 2010.
- (Schnabel et al. 2015) Schnabel, Tobias, et al. "Evaluation methods for unsupervised word embeddings." *EMNLP*. 2015.
- (Chiu et al. 2016) Chiu, Billy, et al. "How to train good word embeddings for biomedical NLP." *ACL 2016* (2016): 166.
- (Lai et al. 2016) Lai, Siwei, et al. "How to generate a good word embedding." *IEEE Intelligent Systems* 31.6 (2016): 5-14.
- (Baroni et al. 2014) Baroni, Marco, Georgiana Dinu, and Germán Kruszewski. "Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors." *ACL* (1). 2014.
- (Levy et al. 2014) Levy, Omer, Yoav Goldberg, and Ido Dagan. "Improving distributional similarity with lessons learned from word embeddings." *Transactions of the Association for Computational Linguistics* 3 (2015): 211-225.
- (Rehurek&Sojka, 2010) Rehurek, Radim, and Petr Sojka. "Software framework for topic modelling with large corpora." In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. 2010.
- (Sak et al. 2008) Sak, Haşim, Tunga Güngör, and Murat Saraclar. "Turkish language resources: Morphological parser, morphological disambiguator and web corpus." *Advances in natural language processing*. Springer Berlin Heidelberg, 2008. 417-427.

- (Say et al. 2002) Say, Bilge, et al. "Development of a corpus and a treebank for present-day written Turkish." *Proceedings of the eleventh international conference of Turkish linguistics*. 2002.
- (Mikolov et al. 2013a) Mikolov, Tomas, et al. "Distributed representations of words and phrases and their compositionality." *Advances in neural information processing systems*. 2013.
- (Kim, 2014) Kim, Yoon. "Convolutional neural networks for sentence classification." *arXiv preprint arXiv:1408.5882* (2014).
- (Santos et al. 2014) Dos Santos, Cícero Nogueira, and Maira Gatti. "Deep Convolutional Neural Networks for Sentiment Analysis of Short Texts." *COLING*. 2014.
- (Lample et al. 2016) Lample, Guillaume, et al. "Neural architectures for named entity recognition." *arXiv preprint arXiv:1603.01360* (2016).
- (Levy&Goldberg, 2014) Levy, Omer, and Yoav Goldberg. "Dependency-Based Word Embeddings." *ACL* (2). 2014.
- (Al-Rfou et al. 2013) Al-Rfou, Rami, Bryan Perozzi, and Steven Skiena. "Polyglot: Distributed word representations for multilingual nlp." *arXiv preprint arXiv:1307.1662* (2013).
- (Bengio et al. 2009) Bengio et al., "Curriculum learning". *ICML*. 2009.
- (Okur et al. 2016) Okur, Eda, Hakan Demir and Arzucan Özgür, "Named Entity Recognition on Twitter for Turkish using Semi-supervised Learning with Word Embeddings", LREC. 2016.
- (Demir&Özgür, 2014) Demir, Hakan, and Arzucan Özgür. "Improving named entity recognition for morphologically rich languages using word embeddings." *Machine Learning and Applications (ICMLA), 2014 13th International Conference on*. IEEE, 2014.
- (Önal&Karagöz, 2015) Önal, Kezban Dilek and Pınar Karagoz, "Named Entity Recognition from Scratch on Social Media", ECML-PKDD, MUSE Workshop, pp2-17, September 2015.
- (Sen&Erdoğan, 2014) Sen, Mehmet Umut, and Hakan Erdogan. "Learning word representations for Turkish." *Signal Processing and Communications Applications Conference (SIU), 2014 22nd*. IEEE, 2014.