Natural Language Processing

Final Term Project

# SVD Based Methods for Word Embedding

*Author :*
Ehsan Mahmoudi
97543012

*Instructor:*
Dr. Shamsfard

February 11, 2019

## Contents

## List of Figures

## List of Tables

## 1 Abstract

In recent years there has been significant increase in use of word embedding methods for use in NLP applications. Amongst those methods neural network based methods are the most popular ones. Although these methods are very popular these days we believe that still SVD methods still are quiet interesting methods to be investigated. In theory, it has been shown that method like Skip Gram with Negative Sampling (SGNS) is equivalent to performing SVD on shifted PPMI matrix. Although in English language SVD based methods has underperformed compared to SGNS methods. In this project we are going to elaborate same thing in Persian language. The main ideas in this project are based on works done by [4] [2] [1].

## 2 Introduction

In recent years there been a significant interest in word embedding methods. Among all methods the Word2Vec [3] has gained significant attention in the community. This attention is in way that almost Word2Vec methods (both Skip Gram and CBOW) became the de-facto standards of these field. In [1] The authors show a very interesting property on SGNS, that it is equivalent of performing matrix factorization over shifted PPMI matrix. Although in practice the implicit matrix factorization never outperform over SGNS, but at least gives us the clue that, methods with matrix factorization approach are still worthwhile to be explored.

### 2.1 Persian Language Embeddings

Unfortunately very few is done in Word Embedding evaluation in Persian language. This includes standard datasets and benchmarks to show were we are standing, regarding the performance of embedding methods. Before this project we started a class assignment project to build up the grounds for evaluating the existing models. This activity included following:

- Build standard web scale corpora
- Build datasets to assess the embedding models for various tasks (Analogy, Categorization, Word Similarity)
- Write Scripts to evaluate the models against the datasets.

Table 1: Best result for Analaogy Task

| Category | Total Tokens | fastext-skipgram-100 |
|---|---|---|
| semantic-capitals | 4691 | 3514 |
| semantic-Whole to part | 420 | 68 |
| semantic-family | 600 | 349 |
| semantic-currency | 6006 | 1298 |
| syntactic-Comparative Adjectives | 2756 | 1927 |
| syntactic-antonym | 506 | 239 |
| syntactic-Superlative Adjectives | 2756 | 1611 |
| syntactic-verb | 1964 | 708 |
| Total | 19699 | 9714 |

As result of the same project we aimed to build models to gain the highest scores in various tasks. But unfortunately the results were significantly below what we expected.

Compared to what **CITATION NEEDED** This result is one of the best results in Persian language. In that article the highest analogy performance reported is which is around 49. Although this result has been acquired because of high tolerance value in analogy task. It accepts the answer if it is in top 50 closest neighbours of the target vector.

This means in Persian language the baseline is much lower compared to achievements reported in English literature. This might have two main reasons. Either the available resources in Persian are not as rich as resources in English and or the structure of language requires different approach, for modelling.

In this project our aim is to explore the matrix factorization methods to see, if we can reach a higher ground compared to what has been achieved in other tasks.

## 2.2 Word Embedding as Matrix Factorization

In [1] it is shown that SGNS is equivalent to perform matrix factorization on following matrix:

$$M^{PPMI}(i,j) = max(0, PMI(w_i, w_j) - k) \tag{1}$$

Where

$$PMI(w_i, w_j) = log\left(\frac{P(w_i, w_j)}{P(w_i)P(w_j)}\right) \tag{2}$$

Once we constructed the matrix $M^{PPMI}$ then we can perform Singular Value Decomposition on it to get the embedding vectors:

$$M^{PPMI} = U\Sigma V^T \tag{3}$$

Now we can assume that $W^{WORD} = U\Sigma$ is going to be our embedded vectors.

## 3 Methodology

In this section we describe a little more technical details of the project that most be considered for successful implementation.

### 3.1 Sparsity

One of the items that are vital for this project to be successful is to keep the sparsity of the matrixes. The start of the work is from building cooccurrence matrix. And this matrix is sparse by nature. But the way that we implement the code is important as this will significantly will impact the performance of the algorithm. And if not implemented properly it might even make it impossible to run our algorithm in such scale.

We used python language and utilised *SciPy* library because it supports implementation of sparse matrixes and also sparse algorithms of SVD methods.

## 4 Results

## 5 Discussions and Conclusion

## References

[1] Omer Levy and Yoav Goldberg. Neural word embedding as implicit matrix factorization. In Z. Ghahramani, M. Welling, C. Cortes, N. D.

Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2177–2185. Curran Associates, Inc., 2014.

[2] Omer Levy, Yoav Goldberg, and Ido Dagan. Improving distributional similarity with lessons learned from word embeddings. *TACL*, 3:211–225, 2015.

[3] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc., 2013.

[4] Roser Morante and Wen-tau Yih, editors. *Proceedings of the Eighteenth Conference on Computational Natural Language Learning, CoNLL 2014, Baltimore, Maryland, USA, June 26-27, 2014*. ACL, 2014.

## A Appendix