Final Project

Detection of Negative Reviews in Online Stores

Team #8

Mikhail Sidorenko, Egor Baryshnikov, Roman Teplykh

Table of Contents

Abstract

The project is devoted to the sentiment analysis of the reviews in online stores. Different methods of reviews/documents embeddings training were analyzed. Special attention was paid to the methods of word embeddings training. We tried out three different methods of word embeddings training - SVD over SPPMI (implicit matrix factorization), SGNS-RO (SGNS with the framework of Riemannian optimization) and SGNS-EMF (explicit matrix factorization). All of these methods were implemented from a scratch, using algorithms from the corresponding papers. The SGNS-EMF method was supposed to have the best performance in linguistic tests and in classification problem because this method isn't based on the assumption of independence of word-context pairs. But, actually, this method yielded the poorest results in all tests. The reason for it may be in bad choice of hyperparameters for this method. The SGNS-RO method yielded the best results in both tests. The possible directions of further work is presented in the last section.

All implemented methods and Jupyter notebooks with the results of experiments can be found in our GitHub repository. Link: https://github.com/Bullldoger/NLA-Project

1. Introduction

Opinions play crucial role in all humans activities because they majorly define our behavior. Usually, when one has to make a decision, she/he wants to know opinions of other people. For companies knowing opinions of their consumers/clients is also important because it may help to understand their needs. Before the advent of the Internet, people used their families and friends as a source of opinions and companies conducted many surveys in order to know consumer opinions.

Nowadays, all necessary information can be found in the Internet on forums, social networks, websites etc. The problem is that there is a huge volumes of textual information to be processed to understand opinions of a particular product and for a human it may be difficult to analyze all of it by reading every review. Sentiment analysis/opinion mining methods can be applied to solve problem review classification in positive/negative classes.

Almost all existing methods of sentiment analysis are described in [1]. The common approach (Fig. 1.1) in reviews classification problem is to create representations of words as vectors in some space, then to get a representation for the whole review/document and to use gathered representations for classification[2, 3, 4].

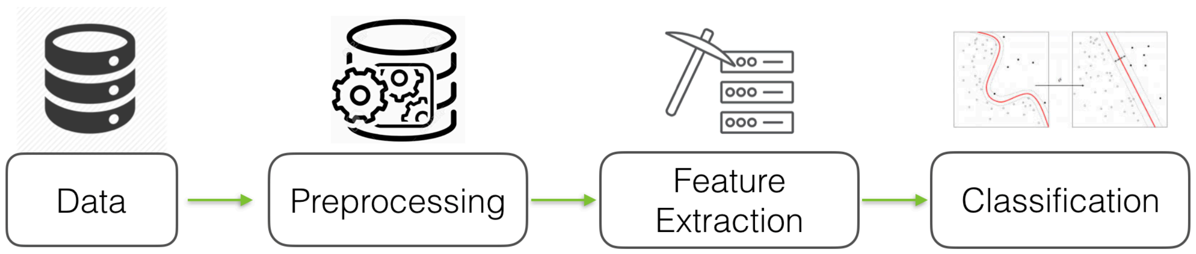


Figure 1.1. The common workflow in sentiment analysis

In sentiment analysis the stage of creation of word/document representations is important as it defines the success of a solution (bad representations will lead to bad results). A lot of different approaches to creating word embeddings have been proposed recently [6, 7, 8, 9] and we decided to analyze them and compare the results obtained by applying a particular method in a review classification problem. The idea is to treat with sentiment analysis problem as a supervised learning problem: we have a dataset with many reviews, each review is an observation and has a label «positive/negative». Each observation is described by a vector of real numbers.

2. Methodology

2.1 Data

For our purposes we use data from the collection of Amazon reviews on different items (available here http://jmcauley.ucsd.edu/data/amazon/). In order to reduce computation time, we chose 5-cores of reviews on items from «Cell Phones and Accessories» category (about 194k reviews).

Each review is presented in JSON format with 9 fields (reviewerID, asin, reviewerName, helpful, reviewText, overall, summary, unixReviewTime ,reviewTime). Fields of interest are *reviewText* (contains review itself) and *overall* (contains user's rating of a product, integers from 1 to 5). In order to extract this information from the raw data, we used JSON parser.

For supervised learning problem each observation must have a label. In order to simplify and accelerate labeling procedure, we used simple strategy based on user's ratings: if the rating less or equal than 3, than the corresponding review is negative (class 0), otherwise, the review is positive (class 1). As a result, we got about 80% of positive reviews and 20% of negative ones.

2.2 Preprocessing

All considered word2vec models are based on the co-occurence word-context matrix, thus at the preprocessing stage the vocabulary of words occurred in reviews and the co-occurence word-context matrix were created. When creating the vocabulary, we took into account only those words, which occur in the corpus more than 200 times. The vocabulary had size 3723.

Contexts of a word in a text are its L-nearest words (L-sized sliding window). We used $L = 2$.

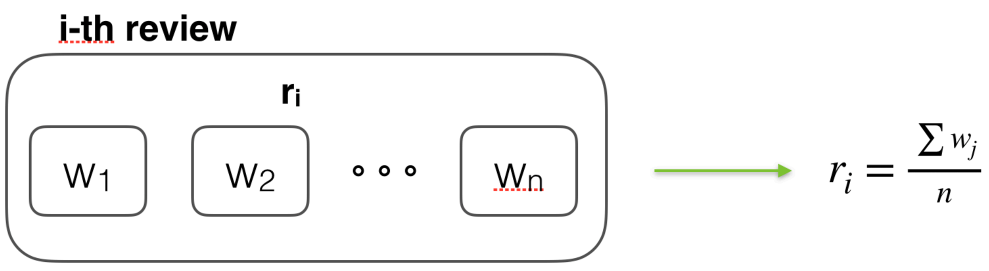


Figure 2.1. Embedding for the ith review, which consists of n words

2.3 Feature Extraction

For each review we are going to create an embedding - vector of real numbers. To do so, we, first of all, created word embeddings, using different methods. As words embeddings were created we can create a reviews embeddings, and to do it we just find the «average» vector of each review as shown in Figure 2.1. When creating review embeddings, we skipped stop words.

We tried out different methods of word embeddings creation, that are described below in a nutshell. The starting point for these methods is the skip-gram neural embedding model trained using the negative-sampling procedure [10, 11]. As an optimization problem it can be written in the following way:

$$\mathcal{L} = \sum_{w \in \mathcal{V}} \sum_{c \in \mathcal{V}} \left(\#(w,c) \log \sigma(\mathbf{w}^\top \mathbf{c}) + k \cdot \mathbb{E}_{c' \sim P_{\mathcal{D}}} \log \sigma(-\mathbf{w}^\top \mathbf{c}) \right) \rightarrow \max_{\mathbf{w},\mathbf{c} \in \mathbb{R}^d}, (1)$$

where $\mathcal{L}$ is an objective and $\mathbf{w}, \mathbf{c} \in \mathbb{R}^d$ are d-dimensional vector representations of a word and a context, respectively. The problem is solved by stochastic gradient descent.

<u>*SPPMI-SVD*</u>

In [6] it was shown that under assumption of independence of values of $\mathbf{w}^T\mathbf{c}$ for different word-context pairs the objective (1) can be transformed to the following:

$$\mathcal{L} = \sum_{w \in \mathcal{V}} \sum_{c \in \mathcal{V}} f(w,c) = \sum_{w \in \mathcal{V}} \sum_{c \in \mathcal{V}} \left(\#(w,c) \log \sigma(\mathbf{w}^\top \mathbf{c}) + \frac{k \cdot \#(w) \cdot \#(c)}{|\mathcal{D}|} \log \sigma(-\mathbf{w}^\top \mathbf{c}) \right) \rightarrow \max_{\mathbf{w},\mathbf{c} \in \mathbb{R}^d} \quad (2)$$

It turns out that this objective depends only on $\mathbf{w}^T\mathbf{c}$ and, therefore, analytical solution of (2) can be obtained:

$$M_{wc} = \mathbf{w}^\top \mathbf{c} = \log \left(\frac{\#(w,c) \cdot |\mathcal{D}|}{k \cdot \#(w) \cdot \#(c)} \right) = \log \left(\frac{\#(w,c) \cdot |\mathcal{D}|}{\#(w) \cdot \#(c)} \right) - \log k = \mathrm{PMI}(w,c) - \log k, (3)$$

where PMI is point wise mutual information matrix.

It was shown that the quality of words' representation can be significantly improved by using shifted positive PMI matrix instead of (3):

$$\mathrm{SPPMI}(w,c) = \max \left(\mathrm{PMI}(w,c) - \log k, 0 \right). (4)$$

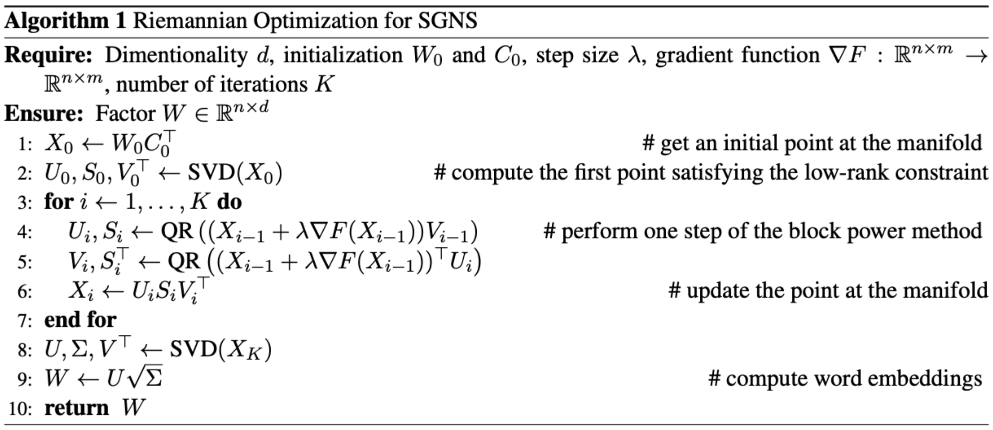
In order to find embeddings as low dimensional vectors, truncated SVD decomposition of SPPMI matrix is computed. It can be presented in following way:

$$U\Sigma V^{\top} \approx \text{SPPMI}, W = U\sqrt{\Sigma}, C = \sqrt{\Sigma}V^{\top}, (5)$$

where W and C are matrices of word and context embeddings.

<u>*SGNS-RO*</u>

In [7] there was proposed alternative approach to creation of word embeddings by SGNS model. It was shown that word embeddings training can be presented as two steps: 1) Search for a low-rank matrix X that provides a good SGNS objective value; 2) Search for a good low-rank representation $X = WC^T$ in terms of linguistic metrics, where $W$is a matrix of word embeddings and C is a matrix of so-called context embeddings. Word embeddings training model proposed in [10, 11] do not take into account that the objective (1) depends on the X and it's possible to compute derivatives w.r.t. X instead of computing derivatives w.r.t. W and C separately. The model proposed in [6] follows these steps, but it turns out that such approach avoid solving original optimization problem and solve surrogate optimization problem.



Algorithm 1 Riemannian Optimization for SGNS

Require: Dimentionality d, initialization W_0 and C_0, step size λ, gradient function $\nabla F : \mathbb{R}^{n\times m} \to \mathbb{R}^{n\times m}$, number of iterations K

Ensure: Factor $W \in \mathbb{R}^{n\times d}$

```
1:  $X_0 \leftarrow W_0 C_0^{\top}$                                    # get an initial point at the manifold
2:  $U_0, S_0, V_0^{\top} \leftarrow \text{SVD}(X_0)$                  # compute the first point satisfying the low-rank constraint
3:  for $i \leftarrow 1, \dots, K$ do
4:      $U_i, S_i \leftarrow \text{QR}\left((X_{i-1} + \lambda\nabla F(X_{i-1}))V_{i-1}\right)$      # perform one step of the block power method
5:      $V_i, S_i^{\top} \leftarrow \text{QR}\left((X_{i-1} + \lambda\nabla F(X_{i-1}))^{\top}U_i\right)$
6:      $X_i \leftarrow U_i S_i V_i^{\top}$                             # update the point at the manifold
7:  end for
8:  $U, \Sigma, V^{\top} \leftarrow \text{SVD}(X_K)$
9:  $W \leftarrow U\sqrt{\Sigma}$                                        # compute word embeddings
10: return $W$
```

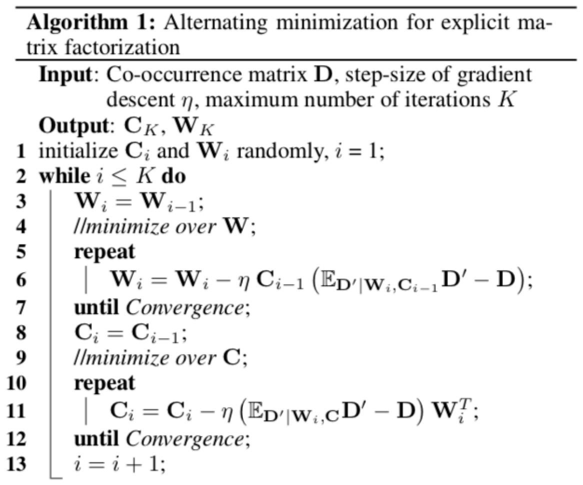
Figure 2.2. Algorithm of SGNS-RO method [7]

The idea of the proposed SGNS-RO method is to optimize the original SGNS objective directly (with the framework of Riemannian optimization) while keeping the low-rank matrix search setup on Step 1. The proposed algorithm is shown in Figure 2.2. Experiments showed that this method outperforms the SPPMI-SVD method in many linguistic tasks, thus it may be interesting to try it on the sentiment analysis problem.

SGNS-EMF

Two above mentioned methods have a common significant constraint - they are based on the assumption that values of $\mathbf{w}^T\mathbf{c}$ are independent for different word-context pairs, but this is not, actually, the case.

The approach, which avoids those constraints, was proposed in [8]. The main idea of it is to consider SGNS in the perspective of representation learning and explicit matrix factorization (EMF): it was proved that the matrices W and C can be iteratively obtained from co-occurence matrix D by the algorithm shown in Figure 2.3.



```
Algorithm 1: Alternating minimization for explicit matrix factorization
Input: Co-occurrence matrix $\mathbf{D}$, step-size of gradient
       descent $\eta$, maximum number of iterations $K$
Output: $\mathbf{C}_K$, $\mathbf{W}_K$
initialize $\mathbf{C}_i$ and $\mathbf{W}_i$ randomly, $i = 1$;
while $i \leq K$ do
    $\mathbf{W}_i = \mathbf{W}_{i-1}$;
    //minimize over $\mathbf{W}$;
    repeat
        $\mathbf{W}_i = \mathbf{W}_i - \eta\, \mathbf{C}_{i-1} \left( \mathbb{E}_{\mathbf{D}'|\mathbf{W}_i,\mathbf{C}_{i-1}} \mathbf{D}' - \mathbf{D} \right)$;
    until Convergence;
    $\mathbf{C}_i = \mathbf{C}_{i-1}$;
    //minimize over $\mathbf{C}$;
    repeat
        $\mathbf{C}_i = \mathbf{C}_i - \eta \left( \mathbb{E}_{\mathbf{D}'|\mathbf{W}_i,\mathbf{C}} \mathbf{D}' - \mathbf{D} \right) \mathbf{W}_i^T$;
    until Convergence;
    $i = i + 1$;
```

Figure 2.3. Algorithm of SGNS-EMF method [8]

Conducted experiments showed that SGNS and EMF are equivalent.

These three methods were implemented from scratch in Python 3.7 as a class Word2Vec in module models.py (https://github.com/Bullldoger/NLA-Project/tree/master/notebooks). In the following section we will compare the performance of these methods on a linguistic test and in classification problem.

Also it should be mentioned, that the implementation of the SGNS-EMF is very slow. It may be because of we didn't manage to find good conditions of convergence for two internal cycles.

2.4 Classification

To solve the classification problem we used two simple out of the box classifiers from *sklearn* library - Support Vector Machines (SVM) and Logistic Regression. As a quality metric of predictions we used f1-score metric because the dataset was imbalanced - number of observations of the first class was higher than for the second.

3. Results

First of all, we compared the performance of those three methods on simlex999 dataset with different values of hyperparameter d - dimension of embedding vectors. The results obtained are presented in Table 1 as values of Spearman's correlation coefficients between predicted similarities and the manually assessed ones. In this experiment SGNS-RO method had the highest score, and SGNS-EMF had very low score. Maybe this is due to the problems with implementation, that were mentioned above. The highest similarity score was observed for d=500.

Table 1. Spearman's correlation between predicted similarities and the manually assessed ones (k = 5, alpha=0.5), simlex999 dataset

d=100	SVD-SPPMI	0.13284
	SGNS-RO	**0.13466**
	SGNS-EMF	0.03252
d=200	SVD-SPPMI	0.12277
	SGNS-RO	**0.13051**
	SGNS-EMF	0.06966
d=500	SVD-SPPMI	0.18781
	SGNS-RO	**0.18920**
	SGNS-EMF	0.06119

In two following experiments we compared the values of the SGNS objective for different values of hyperparameters d and k (number of negative samples). When k=5 the highest values of the objective are obtained by SGNS-RO method (Table 2). The best performance is yielded when d=100, that contradicts the results of previous experiments.

For different values of k we got nearly the same result - SGNS-RO method has the best performance (Table 3).

Table 2. The values of SGNS objective function at the optimal point (all values are multiplied by 10^-9, k=5, alpha=0.5)

	SVD-SPPMI	SGNS-RO	SGNS-EMF
d=100	-0.2383	**-0.2321**	-0.3841
d=200	-0.2381	**-0.2316**	-0.5406
d=500	-0.2357	**-0.2300**	-0.8484

Table 3. The values of SGNS objective function at the optimal point (all values are multiplied by 10^-9, d=200, alpha=0.5)

	SVD-SPPMI	SGNS-RO	SGNS-EMF
k=1	-0.0758	**-0.0742**	-0.3467
k=5	-0.2381	**-0.2316**	-0.5406
k=15	-0.6354	**-0.6157**	-0.6779

In the final experiment we compared F1-score metric values for two classifiers and for different values of hyperparameter d. For high values of d the best representations of the reviews are obtained by SGNS-RO method for both classifiers. Although, for d=100 the SPPMI-SVD method shows the best quality of classification.

It should be mentioned that the classifiers weren't tuned and the quality of classification after tuning may be even higher.

Table 4. F1-score values (k=5, alpha=0.5)

		LR	SVC
d=100	SVD-SPPMI	**0.87892**	**0.87901**
	SGNS-RO	0.87890	0.87888
	SGNS-EMF	0.86754	0.86849
d=200	SVD-SPPMI	0.88341	0.88338
	SGNS-RO	**0.88345**	**0.88341**
	SGNS-EMF	0.87446	0.87492
d=500	SVD-SPPMI	0.89012	0.89016
	SGNS-RO	**0.89019**	**0.89023**
	SGNS-EMF	0.88568	0.88580

Also it was estimated that the SGNS-EMF method with values of step for GD more than 10^{-7} doesn't converge or converge very slow with oscillations.

Further Work

As a further direction for this work we can, first of all, have an another attempt on finding out why the SGNS-EMF method has such poor performance. We can try to accelerate this algorithm by using, for instance, adaptive steps for GD.

The research of application of more complex methods for training reviews/documents embeddings is also needed. We could try to use methods, based on recurrent neural networks, that were described in [1].

Contribution of team members

Mikhail Sidorenko: analysis of papers, implementation of SGNS-RO and SGNS-EMF methods, creating of the presentation, creating of the report.

Egor Baryshnikov: conducting experiments and analysis of its results, help with the presentation creation, analysis of papers.

Roman Teplykh: analysis of papers, implementation of SPPMI-SVD (baseline), data preprocessing, help with the report, GitHub repository creation.

References

1. Lei Zhang, Shuai Wang, Bing Liu, 2017. «Deep Learning for Sentiment Analysis: A Survey»
2. Tang D, Qin B, Liu T. Document modelling with gated recurrent neural network for sentiment classification. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2015), 2015.
3. Tang D, Qin B, Liu T. Learning semantic representations of users and products for document level.
4. Chen H, Sun M, Tu C, Lin Y, and Liu Z. Neural sentiment classification with user and product attention. In sentiment classification. (ACL 2015), 2015.
5. Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2016), 2016.
6. Levy O., Goldberg Y., 2013. «Neural Word Embeddingas Implicit Matrix Factorization»
7. Alexander Fonarev, Oleksii Hrinchuk, Gleb Gusev, Pavel Serdyukov, Ivan Oseledets, 2017. «Riemannian Optimization for Skip-Gram Negative Sampling».
8. Yitan Li, Linli Xu, Fei Tian, Liang Jiang, Xiaowei Zhong and Enhong Chen, 2015. «Word Embedding Revisited: A New Representation Learning and Explicit Matrix

Factorization Perspective», Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI 2015).

9. Andrew J. Landgraf1, Jeremy Bellay, 2017. «word2vec Skip-Gram with Negative Sampling is a Weighted Logistic PCA».
10. Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. CoRR, abs/1301.3781, 2013.
11. Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In NIPS, 2013.