Delft University of Technology

WI4231 Mathematical Data Science

LSI with Donald Trump's tweets

Project Report

Erik Jansson	4985370
Tuomas Koskinen	4989325
Ville Kujala	4991974
Emanuel Ravemyr	4989503
Federica Trevisan	4988078

April 11, 2019

1 Summary

In this paper, an analysis of Donald Trump's tweets is presented. The analysis was done with the use of the Latent Semantic Indexing (LSI) Algorithm, that has been implemented and applied to the preprocessed data set. The results are presented in a qualitative manner for selected queries, which are compared and analysed for different values of k. It was found that k-values in the low hundreds seemed to yield good results which did not only contain direct word matches, but also some matches that belong to a context without the words from the query. This indicates that LSI can be very useful for natural language processing. However, it is the opinion of the authors that the used data set is not sufficiently large for the method to be optimal.

2 Theory

The LSI algorithm is used in language processing. It builds on the idea that semantically adjacent words will appear in similar way in the text. It is based on the singular value decomposition (SVD) of a term-document matrix. The idea is to project the term-document matrix into a new space defined by the SVD of that term-document matrix, which then allows for querying vectors from this new space and for example seeing which documents or words are close to each other in the space.

The first step in the algorithm is to, given the text-chunks, construct some sort of matrix that allows to determine how and where different words occur in the text. This matrix is known as the term-document matrix, and can for instance be a tf-idf matrix, or, in this case an occurance matrix. This matrix will be sparse, however, and the overall goal of LSI is to find, using SVD, an approximation of this matrix.

Since this matrix is sparse, special methods are required in order to make the SVD computation efficient. The scipy library in Python has an implementation of a sparse SVD algorithm that uses the Arnoldi iteration method, which is a Krylov subspace method. Using this method it is possible to obtain the singular value decomposition, that is to say, the matrices,

$$X = U\Sigma V^T, \tag{1}$$

where Σ is a diagonal matrix with the singular values of X on the diagonal U and V^T are orthonormal matrices, where U are the eigenvectors of XX' and V the eigenvectors for $X'X$. The basis for LSI is to truncate the obtained matrices by choosing only the k first singular values. In practice this is done the other way around, since this is necessary for the chosen algorithm. After this "query" vectors, for instance rows in the score matrix X, may be projected into an euclidean space. Apart from the semantical gains, one reason for doing this is to reduce noise. The projections are done by

$$q_k = \Sigma_k^{-1} U_k^T q, \tag{2}$$

where q is the vector to be projected, q_k is the projection and Σ_k and U_k are the k-column versions of Σ and U [3]. With a way to get coordinates in the k-dimensional euclidean space for any vector of terms it is possible to for example see which terms or vectors of terms are close together or similar in some sense. In order to compare the resulting approximations, cosine similarity is often used, that is to say, $S = \frac{\langle a,b \rangle}{\|a\| \ \|b\|}$ is chosen as the similarity measure.

3 Method

In this paper, the used data are tweets produced by president of USA, Donald Trump in the year 2016 [1]. The tweets were downloaded and then preprocessed before any algorithms was applied [2] to remove common words as well as undesired letters and signs (such as @ and #). After the preprocessing the data consists of about 4000 tweets and 5000 unique words.

Instead of using a tf-idf matrix as a base for the LSI, an occurrence matrix X is used. In this matrix every row represents a tweet and every column represents a word. x in position ij indicates that word j appears in tweet i x times. This approach was chosen since taking the SVD of the tf-idf matrix, some of the values become very small, which caused floating point errors in the implementation used in this paper.

A truncated SVD is calculated using the Arnoldi iterations method, where only the k biggest singular values are chosen. To find values of k which give small errors between the resulting SVD matrix and the original matrix, the Frobenius norm between the two was calculated for a sequence of k-values. The result of this can be seen in Figure 1. However, small errors in the Frobenius sense did not seem to guarantee good performance in the linguistic sense from LSI, see [3]. This is why certain aspects of the analysis of results are done for k's which result in a larger error than one would expect from Figure 1.

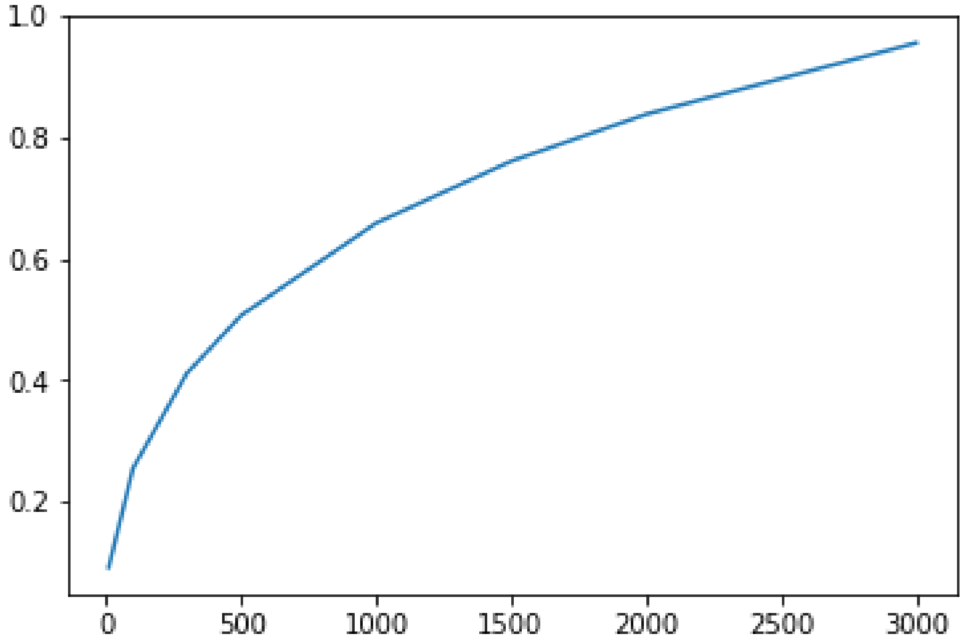

Figure 1: The 1-error in Frobenius norm for different values of k

When comparing query vectors, that is to say, words or sentences, with tweets, one often wishes to find the closest tweet. In order to do this, the euclidean distance on the resulting normalised k-dimensional vectors was used as measure. A very efficient approach, given that setting, is to use the data structure KD-tree, which essentially is a binary tree meaning that the resulting ranking will be very fast. An implementation for this is found in the SK-learn package, but it does not support the cosine similarity distance; however, after normalising the resulting vectors, it turns out that the resulting ranking is the same when using the Euclidean distance on normalised vectors [3]. This approach was found to be faster than creating obtuse distance matrices in the naive way.

3.1 Results for querying

As the results of LSI are hard to interpret directly from the k-space vectors, being simply distances and vectors in a k-dimensional Euclidean space with no inherent semantic meaning, the resulting closest vectors were mapped back onto tweets. In this paper the performance of LSI was determined by qualitatively deciding whether the closest neighbours to a tweet makes sense in a contextual way, for a set of queries. Furthermore, the qualitative analysis is done for several different values of k in an effort to decide on a suitable value. This is computationally costly, as it requires us to compute the SVD several times, but considering that the size of the input matrix X is relatively small, it was found to be a reasonable approach.

In this paper, this is done by analysing the 20 closest neighbouring tweets (in the k-space sense) for five different queries, which were selected based on the the authors' expectation on the words to belong to different topics and contexts. In addition, it was decided that five different values of k would be tested, with most of them being of order $\sim 10^2$, as these values seemed to yield better accuracy in the initial tests when compared to larger values of k. The selected queries were: "God Terrorism", "Heroin Epidemic", "Son Daughter", "Europe NATO", "Russia Putin" as they were considered to be sufficiently different while being rather frequently occurring throughout the data set.

4 Results

For low values of k (10) it was found that the results were essentially random, which is as expected. The nearest tweets contained word matches, or matches that did not contain any obvious link to the words in the query. This indicates that using such low values does not yield very good solutions in a text context sense. Conversely, high values of k (800) seemed to mostly return tweets with direct word matches. While this is not necessarily something to avoid, one would be more pleased to find contextual matches rather than direct word matches. For example, the queries with "God Terrorism", "Europe NATO", "Son Daughter" almost only contain tweets containing words from the queries. The query with "Heroin Epidemic" on the other hand returns several results with focus on Drugs and border safety, topics which relate to the words in the query, without containing any direct references to them. This might be because the words in this query are less frequent than the words in the different queries, but is certainly an interesting result.

For even larger k(3500), the results seemed to further converge to word matches and tweets seemingly unrelated to the query words. The unrelated matches were distinct in each of the queries and seemed to have some overlap with the clusters that were found in the previous assignment in this course. [2]. This could be due to more and more of the noise being included in the k-dimensional approximation space. For this reason, it was chosen to focus on k in range $100 - 800$.

The queries in this interval seem to return many word matches, as well as a significant amount of contextual matches. For example, in the query "God Terrorism" many tweets are found in which Trump refers to attacks which clearly is related to the topic of terrorism.

In the query with "Heroin epidemic" the contextual matches are even more prevalent, as the tweets to a large extent refers to "Drug problems" as well as immigration and border security. In fact there are very few word matches for this query which could indicate that Trump considers the issues with drugs in the US to be a consequence of the lack of a strong southern border, but one should be reminded that this is a limited setting and no huge conclusions can be drawn from these experiments.

The "Son daughter" query contains several word matches, but also some tweets where the names of Trumps sons and daughters are mentioned, as well as mentions of support and children.

The query on NATO and Europe contains a significant amount of word matches in different forms. One context that can be discerned outside the word matches are tweets related payment and money. This is perhaps not a obvious result, but is most likely connected to the fact that many of Trump's tweets in the data set concerning NATO seem to express the opinion that the US pays a disproportionate amount to the organisation.

The final query, on Russia and Putin is mainly filled with word matches, as well as some mentions of Hilary Clinton. This is probably connected to the presidential campaign as well as Clinton's past as Secretary of State, which is mentioned in many of the tweets. The query also returns tweets with the word Deal, which is often included in tweets where other nations are mentioned, which indicates that it relates to international agreements.

In general, the results that were considered the best were found with k around the low hundreds: [100,300] where there was some trouble to distinguish a "best" value, as the results are similar but just with more or less noise, word matches and context matches, which are hard to weigh against each other.

When some different queries were tried, it was discovered that adding the word "bad" to a query seemed to return mostly tweets related to Clinton, which indicates that her name is very closely related to "bad"

4.1 Word distances

For words *Russia*, *Hillary*, *Drug*, *Immigration*, *Korea* the five closest words in the SVD space were computed for $k = 10, 100, 1000$, results are given in Table 1. These words were picked because they appeared relatively often in the tweets and the results were considered interesting. It can be noticed that the results seem intuitive for the most part. Qualitatively the results given by $k = 100$ seem to be the best. For example, the words mapped closest to *Russia* are *Putin*, *Crimea* and *Ukraine* which all appear in tweets related to the Crimean crisis. *Korea* give us words *Warhead*, *Testing*, *Nuke* due to tweets about North Korea's nuclear weapon testing. The somewhat surprising word *Sweetheart* appears due to the following tweet: "*Russia has more warheads than ever, N Korea is testing nukes, and Iran* *got a sweetheart deal to keep theirs. Thanks, @HillaryClinton.*". Other surprising results have similar explanations. The data is not large enough and as a result individual tweets can cause odd result.

	k = 10	k = 100	k = 500
Russia	dignity humiliating vladimir pushing poison	putin crimea ukraine deflect sniper	crimea fury insane imagination stockpile
Hillary	crooked suffering habit prosecution avoided	sleeping sigh slogan rosie ate	sleeping accomplishment habit nominating refers
Drug	facebook facility flag lord create	stop pouring epidemic deport heroin	pouring epidemic cartel unleash warn
Immigration	ending refreshing way drake theoretical	illegal crossing midwest border heroin	consulting earner altogether suspend region
Korea	survive litany restrict moderator brazilian	sweetheart warhead testing nuke stockpile	sweetheart warhead testing russia stockpile

Table 1: For each selected word and value of k, the 5 closest words in the SVD space.

5 Conclusions

LSI is a good way to map term document matrices into a lower dimensional space to extract meaning and context. A big difference between LSI and other SVD based methods is that minimising the error between the original space and the lower dimensional space does not necessarily seem to provide better results. This could be because the new space becomes overfitted to the original space resulting in word matching rather than context searching.

In literature, most applications use LSI as a precursor to some other classification algorithm, as clustering or some machine learning implementation. As this is not done in this report, it is hard to evaluate the true potential of the algorithm. Another constraining factor in this report is the size of the data set, which is just around 2 million elements and a matrix rank in the low thousands. This is much smaller than the sizes mentioned in literature, which further complicates things. With more data, and time, the LSI approach could for example be used to classify new tweets by Trump (under certain conditions) with a classification algorithm.

References

[1] Trump twitter archive. `https://www.trumptwitterarchive.com/`. Accessed 2019-03-04.

[2] Kujala Ravemyr Trevisan Jansson, Koskinen. Analysis of donald trump's tweets. `https://drive.google.com/file/d/16Kq_A1cyCalQ6G8xqsCE64tLZrojzxbo/view?usp=sharing`, 2019.

[3] Raghavan Manning, Schütze. *Introduction to Information Retrieval*, pages 118–122, 403–419. Cambridge University Press, 2008.