Topic modeling on Twitter messages

Data Science Computer Lab Project 2

András Mátyás Biricz
Benedek Horváth

December 14 2018

Contents

1 Introduction

Let us consider our own social connections. In our childhood our parents took us to the playground, where we got acquainted to other children from the neighbourhood, later we went to school and got to know new people, we work at a company, where people also know each other, we may have friends together whom we do our hobby etc. Through our lifetime we belong to several communities and these groups often have some overlapping but only to a certain extent. It is a good idea to consider social connections as a network. In that case it is evident that there is clustering in this network to a very high extent. Many other real-word networks exhibit a high degree of clustering, however, finding these communities is not a trivial task, and there are many approaches in network science for community-finding algorithms (for details see the excellent review paper of Fortunato [1]).

While representing social connections is natural, network representation is efficiently used in a counter-intuitive way in several other fields such as linguistics. The network of co-occurring words can be though of as a network of close concepts. In these language networks, community finding corresponds to finding related concepts or groups of words, which can be attributed to a certain topic.

Our aim in this work is to do topic modeling on data from one of the most wide-spread social media, Twitter. Topic modeling means to search for widely discussed topics in a huge set of documents. It is performed by finding word clusters from words that co-occur often and their meaning is related to each other. Similarly, document clusters can also be defined: texts in which similar topics with similar weights are represented. In this work the term document refers to one tweet.

Topic modeling can be performed several ways. Here we use two different methods: the probabilistic Latent Dirichlet Allocation (LDA) and the network-based hierarchical Stochastic Block Model (hSBM). Gerlach [2] argues that the network-based approach to topic modeling is superior to probabilistic methods.

2 Materials and Methods

2.1 Latent Dirichlet Allocation (LDA)

LDA is among the most popular methods for topic modeling. It has a Bayesian statistical approach to the problem. Given a prior distribution, it fits a model to the documents and gives the most significant topics with the most significant keywords. The number of document clusters, i.e. topics is an input parameter of the model. The prior distribution that is usually assumed is Dirichlet distribution. One problem with the LDA is that it is highly biased to the Dirichlet prior, which does not match some known properties of real text data, such as Zipf's law. Another problem is the danger of over-fitting that is hard to avoid [2].

For the application of the LDA algorithm on the dataset we used the implementation of the *gensim* python package and its python wrapper for the *mallet* java package[1]. The latter performed much better, the runtime was only 3 minutes for the mid-sized dataset

[1]http://mallet.cs.umass.edu/

(around 220 000 tweets, see later in section 3), while the former needed 90 minutes for the same task. It also has to be mentioned that *mallet* also produced better results (see later in section 4.2). A code implementing LDA topic modeling was available at [4] and the implementation of some related visualisation tools was available at [5]. We used some parts of both codes.

2.2 Stochastic Block Model (SBM)

Stochastic Block Model can be used for topic modeling by representing texts and its words in a bipartite network in which one type of node is document (i.e. tweet or pre-defined groups of tweets) and the other type is word, while the words are connected to all documents they appear in via edges. In this representation searching for topics is identical to community finding in the network. Since SBM is an efficient method for finding clusters in a network, it can be used for finding topics in tweet texts.

The regular SBM has a drawback when applied to large networks. Namely, it cannot be used to find relatively small groups, as the maximum number of groups that can be found scales as $B_{max} = O(\sqrt{N})$, where N is the number of nodes in the network. In order to overcome this issue, we need to introduce a nested SBM. This model also provides a multi-level hierarchical description of the network (thus, this version is usually referred as hSBM, hierarchical Stochastic Block Model). See a meaningful figure of the structure of a nested SBM in Fig. 1. With such a description, we can uncover structural patterns at multiple scales and $B_{max} = O(N\sqrt{N})$.

The non-parametric statistical inference is done by minimizing the description length of the network. It measures the amount of information required to describe the data, it is the informational entropy of the system. Much more details are to be found on this page.[2]

The hierarchical version of this model (hSBM) is implemented by the author of the paper [2] and the source code is available to the public on Github [3]. We used this in our work. It turned out that we had to prepare special-sized datasets to be able to acquire inference of the network. The biggest dataset that could be processed with hSBM was the mid-sized one (see section 3), the runtime for this was around 3 hours and the memory consumption reached a level of 24 GB.

3 Data Acquisition

While working with Twitter data, one of the biggest challenges was to process the huge amount of data. We definitely faced a big data problem, since we had approximately 630 GB of compressed data containing a little more than one billion tweets in *json* format with a lot of metadata (e.g. language, country, geographic coordinates, date and time) in addition to the text of the tweets. These tweets were collected in a certain time interval (from 10/16/2017 to 11/19/2018), without any filtering. A low limit of data availability for this time interval was applied according to Twitter data policy.

The first step of the processing was to filter out non-English tweets and to extract only the text from the *json* files paying attention to also remove the duplicate tweets. For this

[2] https://graph-tool.skewed.de/static/doc/demos/inference/inference.html.

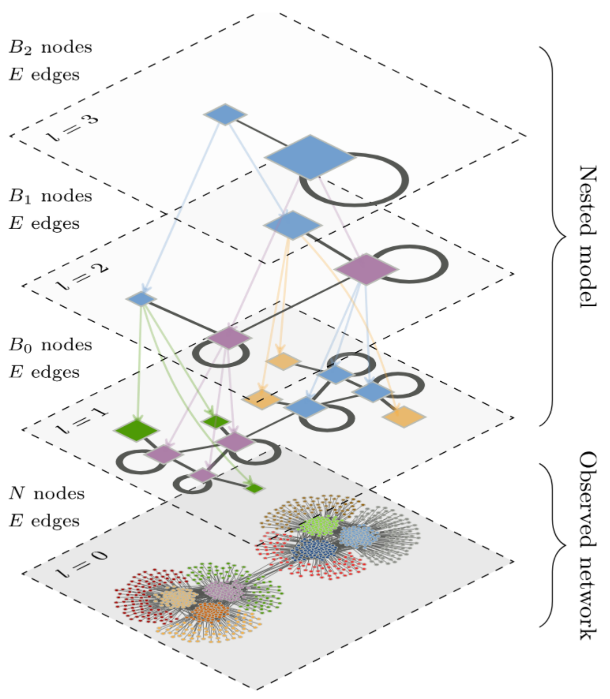

Figure 1: Example of a nested SBM with three levels. Source: https://graph-tool.skewed.de/static/doc/_images/nested-diagram.svg.

we used a python package *polyglot* and the language metadata of the *json* file. Because of the huge size of files they could only be read into the memory and filtered row by row, i.e. tweet by tweet. In order to do so, we had to use buffered file input/output operations and work with compressed files directly. This way one python kernel consumed only around 100 MB of memory. To speed up the process more kernels were launched and the initial partitioning of the data made it possible to reasonably distribute the files to each of the kernels. Running on 4 threads the process took around 10 hours, a processing speed of about 30 MB/s could be reached.

The next crucial step was the language processing: the tweets had to be cleared from junk (links, emoticons, hashtag marks, user names), tokenized (i.e. split into words) and stemmed (multiple grammatical format of one word always has to be considered the same word when building a topic model).

Our first finding was that the processing of all the collected English tweets (around 500 million) would be a huge challenge. When testing the topic modeling algorithms it became clear that only a small portion of the initial data could be fed into the hSBM model (O(100 000) tweets, 0.02%) and that the output was very noisy, no real clustering of words into topics could be made. To overcome this issue we chose 10 different interesting keywords to do additional filtering of the tweet messages. Only the ones were kept that contained at least one of the following keywords:

- nfl,
- hungary,
- climate,
- election,
- physics,
- concert,
- market,
- data mining,
- cinema,
- terror.

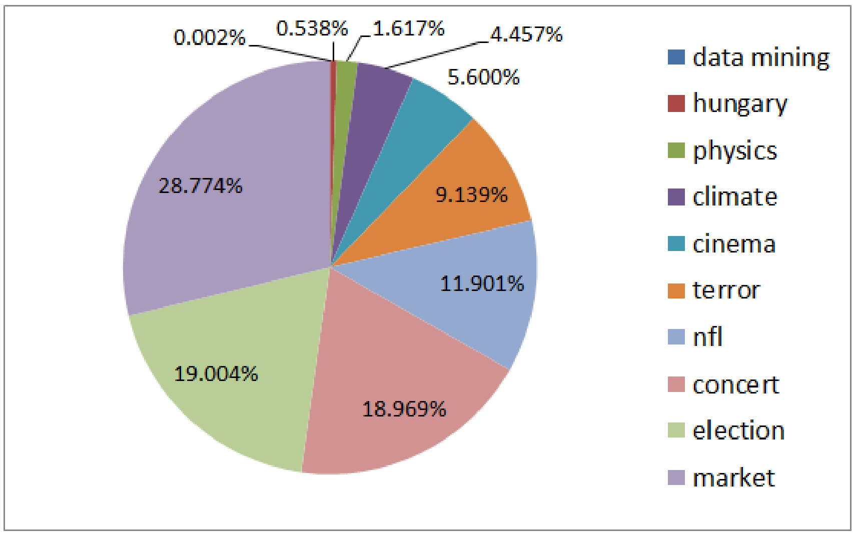

Figure 2: Representation rate of the 10 keywords in the filtered dataset.

When choosing the keywords our aim was to make a less noisy and smaller dataset of tweets that still contained several interesting, different topics. The representation rate of each keyword in the final dataset can be seen in Fig. 2. We modified our language processing code and unified all of the sub-processes mentioned above, but one step was made differently: the filtering of retweets. First, we used a method described on the webpage of Twitter[3] to look for a specific *retweeted status* in the keys of the json file, but then we realised this was not a good way. That way we would filter out messages that had been written as an answer to another tweet. We thought these kind of messages would be better to keep for the topic models. We decided not to do any filtering of retweets in the first round of processing, where we collected the tokenized and stemmed tweets according to the keywords into different files. This complex data acquisition process for all the 1 billion tweets was a demanding task for the computer: it ran more than 20 hours on 8 threads.

The output of the previous process were 10 separated files with tweets filtered according to our keywords. To unify the files and keep the dataset reasonably small for further analysis we made three datasets with different size: one containing all the filtered messages (around 2.7 million) and two other containing one tenth and one hundredth, respectively. It has to be mentioned that this selection was done for each separated file (i.e. keywords) and not after unifying all. This way we kept the original occurance rate of the keywords. Duplicate tweets were considered to be exactly the same messages or ones differing only in the last word. After applying a special algorithm for deleting them, around 85% of the data were kept for the topic models.

[3]https://developer.twitter.com/en/docs/tweets/data-dictionary/overview/intro-to-tweet-json.html

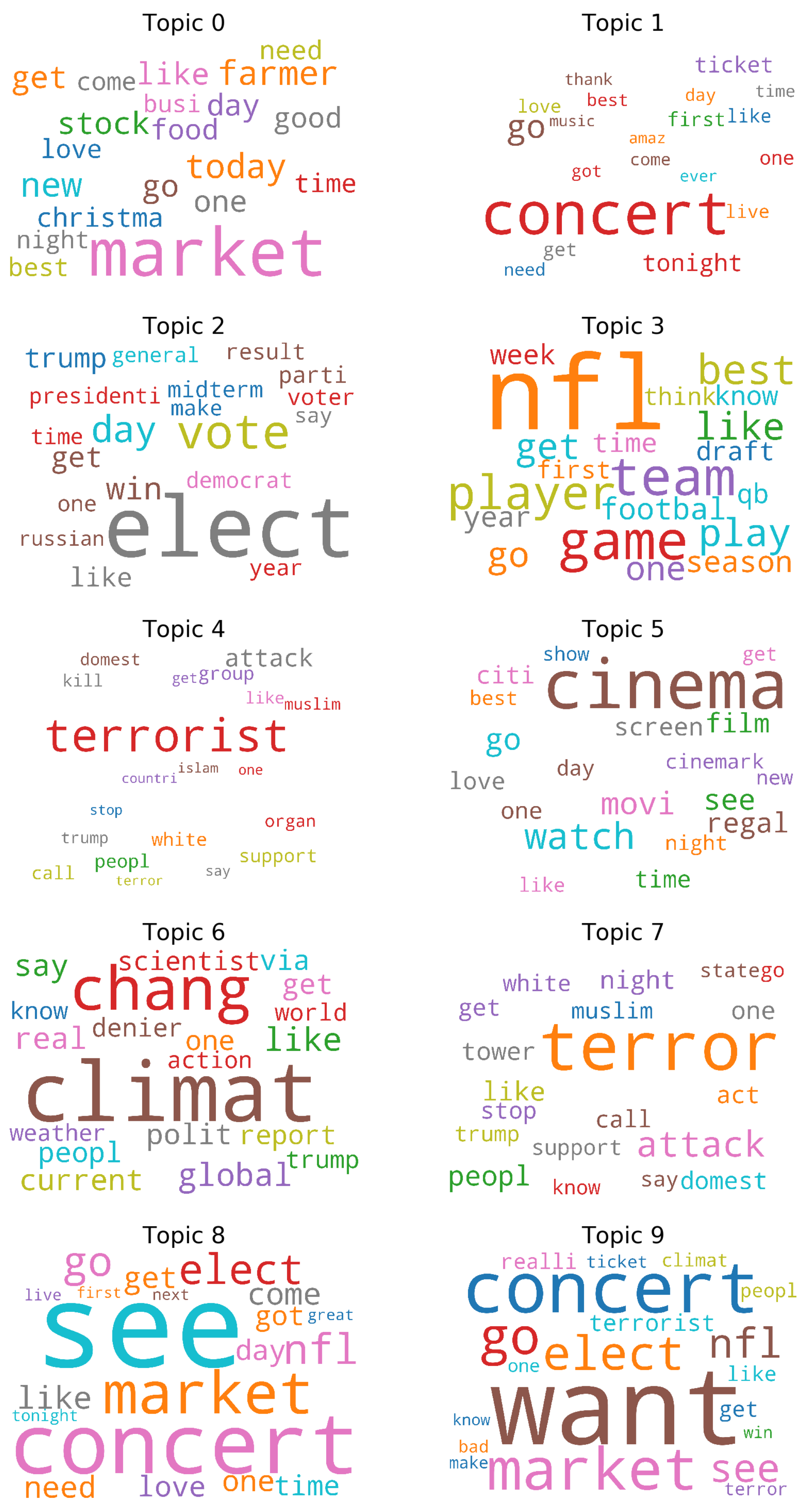

Figure 3: Word cloud visualization of the topics in the document clusters obtained from the first layer (layer 0) of hSBM.

4 Results and discussion

After getting through the data acquisition challenges and running some test calculations we decided to perform our detailed analysis on the mid-size dataset containing 220 000 tweets in total from the 10 specified topics.

4.1 hSBM

The Stochastic Block Model algorithm found 8 well-separated document clusters, i.e. topics. A spectacular visualization of them can be seen in Fig. 3. The figure contains the word clouds of the 10 clusters found by hSBM, i.e. figures that show the most frequent words of a certain topic with sizes proportional to their frequency. Seven topics of the results comply with our preset keywords (*market, concert, election, NFL, terror, cinema, climate*), while the remaining three clusters seem to be repetition of one of these topics (see topic 4 and topic 7, where terrorist and terror are the most represented) or just meaningless mixed categories (see topic 8 and 9, in the last row). Physics, Hungary and data mining do not appear as clusters, because of being more specific topics compared to all the others, thus being less represented among the tweets, as it is clearly visible in Fig. 2. The representation rate of tweets containing these three keywords hardly outnumber 2 % in total. It is also spectacular that the word *change* is very highly represented in the topic of climate: the majority of tweets mentioning the word climate must be about climate change, one of the most important global problems of our world nowadays.

The hSBM model does clustering among words and documents as well and it determines the occurrence rate of word groups in each tweet. However, the clustering of documents did not worked on our data, so we had to find a workaround to classify the topic clusters arising. Using the data of word groups and the distribution of word groups in each tweet it is possible to find an efficient method for this process. Let us consider for example the five most representative words in each word clusters and use them as basis vectors of a vector space that contains tweets. Occurrence rates of these words are the coefficients of basis vectors. The dimension of this tweet vector space is 5 (the number of the selected top words from word groups) times the number of all word groups. (In reality it is a bit less, since some word groups contain less than five words.) To specify the numerical value of the dimension, it is ranging from several hundreds up to one thousand, so we need to perform dimension reduction to get data that are easy to interpret. We used the t-Distributed Stochastic Neighbour Embedding (t-SNE) method to reduce the number of dimensions to two. The hSBM model isolates clusters in more layers, as it was mentioned earlier. For each layer different word clusters, i.e. basis were obtained. t-SNE was performed for each layer.

Fig. 4–6. show the tweets in the reduced 2D-space, for three hierarchical layers. The category labels in the plots show the word group that is most represented in the certain tweet. For the first two layers (Fig. 4–5.) clusters are well isolated. The categories are mostly the keywords according to which the data were collected. The model worked quite efficiently to cluster the high number of tweets about several topics. The best clustering can be seen in layer 1 (Fig. 5.). On the other hand, clustering is not really visible in the case of layer 2 (Fig. 6.). It means that no higher hierarchical structure is present in the data.

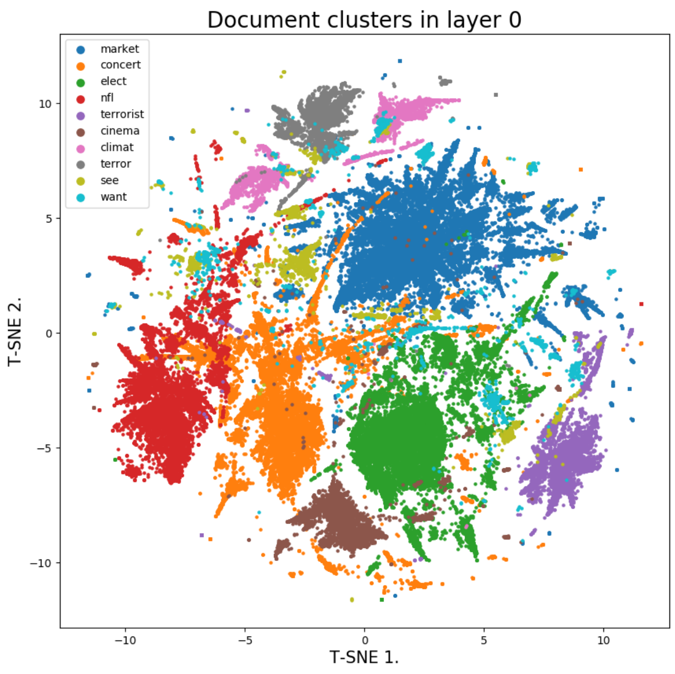

Figure 4: Projection of the vectors in the word-basis representing tweets, for the hSBM layer 0.

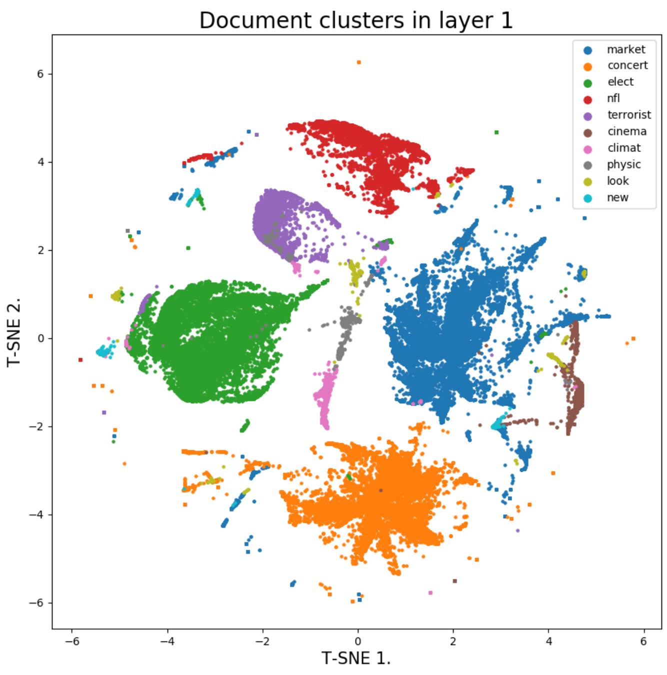

Figure 5: Projection of the vectors in the word-basis representing tweets, for the hSBM layer 1.

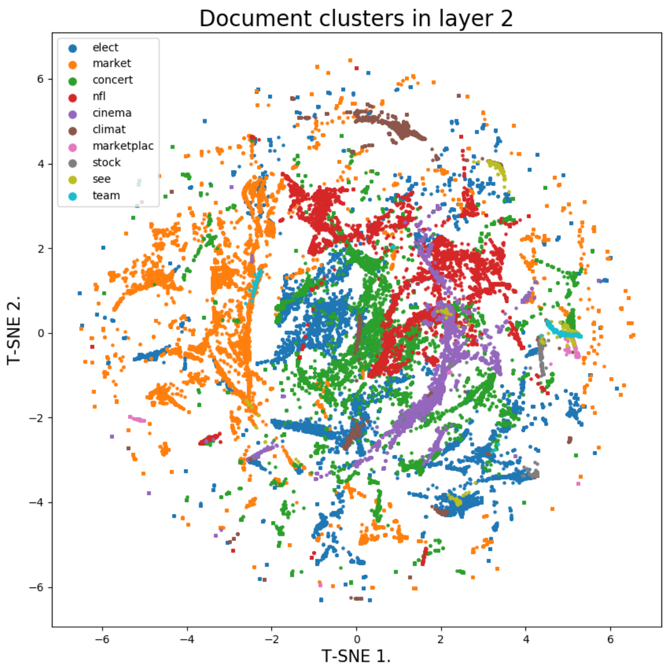

Figure 6: Projection of the vectors in the word-basis representing tweets, for the hSBM layer 2.

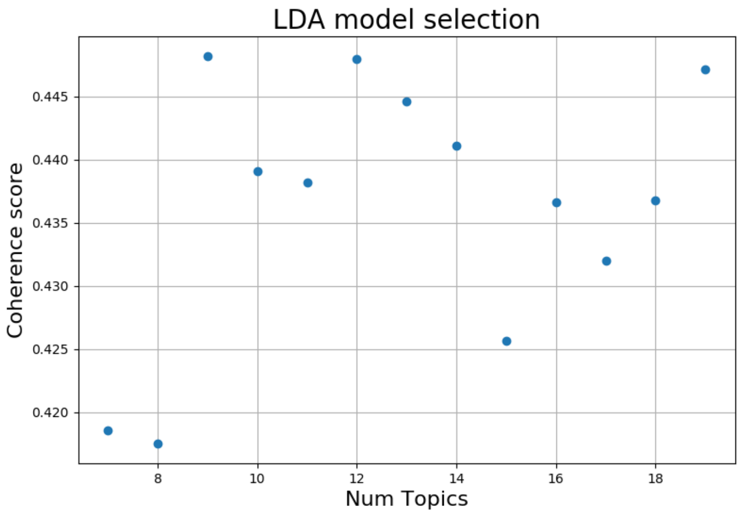

Figure 7: Coherence score of the LDA model as a function of pre-set number of topics.

4.2 LDA

In order to perform LDA calculation for topic modeling the number of topics desired has to be specified. After a calculation is performed its ”goodness” can be evaluated via the *coherence score*. We ran calculations in the range between 7 and 20 topics and found the best score for 12 topics (see Fig. 7.), which is quite close to the number of keywords specified in advance, however not the same. The result of an LDA calculation is the desired number of document clusters (topics) with a varying number of keywords and their occurance rate. For each tweet, the contribution rate of each topic can also be obtained.

Fig. 8. shows the wordclouds of the 12 clusters computed from these weights. The same 7 preset keywords are identified as topics as in the case of hSBM, while the remaining 5 topics are replication of others or less meaningful.

5 Conclusions

Our aim was to discover the most frequently occurring words according to pre-selected topics using two topic modeling algorithms, LDA and hSBM. After tedious data acquisition we could feed the models with appropriate input and perform a computationally massive investigation on around 220 000 twitter messages. In spite of the expectations of poor-descriptive results due to the properties of tweets we gained a very amazing output. The inferred document clusters using the data of the hSBM model visualized on the 2-dimensional t-SNE subspace is consistent with the word cloud created within those clusters: we can find very apprehensive words describing the labels. Results with similarly good quality apply for the word clouds based on LDA, by choosing the best possible number of topics.

The obtained results can be further analyzed and can also serve as input to other research. A possible example could be the more detailed examination of relations between words used in social media.

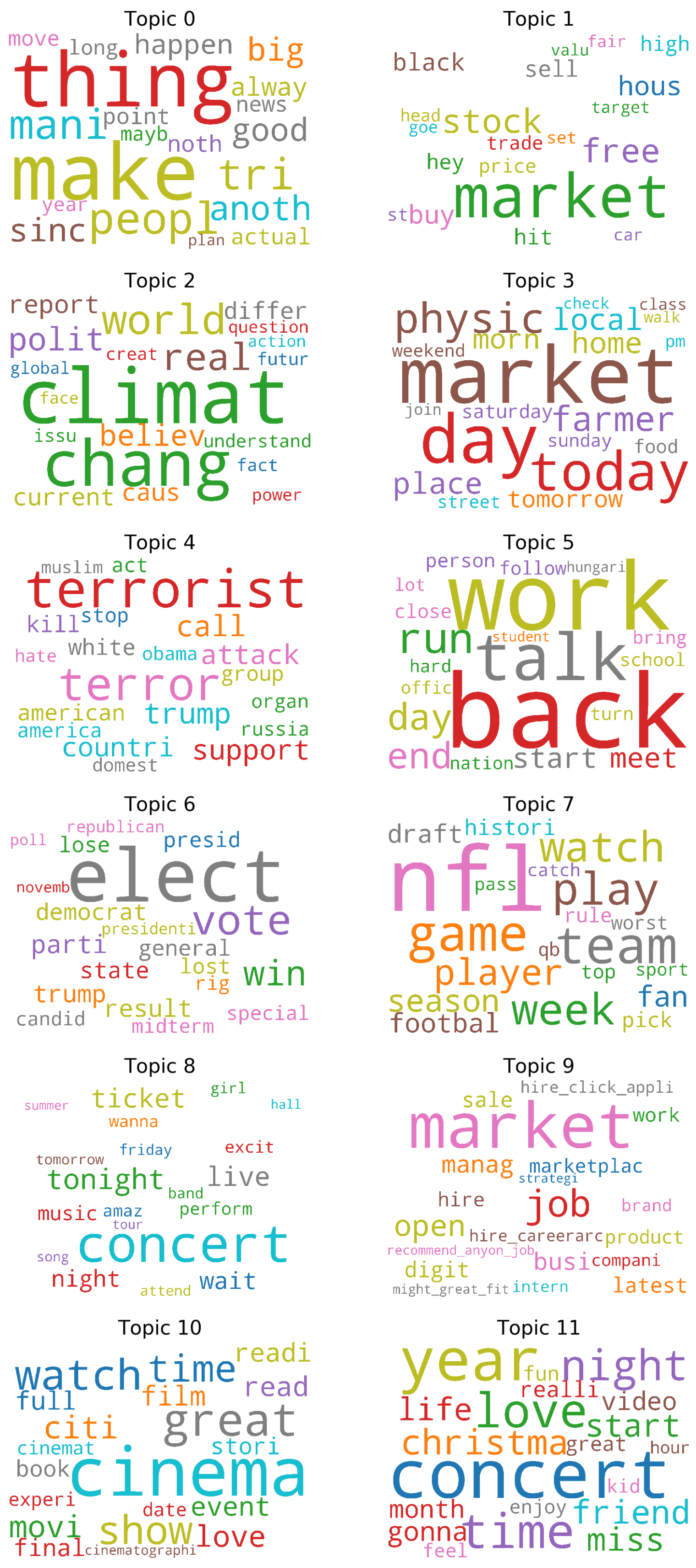

Figure 8: Word cloud visualization of the topics obtained with LDA.

References

[1] Fortunato, S. Community detection in graphs. (2009). doi:10.1016/j.physrep.2009.11.002

[2] Gerlach, M. et. al. A network approach to topic models. Sci. Adv. 4, eaaq1360 (2018).

[3] Using stochastic block models for topic modeling. Available at: https://github.com/martingerlach/hSBM_Topicmodel. (Accessed: 14th December 2018)

[4] Gensim Topic modeling – A Guide to Building Best LDA models. Available at: https://www.machinelearningplus.com/nlp/topic-modeling-gensim-python/. (Accessed: 13th December 2018)

[5] Topic modeling visualization – How to present results of LDA model? Available at: https://www.machinelearningplus.com/nlp/topic-modeling-visualization-how-to-present-results-lda-models/. (Accessed: 14th December 2018)