Visualizing the contributions of a user in the Social Web

AUTHORS:
Maria Schmidt
Maria-Stamatoula Karavolia

UNIVERSITY:
University of Fribourg
COURSE NAME:
Web Monitoring and Analysis
SUPERVISOR:
Aleksandar Drobnjak – Research Assistant

May 25, 2016

Contents

List of Figures

1 Introduction

In the recent decades, the notion of social networks and methods for social network analysis have attracted the interest of researchers and the curiosity of the social behavioral sciences, since there is a wide spread usage of the internet by people all over the world that interact with each other and exchange web content through numerous online communities such as Facebook[1], Twitter[2], Google+[3], LinkedIn[4], Pinterest[5] and many other. Such social networks have rapidly grown in popularity, because they are no longer constrained by the geographical limitations of a conventional social network in which interactions are defined in more conventional ways such as face-to-face meetings, or personal friendships. The main interest is due to the attractiveness of analyzing relationships between entities, behavioral patterns that arise in such networks and the implications that are followed by such analysis. The social network perspective opens new directions in answering questions from social behavioral science by giving definitions to aspects of the political, economical, or social structural environment [26]. Consequently, the answers in such questions have great impact in the society, through understanding how individual behaviors are being expressed [3], how relationships between people are created or ended [2], which characteristics describe people [4] and how social communities are formed [11] and evolve over time [14].

In general, a social network is defined as a network of interactions or relationships, where the nodes typically may represent either users or web content (e.g. posts, news, videos, messages and other), and the edges consist of the relationships or interactions between nodes.

Social networks are typically rich in text, because of a wide variety of methods by which users can contribute text content to the network. For example, typical social networks such as Facebook allow the creation of various text content such as wall posts, comments, and links to blog and web pages. Studying the characteristics of content, for instance in the messages, becomes important for a number of tasks, such as topic detection, personalized message recommendation, friends recommendation, sentiment analysis and others. Topic models [5] are powerful tools to identify latent text patterns in the content. They are applied in a wide range of areas including recent work on Twitter [22].

With today‘s ubiquity and popularity of social network applications, the ability to analyze and understand large networks in an efficient manner becomes critically important. Visualization is becoming an important tool to gain insight on the structure and dynamics of complex social networks. Visualizing social networks is easy to understand and provides detailed information about the actual relations modeled in the data. Visualization of

[1] `http://www.facebook.com`
[2] `http://www.twitter.com`
[3] `http://plus.google.com`
[4] `http://www.linkedin.com`
[5] `https://www.pinterest.com`

social networks has a rich history, particularly within the social sciences, where node-link depictions of social relations have been employed as an analytical tool since at least the 1930s. Linton Freeman documents the history of social network visualization within sociological research, providing examples of the ways in which spatial position, color, size, and shape can all be used to encode information [12]. Freeman mentioned that visualizing social networks is more than simply creating intriguing pictures, it is about generating learning situations: "images of social networks have provided investigators with new insights about network structure and have helped them communicate those insights to others". Moreover, visualization can be very helpful in data analysis, for instance, for finding main topics that appear in larger sets of documents. Extraction of main concepts from documents using techniques such as Latent Dirichlet Allocation, can make the results of visualizations more useful.

The rest of the document is organized as follows. In section 2 we provide the motivation and our research questions. In section 3 we speak about the different aspects we want to investigate and visualize in our work In the next section we describe the visualization of the text corpus and we provide an overview of related work. In Section 4 we describe the Topic models that we used and their relative work. Next, in Section 5 we provide the similarity between short texts and the metric that we used. In Section 6, we introduce the steps that were followed for our prototype implementation, namely Framework, Data Collection and Data visualization. Finally, in Section 7 we conclude our document providing future work directions.

2 Motivation

Social networks have emerged as an important factor in information dissemination, search, marketing, expertise and influence discovery, and potentially an important tool for mobilizing people. More particularly, Twitter and Facebook have been a crucial source of information for a wide spectrum of users and are given access to massive quantities of data for further analysis. Thus, an interesting aspect in social networks is the visualization of the contributions of a user posts and their impact across different platforms. Over time the contributions of an user in social networks is growing and this creates the need to have a simple overview of the data by visualizing it. Data visualization offers a quick way to present the data in a way that can reveal valuable hidden insights. Thus, through the visualization, users can easily understand what are the hot posts, that is those posts are able to attract a greater attention or interest.

2.1 Research Questions

Several research questions arise for visualizing the contributions of user posts in social networks that require additional work. Three interesting research questions are provided below:

1. How can we visualize the contribution of an user in social networks?
2. How can we interrelate the posts published by the same user?
3. How can we find similar posts in different social networks?

Answering these questions is important to understand how much an user contributes to a social network but, also to understand the impacts of his posts. By visualizing his contribution over time we can see for instance, in which month of 2015 he published the most posts in Facebook or Twitter. Moreover, by interrelating his posts we can gain an interesting view about the topics that a user is discussing. Last, by finding the similarity of his posts we can see how many likes has a similar or a same post in Facebook and Twitter.

3 User contribution visualization

As a result of our research about social networks and social network contributions we came up with four different aspects which we wanted to visualize during our work. These aspects are: the amount of contribution over time, a summary of posts, topic models and the impact of one post in different networks. For each of this aspect there are different reasons why it makes sense to visualize it.
If we visualize the contribution over time we can identify interesting points in time. For example if there is a month, where an user wrote more posts than usual we can assume that this month was important for this user and we can start to do more research about what happened at this time. Also an interactive visualization about the contribution over time enables an user to read post from a specific time in the past. An example here could be a student that want to find out about the social network behavior of politicians during a election campaign. He knows the month which are important for the election. A visualization about the contributions of politicians over time with the possibility to click on one month and get all the post of this month, would make it easy for the student to get his desired posts.
With the summary of posts we want to show about what a user is talking in the easiest way possible. A presentation like this makes it possible to get a quick overview about the posts from an user. If for example we find, with the visualization of the amount of contribution, an interesting month a summary about the posts can help to identify why the user posted so much in this month.
Another aspect are the topic models, which give us an overall overview about which topic (themes) an user is posting. Here it can be interesting, that once found the topics, to read the posts which belong to a special topic. For example if we analyze the topics posted by a politician and we find one topic which contains the words "health, insurance, care", we can assume that the politician talks in some of his posts about health-care. We than can get the posts which belong to this topic and read what this politician has to say about health-care.
Our last aspect is maybe not as much important for a normal user than for the writer itself. Showing the different impacts (number of likes, shares, comments) a post has in different social networks enables the writer of the post to see in which social network he has more followers and people who agree with him. Also they can compare the impact of posts in the same social-network to analyze if one post has more likes than the other and find out why this could be. Especially for people who are promoting themselves in social-networks it is important to see how much impact there posts have. They need to know what it is that make people like or share there posts, because just than friends of these people see there posts as well and there message gets spread.

4 Visualize text corpus

With a visualization of the text corpus we can create an overview about the content which makes it easy to see what the corpus is about without actually reading it.

4.1 Literature review

The visualization of text and whole text corpus was already discussed in the papers [8], [9] and [10].

In [8] Fortuna et al. visualized a text corpus using a two-dimensional point map. Documents where represented as points on this map. They also calculate a density for each point which was visualized with a background texture. Around each document point they draw the most common keywords for this document to give the user an idea about the content of a document.
To visualize the document the authors reduced the dimensionality of the document by applying linear subspace methods, like Principal Component Analysis (PCA) and Latent Semantic Indexing (LSI), and multidimensional scaling methods.
Linear subspace methods focuses on finding direction in the original vector space. Projecting the data (text documents) only on the first two directions gives a two-dimensional point representation of the documents.
With the multidimensional scaling methods the document points are positioned into two dimension in a way that they minimize a given energy function.
The basic algorithm of this work was the following:

Algorithm 1: Algorithm of [8] to map documents to two-dimensional points

1. Calculate k dimensional semantic space generated by input corpus of documents
2. Project documents into the semantic space
3. Apply multidimensional scaling using energy function on documents with Euclidean distance in semantic space as similarity measure

After they mapped the documents to two-dimensional points they used other techniques to make the structure of documents more explicit to a user. One technique was to generate a landscape by using the density of the points and visualizing it with a background texture. A second technique described was to assign a set of keywords to each point, which were visualized around the document points within a circle R.

While [8] visualized a text corpus as a point map in two-dimensional space, the authors

of [9] used a graphic approach to visualize a text corpus. In their work they propose different method to project a term-document matrix into a low-dimensional space for visualizing. Before the visualize the corpus they analyse it with a method called *Centering Resonance Analysis* (CRA), which produces a stand-alone, abstract representation of the texts. CRA creates for a text T a graph G(T)=(V(T), E(T)). This graph is build as follows:

Algorithm 2: Algorithm of [8] to map documents to two-dimensional points

1 Extract noun phrases from each sentence. A noun phrase contains nouns and adjectives.
2 Form the set of vertices V(T) with the words of the extracted noun phrases, such that each vertex corresponds to one word.
3 Establish an edge between two vertices, if the corresponding word co-occur in the same noun phrase.

The goals for their visualization was than, that documents with similar content are close to each other, important words appear bigger and display words close to the document they are contained in. They propose different methods and measures to solve their problems.

The authors of [10] also propose a graph structure for visualization. In comparison to [9] they just work with one document, rather than a whole corpus.
After they preprocessed there text (stemming, removing of stopwords), they scan the text with a 2-word gap. For each word that appears for the first time in the text a new node is generated. If a new pair appears in a 2-word gap a new edge is established between the corresponding nodes and the weight is set to 1. The weight of an already existing edge is increased by 1 if the two words appear together again. The more frequent the combination of two words, the higher is the weight of their connection. In a second pass they use a 5-word gap to identify word pairs.
For visualization they use a Force Atlas algorithm, which pushes cluster of nodes away from each other, while aligning the nodes that are connected to the cluster around them. In our case a cluster is a set of nodes which are highly connected.
The size of the nodes is set accordingly to their betweenness centrality, which indicates how often the node appears on the shortest path between any two random nodes in the network. The higher the betweenness centrality of a node, the more influential is the node in the text.
In a next step they detect clusters in the text in order to visualize them with different colors.
As a last step they identify the pathways for meaning circulation by exploring the relations between the clusters and the role of the influential nodes.

4.2 Graph based visualization

For our work we also decided for a graph based visualization, which is based on the idea of [10].
Our goal was to visualize a summary of posts, which were written at a specific time - this could be a month or a time range like the last 14 days.
The algorithm we used to create the graph G=(V,E) was than as follows:

Algorithm 3: Algorithm to create graph for summarizing posts

Input : range of time, number k of most used word to consider
Output: Graph G = (V, E)

```
set C ={all post which publishing date lays in the range of time}
Remove stop words from C
Scan the text and count how often each word appears
Order the list of distinct words by their occurrence in descending order
Take the top k words from this list and form a list L = {k most used words}
foreach word w in L do
    create node X with label w
    set the size of X accordingly to the occurrence of the word in the corpus
end
Scan the corpus again and
foreach word w_1 and w_2 in L do
    if w_1 and w_2 co-occur together and edge E(w_1 ,w_2) does not exist then
        draw edge E(w_1 ,w_2)
        set weight of E to 1
    end
    else
        increase weight of E by 1
    end
end
```

In our case we just used the most common English words as stop words and did not calculate new once.

This procedure gives us a graph, where each node has a specific size and each edge has a specific weight. This values a later used to visualize more frequent words bigger and more frequent connections thicker.

5 Topic models

As our collective knowledge continues to be digitized and stored—in the form of news, blogs, web pages, scientific articles, books, images, sound, video, and social networks—it becomes more difficult to find and discover what we are looking for. We need new computational tools to help organize, search and understand these vast amounts of information. To this end, machine learning provides topic models which are a suite of algorithms for discovering themes (or topics) that spread through a collection of documents.

In this section, we provide an overview of existing literature on topics models and then we describe two types of existing topic models: (i) the probabilistic model: Latent Dirichlet allocation and (ii) the non-probabilistic model: Non-negative Matrix Factorization.

5.1 Literature review

Topic modeling is gaining increasingly attention and is applied in a wide range of areas including on social networks such as Twitter and Facebook. From the view of methodology, topic models are separated into two groups: the non-probabilistic and probabilistic approaches.

Most of probabilistic approaches are based on Latent Dirichlet Allocation (LDA) [6], which is the most popular standard tool in topic modeling. As a result, LDA has been used and extended in a variety of ways, and in particular for social networks and social media, so a great number of research papers that deal with LDA have been proposed.

Ramage et al. [22, 23] extended LDA to a supervised form and studied its application in micro-blogging environment. More particularly, in [23] the Labeled LDA, a novel model of multi-labeled corpora that directly addresses the credit assignment problem is introduced. In [22], a scalable implementation of a partially supervised learning model (Labeled LDA) is proposed for discovering topics in microblogs like Twitter. This model maps the content of the Twitter feed into dimensions such as substance, style, status, and social characteristics of posts. Thus, this approach helps to efficiently characterize selected Twitter users along these learned dimensions and indicates that topic models can provide interpretable summaries of users' tweet posts. Phan et al. [21] studied the problem of modeling short text through LDA. In particular, a general framework, based on LDA, for building classifiers with hidden topics discovered from large-scale data collections that can deal successfully with short and sparse text & Web segments.

Moreover, Chang et al. [7] proposed a novel probabilistic topic model to analyze text corpora and infer descriptions of the entities and of relationships between those entities on Wikipedia. McCallum et al. [17] proposed a probabilistic generative model to simultaneously discover groups among the entities and topics among the corresponding text. Zhang et al. [30] introduced a model to incorporate LDA into a community detection process.

More specifically, in this paper they designed a hierarchical Bayesian network based approach, namely GWNLDA(Generic-Weighted Network-LDA) which is inspired by LDA for discover probabilistic communities from complex networks. Similar work can be found in [16] and [19].

Standard LDA is often less coherent when applied to microblog content like Twitter because tweets are short. To overcome this difficulty, some previous studies proposed to aggregate all the tweets of a user as a single document. In [1], a topical classification of Twitter users and messages is provided. This paper deals with the problem of using topic models in microblogs by proposing schemes based on LDA and one extended model based on the Author-Topic model. It also presents that topic models aims some classification problems by indicating that topic models learned from aggregated messages by the same user obtaining higher accuracy. Also, a different approach on topic modeling of Tweets is provided in [18]. This paper focus on how to improve clustering metrics and topic coherence with existing algorithms. More specifically, it provides two novel schemes that lead to significantly improved LDA topic models on Twitter content without requiring any modification of the underlying LDA machinery. The first one is about pooling tweets by hashtags that yields a great improvement in all metrics for topic coherence across three diverse Twitter datasets, and the second is an automatic hashtag assignment scheme further improves the hashtag pooling results on a subset of metrics.

Non-probabilistic topic models are also very popular. One of the most known representative model is the Non-Negative Matrix Factorization (NMF) [20, 15]. Yan et.al [28], proposed a novel term weight called Ncut-weighted, which measures term's discriminability according to the words cooccurrences, for short text clustering. More particularly, the experiments show that the clustering performance of NMF is greatly improved with terms weighted by the Ncut-weight. Due to the severe sparsity of short texts, in[27], a different approach on the non-negative matrix factorization is introduced. This approach first learns topics from term correlation data using symmetric non-negative matrix factorization, and then infers the topics of documents. The experimental results on three short text data sets show that this method provides substantially better performance than other baseline methods like LDA.

5.2 Latent dirichlet allocation

The idea behind Latent Dirichlet allocation (LDA) [6, 13, 25], which is an unsupervised machine learning technique, is to model documents as arising from multiple topics, where a topic is defined to be a distribution over a fixed vocabulary of terms. Specifically, we assume that K topics are associated with a collection, and that each document exhibits these topics with different proportions. The interaction between the observed documents and hidden topic structure is manifest in the probabilistic generative process associated

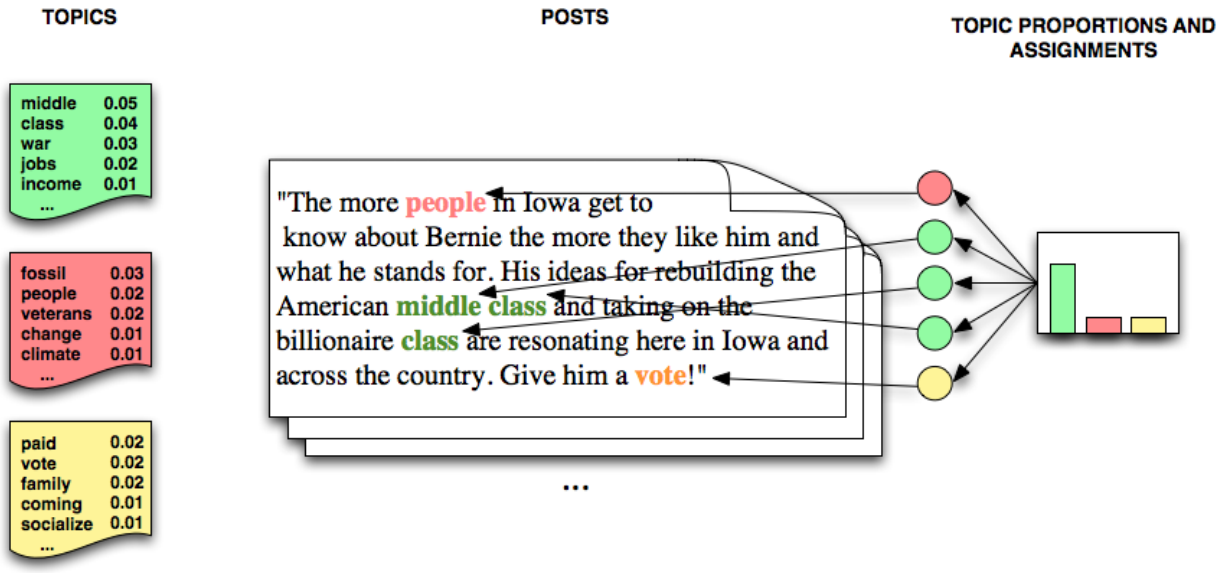

Figure 1: An LDA example.

with LDA. This generative process is as follows:

To generate a document:

1. Randomly choose a distribution over topics.
2. For each word in the document
   (a) Randomly choose a topic from the distribution over topics in step #1.
   (b) Randomly choose a word from the corresponding distribution over the vocabulary

So, this statistical model reflects the intuition that documents exhibit multiple topics. Each document exhibits the topics with different proportion (step #1); each word in each document is drawn from one of the topics (step #2b), where the selected topic is chosen from the per-document distribution over topics (step #2a).

In Figure 1, an LDA example is illustrated. We assume that we have three topics, which are distributions over words, exist for the whole collection (far left). Each document is assumed to be generated as follows. First choose a distribution over the topics (the histogram at right) and then, for each word, choose a topic assignment (the colored coins) and choose the word from the corresponding topic. As we can see, in this example the particular document is assigned to the first topic with higher probability compare to the others.

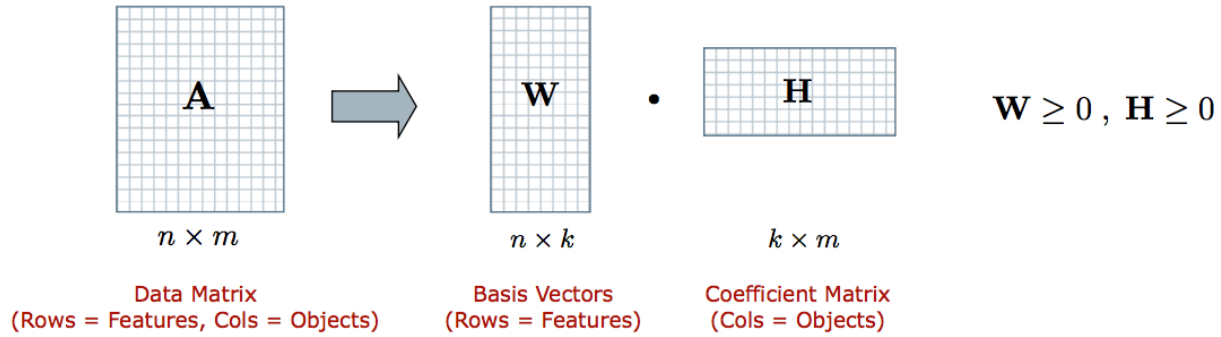

Figure 2: Illustration of approximate non-negative matrix factorization: the matrix A is represented by the two smaller matrices W and H, which, when multiplied, approximately reconstruct A

5.3 Non-negative Matrix Factorization

Non-negative matrix factorization (NMF) is an unsupervised family of algorithms from linear algebra that simultaneously perform dimension reduction and clustering. NMF was first introduced by Paatero and Tapper [20] as positive matrix factorization and subsequently popularized by Lee and Seung [15].

In Figure 2, NMF takes a non-negative matrix $\mathbf{A}$ as an input, and factorizes it into two smaller non-negative matrices $\mathbf{W}$ and $\mathbf{H}$, each having k dimensions. When multiplied together, these factors approximate the original matrix $\mathbf{A}$. The specified parameter k controls the number of topics that will be produced. The rows of the matrix $\mathbf{W}$ provides weights that indicate the strength of association between documents and topics. The columns of the $\mathbf{H}$ that indicate the strength of association between terms and topics. By ordering the values in a given column and selecting the top-ranked terms, we can produce a description of the corresponding topic.

6 Similarity between texts

In the field of Information Retrieval the cosine similarity measure is widely use to calculate the similarity between two documents. This measure is based on a bag-of-words representation of the documents. A further explanation of it follows in the next paragraph.
Due to the fact that for our application we want to find texts which are nearly identical the cosine similarity performs quite well. But one has to admit that in general when comparing short texts cosine similarity is working poorly as they are often just a few terms in common. Because of this, after explaining the cosine similarity, we will review some other approaches to calculate the similarity between short text, which can be used in further work, when one wants to find similar post among different people, to get information about what people are talking in a social network.

6.1 Cosine Similarity

For creating the user time-line, we need to find similar posts that a user share both in Twitter and Facebook. For this purpose, we calculate the cosine similarity metric. The cosine similarity is based on the bag-of-words approach. For a corpus we have a fixed vocabulary V with words w_1 to w_n. A document in the corpus is than represented as vector in an n-dimensional space where the i-th entry is a not 0 if the documents contains the i-th words of the vocabulary.
The similarity between two documents is than measure over the angle between the vectors of the documents. Meaning as smaller the angle between the vectors as more similar the corresponding documents are. As a smaller angle θ means directly a larger $cos(\theta)$ one can calculate the similarity between two document vectors t_a and t_b as follows:

$$\cos(\mathbf{t_a}, \mathbf{t_b}) = \frac{\mathbf{t_a} \mathbf{t_b}}{\|\mathbf{t_a}\| \|\mathbf{b}\|} \tag{1}$$

This is called the cosine similarity.
In our worked we weighted the entries of the documents vector with the tf-idf weighting scheme , which measures how important a word is to a document in a collection. The tf-idf weighting scheme assigns to a term t a weight in document d like this:

$$tf - idf_{t,d} = tf_{t,d} * idf_t \tag{2}$$

where $tf_{t,d}$is the frequency of term t in document d given by

$$tf_{t,d} = \frac{number \quad of \quad times \quad t \quad appears \quad in \quad d}{total \quad number \quad of \quad words \quad in \quad d} \tag{3}$$

and idf_t is the inverse document frequency of term t given by

$$idf_t = \frac{Total \quad number \quad of \quad documents}{Number \quad of \quad documents \quad with \quad term \quad t \quad in \quad it} \tag{4}$$

6.2 Literature review

On problem about using the cosine similarity between short texts is, that normally these text have very few or no terms in common even tough they are similar in a semantic meaning. This problem is addressed in [24] and [29].

The authors of [24] are introducing a new method for measuring the similarity between short texts by expanding their terms.
To measure the similarity between two documents d_1 and d_2 the authors propose the following approach:
First they do a web search with each document as a query to find a number of documents that contain terms of d_1 and d_2. For each document they create than a context vector which contains words that tend to occur with the terms from the original text. The similarity between d_1 and d_2 is than calculated as the cosine similarity between the context vectors.
It is also shown that there algorithm produces a valid kernel function which can be used in any kernel-based machine learning algorithm. In an evaluation the authors compare they method to the cosine similarity and show that they method performs well on their queries.

The work of [29] is based on the previous described work. The authors adapt the idea of Sahami & Heilmann and propose another weighting scheme to measure the importance of the words in the expanded vector. While in [24] the TFIDF measure was used, the authors of this work are using a keyword extraction approach. Their idea is that words which appear in the title of a document or at the beginning are typically more important to the topic of the document than other words. A keyword extraction system is used to capture these facts.
In another section the authors also propose to use machine learning to improve the similarity measure. They are taking the output of existing similarity measure as features and combine them to increase the overall coverage.

7 Prototype implementation

In this section we provide a detailed description of our framework and the technologies that used in the front and back ends, the data collection and last the different aspects of data visualization.

7.1 Framework

Our web application is based on Django[6], which is a free and open-source web framework, written in Python, and follows the model–view–controller (MVC) architectural pattern. We used HTML, CSS and JavaScript in the front-end and Python, JavaScript in the back-end. First, the application is using the retrieved data for politicians and athletes from JSON files for all the tasks that will be performed. Then, the application analyzes the data further for visualizing different aspects of user contributions. For producing the dynamic, interactive data visualizations we use the JavaScript library D3[7]. The application provides an overview of that data with different visualizations for presenting the user contribution over the time, user's summary of posts, impact of posts and the topics that a user is discussing.

7.2 Data Collection

Our data collection consists of real-world data from Facebook and Twitter and focus on 28 public persons such as, politicians and athletes because they tend to post the same content on Twitter and Facebook more than a normal user. The attributes of the data along with their definitions are displayed in Table 1. The below data was saved in different files for each user in JSON format.

Table 1: Description of the attributes of data

Attribute	Description
ID	The id of the post
date	The date of the post
text	The text of the post
likes	The number of likes of a post

[6] `https://www.djangoproject.com/`
[7] `https://d3js.org/`

7.2.1 Retrieve data from Facebook

The data from Facebook was obtained by using the Facebook Graph API with the Facebook JavaScript SDK. First we got all the post for one user by using his/her username and saved them in a database. After obtaining all the posts we called the API again with the post-id of each post to get a summary of all the likes,which we also saved in a database.

7.2.2 Retrieve data from Twitter

The data was obtained by quering the timeline API of Twitter with the username of each person related to politicians and athletes. For this procedure we used the Tweepy[8], which is a Python library for accessing the Twitter API. We were able to collect a fixed number of tweets because Twitter only allows access to a users most recent 3240 tweets.

7.3 Back-end

Before we could visualize the data, we had to analyze it and convert it to a format which could be put in a visualization. We used Python as a programming language to process the data. For the calculation of the topic models or similarities of posts we used a library called scikit-learn, a free software machine learning library.

7.4 Data visualization

The data visualization was done with JavaScript D3, a JavaScript library for producing dynamic, interactive data visualizations for the web. Embedded in an HTML webpage, one can create SVG objects, customize them, add data bindings and transitions with the D3 library. A big advantage of JavaScript D3 is, that there exist many examples using the library, which makes it easier to learn it and produce nice visualizations.
In the following paragraphs we want to illustrate which layouts and diagrams of the library we used to visualize our data.

7.4.1 Contribution over time

We decided to show the contribution over time in a monthly intervall. Therefore we calculated how many posts a user wrote in each month in each social network. After this we used a matrix diagram to visualize the contributions. Each row in the matrix corresponds to a year and social network and each column corresponds to a month. Consequently the value of a cell displays the contribution of a user in a month of a specific year in one

[8] `http://www.tweepy.org/`

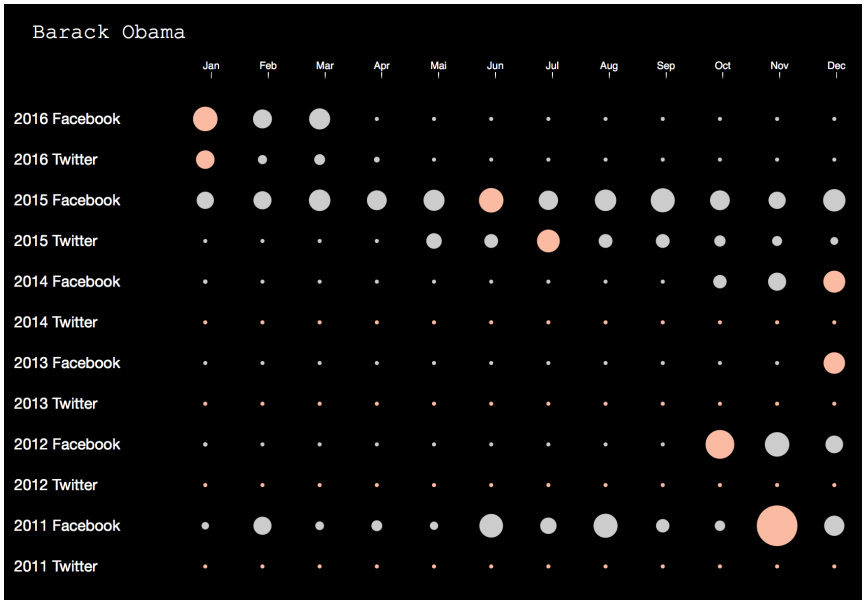

Figure 3: Example contribution over time for Barack Obama

social network. Instead of writing the real value in the cell, we used circles to visualize the amount of contribution. The number of posts where mapped to a range of 0 to 20, which than where used as the radius of the circles. Hence the size of a circle was the greater the more posts there were published in this time.
In figure **??** one can see an example of this approach. In this diagram we showed the contribution over time for Barack Obama. As already mentioned in XXX a visualizing like this one can help to identify month where important events happened. If we would analyse the graphic we probably would say that the November in 2011 was an important month for Barack Obama. as he posted there way more than in the other months. If an user of our prototype is interessted in the posts of Novemeber 2011, he can click on the corresponding circle and the program shows all posts of this month at the side. Figure XXX shows how this looks like.

7.4.2 Summary of posts

For visualizing a summary of the posts we used the approach shown in 4.2. To recap this approach gives us a graph , where each distinct word of our corpus is reprensted as a node and each co-occuring words are connected by an edge. Each node has a occurence-count based on how often the [X] word is used and each edge has a weight based on how often the connected words co-occure together.
Before visualizing the posts, an user can choose a time range, this can be a month in a year or the last X days. The algorithm than just visualizes the summary of all the posts, which where published in this time range.

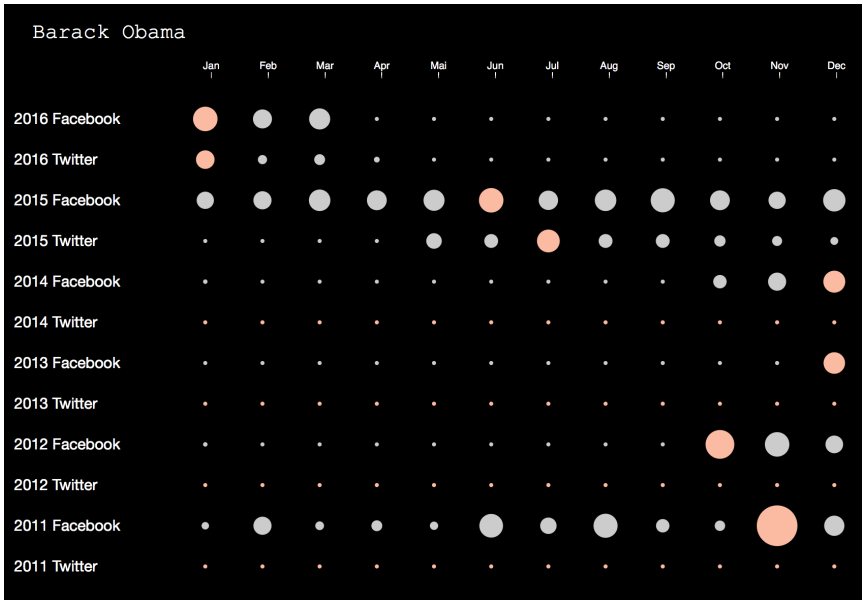

Figure 4: Example of the summary of posts of Barack Obama of January 2016

After getting the posts and calculating the graph, the structure is drawn. We used the Force Layout from the JavaScript D3 library, because this one takes automaticly care, that strongly connected components are close to each other, while components which are not strongly connected are further away from each other. this results in a better overview of the data.
The size of an label of a node was adjusted to the occurence-count of this node. This means that the more often a word occures in the text, the bigger the size of the label of the corresponding node. For the edges we did something similar. As more often two words appear together, as thicker we draw the connection between the corresponding nodes.
In figure 4 one can see an example of our approach. We visualizing the summary of the posts of Barack Obama of January 2016 by considering the 25 most used words. As we can see the more connected components around "Barack", "President" and "Obama" are closer to each other, than the other nodes, which make it easier to read the structure. For further information an user of our prototype can go over one edge with his mouse which opens a textbox with all the posts in which the two corresponding word co-occured together. An example of this is shown in figure XXX

7.4.3 Topic models

For calculating the topic models we used the "LatentDirichletAllocation" class from the scikit-learn library. When initializing an object of this class, one has to decide how many topics should be created. A method called "fit" calculates these topic models than and returns for each topic model a ranked list of words, which are used to describe the topic.

We also used a method the method "transform" which returns document-topic matrix, where a cell $c_{i,j}$ indicates how probable the document i belongs to the topic j. With this matrix we assigned to each document the most probable topic, which gave us an overview of how many documents belong to a topic.
An example for a part of the output of the "fit" method can be seen in table 7.4.3. Here we calculated 5 topic models for the posts of Barack Obama.
For the visualization of the models we wanted to show the top 5 words for each topic to the user, as well as how many documents belong to each topic. While the top 5 words give us an description of the models, the visualization of the amount of documents for each topics gives us an impression about how important each topic is. It is to assume, that a topic which has way more documents than the other topics, is more important to the writer than an other topic.
We choosed a Bubble Chart from the D3 library to realize our ideas. In the chart we assigned each topic a bubble (circle), which size is depending on how many documents belong to this topic. Inside the chart we wrote the top 5 words, to describe the model. Figure **??** shows an example of this approach. Here we created 10 topics for the post of Barack Obama. The number of topics to be calculated can be chosen by the user of our framework. For further information a user can go with his mouse over one topic and see the number of documents which belong to this topic, as well as other words, which describe this topic. If the user clicks on one of the bubbles, the documents (post) are shown at the side. This approach enables an user to find out what a person is talking about, as well as just getting the posts about the topics the user is actually interested in.

	$word1$	$word2$	$word3$	$word4$	$word5$
$Topic1$	*house*	*president*	*white*	*obama*	*barack*
$Topic2$	*president*	*change*	*obama*	*climata*	*people*
$Topic3$	*president*	*obama*	*reform*	*day*	*court*
$Topic4$	2012	*check*	*deal*	*love*	*iran*
$Topic5$	*obama*	*president*	*watch*	*america*	*day*

7.4.4 Impact of posts

To compare the impact a post has in different social networks, we had to first find similar post from one person in the different social networks. We used the following algorithm to accomplish this task:

To speed up these process we just calculated the similarity between posts, which were written the same day +/- 30 days.
We predefine a threshold to accept two similar posts to have similarity at least 60%. This portion lets us a great amount of similar posts and also recognize twitter posts that have urls, hashtags and mentions.

Algorithm 4: Algortihm to find the most similar Twitter post for a Facebook post

```
Input  : Collection X of posts from Facebook and collection Y of post from Twitter
Output: The most similar post in Twitter for each Facebook post
1 foreach post x in X do
2 |  init maximum_similarity = 0 init most_similar_post = None foreach post y in
  |  Y do
3 |  |  calculate cosine similarity between x and y if similarity >
  |  |  maximum_similarity then
4 |  |  |  maximum_similarity = similarity most_similar_post = y
5 |  |  end
6 |  end
7 end
```

The visualization was than done with a vertical Bar Chart. We created a time-line which showed the date on which two similar post were written and one bar for the number of likes for the post in its social network. With this chart an user of our framework gets an information about the number of likes of each post which were published on Facebook and Twitter and can compare these numbers. Figure **??** shows an example of this visualization. In this figure we show the impact of post written by Lewis Hamilton. To find out the content of the post an user can just go over one bar with the mouse and the content is displayed.

8 Conclusion

In this document, we analyzed posts from Facebook and Twitter for politicians and athletes by visualizing them for different aspects, namely (i) contribution over time (ii) summary of posts (iii) identify topics from posts and (iv) impact of posts. So, visualization helps to better analyze the data and reveal some hidden insights of them.

Moreover, we provided detailed description about the topics models, such as Latent Dirichlet Allocation and Non-negative Matrix Factorization that used for extracting topics from user's posts. This approach aims to have an interesting view about the themes (or topics) a user is discussing in social network. According to find the similarity of a user's posts, the cosine similarity metric is used. In this way, we had a view about how popular is a similar or same post in a social network.

As a future work, it would be interesting to find similar posts among users on a socials network. This will aim to observe if the users are interested in talking about similar topics and then it is quite possible that they are interested in similar things. Another interesting direction is to take into account more values for the impact of posts such as comments, shares, retweets etc. By looking, for instance the comments of a user, we can gain an interesting view about the sentiment of the post i.e if it positive or negative. Lastly, it would also be interesting to investigate other approaches to visualize posts of a user as well as use the different visualizations together.

References

[1]

[2] L. M. Aiello, A. Barrat, R. Schifanella, C. Cattuto, B. Markines, and F. Menczer. Friendship prediction and homophily in social media. *ACM Transactions on the Web*, 6(2):9:1–9:33, 2012.

[3] F. Benevenuto, T. Rodrigues, M. Cha, and V. Almeida. Characterizing user behavior in online social networks. In *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement Conference*, IMC '09, pages 49–62, Chicago, Illinois, USA, 2009.

[4] S. Bhagat, G. Cormode, and S. Muthukrishnan. Node classification in social networks. In C. C. Aggarwal, editor, *Social Network Data Analytics*, pages 115–148. Springer US, 2011.

[5] D. Blei and J. Lafferty. Topic models. *Text Mining: Theory and Applications*, 2009.

[6] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, Mar. 2003.

[7] J. Chang, J. L. Boyd-Graber, and D. M. Blei. Connections between the lines: augmenting social networks with text. In J. F. E. IV, F. Fogelman-Soulié, P. A. Flach, and M. Zaki, editors, *KDD*, pages 169–178. ACM, 2009.

[8] G. M. Fortuna B, Mladenic D. visualization of text document corpus.

[9] G. M. Fortuna B, Mladenic D. Wordspace - visual summary of text corpora.

[10] G. M. Fortuna B, Mladenic D. Wordspace - visual summary of text corpora.

[11] S. Fortunato. Community detection in graphs. *Physics Reports*, 486(3,Äì5):75 – 174, 2010.

[12] L. C. Freeman. Visualizing social networks. *Journal of Social Structure*, (1), 2000.

[13] G. Heinrich. Parameter estimation for text analysis. *Web: http://www. arbylon. net/publications/text-est. pdf*, 2005.

[14] R. Kumar, J. Novak, and A. Tomkins. Structure and evolution of online social networks. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, pages 611–617, New York, NY, USA, 2006. ACM.

[15] D. D. Lee and H. S. Seung. Learning the parts of objects by nonnegative matrix factorization. *Nature*, 401:788–791, 1999.

[16] Y. Liu, A. Niculescu-Mizil, and W. Gryc. Topic-link lda: Joint models of topic and author community. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 665–672, New York, NY, USA, 2009. ACM.

[17] A. McCallum, X. Wang, and N. Mohanty. Joint group and topic discovery from relations and text. In *Proceedings of the 2006 Conference on Statistical Network Analysis*, ICML'06, pages 28–44, Berlin, Heidelberg, 2007. Springer-Verlag.

[18] R. Mehrotra, S. Sanner, W. Buntine, and L. Xie. Improving lda topic models for microblogs via tweet pooling and automatic labeling. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, pages 889–892. ACM, 2013.

[19] R. M. Nallapati, A. Ahmed, E. P. Xing, and W. W. Cohen. Joint latent topic models for text and citations. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '08, pages 542–550, New York, NY, USA, 2008. ACM.

[20] P. Paatero and U. Tapper. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5(2):111–126, 1994.

[21] X.-H. Phan, L.-M. Nguyen, and S. Horiguchi. Learning to classify short and sparse text & web with hidden topics from large-scale data collections. In *Proceedings of the 17th International Conference on World Wide Web*, WWW '08, pages 91–100, New York, NY, USA, 2008. ACM.

[22] D. Ramage, S. Dumais, and D. Liebling. Characterizing microblogs with topic models. In *Proc. ICWSM 2010*. American Association for Artificial Intelligence, May 2010.

[23] D. Ramage, D. Hall, R. Nallapati, and C. D. Manning. Labeled lda: A supervised topic model for credit attribution in multi-labeled corpora. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*, EMNLP '09, pages 248–256, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[24] H. T. Sahami M. A web-based kernel function for measuring of short text snippets.

[25] M. Steyvers and T. Griffiths. *Probabilistic topic models*, volume 427. Handbook of Latent Semantic Analysis, 2007.

[26] S. Wasserman and K. Faust. *Social network analysis: Methods and applications*, volume 8. Cambridge University Press, 1994.

[27] X. Yan, J. Guo, S. Liu, X. Cheng, and Y. Wang. Learning topics in short texts by non-negative matrix factorization on term correlation matrix. In *Proceedings of the SIAM International Conference on Data Mining*, 2013.

[28] X. Yan, J. Guo, S. Liu, X.-q. Cheng, and Y. Wang. Clustering short text using ncut-weighted non-negative matrix factorization. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, CIKM '12, pages 2259–2262, New York, NY, USA, 2012. ACM.

[29] M. C. Yih W. Improving similarity measures for short segments of text.

[30] H. Zhang, C. L. Giles, H. C. Foley, and J. Yen. Probabilistic community discovery using hierarchical latent gaussian mixture model. In *Proceedings of the 22Nd National Conference on Artificial Intelligence - Volume 1*, AAAI'07, pages 663–668. AAAI Press, 2007.