SocialSOM: Topic Detection on Twitter by Organizing Tweets on User Similarity

Bernardo Simões 20-25 páginas aproximadamente

Technical University of Lisbon - Taguspark Campus,
Av. Prof. Doutor Aníbal Cavaco Silva — 2744-016 Porto Salvo, Portugal
bernardo.simoes@ist.ul.pt
http://www.ist.utl.pt/en/

Abstract. *70 and at most 150 words, What did I do, in a nutshell?, summarize the paper, should be written last , very short context ,what the objectives of the study were*

Keywords: topic detection, twitter, self-organizing maps, classification, clustering

1 Introduction

With the evolution of social networks websites like Facebook and Twitter, the amount of pertinent content about a specif issue is increasing dramatically, which calls for new ways to make sense and catalog this data. The usage of social networks for branding quality and on-line marketing is specially compelling since 19% of all tweets [15] and 32% [25] of blog posts are about brands or products. In the other hand find topic sensitive information on social networks is extremely complicated due to the fact that documents have very little content, slang vocabulary and orthographically mistakes or abbreviations.

The value of data presented in sites like Facebook or Twitter as proven its value in papers like "Predicting the future with social media" Asur and Huberman [1] where it is possible to predict with high precision the value of a movie box office weeks before it debuts, through real time monitoring of the velocity of reference of Hashtags referencing debuting movies.

The academic and enterprise world is now starting to look at Machine Learning for new ways to achieve revenue and visualize data representing the way the world works. It is not strange to see that the Machine Learning course is the one with more students enrolling this year [1] with more than 760 students enrolled.

With emerging new techniques like Deep Learning Bengio et al. [4] which focuses on abstract representations that can lead to more useful representations, one example of this kind of work is Le et al. [21] "Building High-level Features

[1] http://www.forbes.com/sites/anthonykosner/2013/12/29/why-is-machine-learning-cs-229-the-most-popular-course-at-stanford/

Using Large Scale Unsupervised Learning" where a 9-layered locally connected sparse autoencoder with pooling and local contrast normalization on a large dataset of images (the model has 1 billion connections, the dataset has 10 million 200x200 pixel images downloaded from the Internet) trained using model parallelism and asynchronous SGD on a cluster with 1,000 machines (16,000 cores) during three days. Which achieved 81.7 percent accuracy in detecting human faces, 76.7 percent accuracy when identifying human body parts and 74.8 percent accuracy when identifying cats.

The application of unsupervised learning stretches to apply itself to multiple areas, such as Knight et al. [18] work on solving an enciphered, 105 page, from 1866 book.The document was named Copiale Cipher and was found in the East Berlin Academy after the Cold War and has been undecipherable ever since. The decifering of the document was possible due to the usage of k-means algorithm from the Scipy cluster package [2] which is a common and simple tool for data scientists.

While most Data Scientist are struggling to find new smarter algorithms to visualize and understand data, Halevy et al. [11] claims that "its ll about the data" and that solutions to problems like speech recognition and automatic photo enhancements for example, can be solved by just feeding more data to the already in use algorithms. This not a new concept, actually Banko and Brill [3] at Microsoft stated a the same principle a couple o years before applied to most core natural language. One example of this principle is Hays and Efros [12] work where he presents a new way to do scene completion where it is possible to remove elements from pictures which disappear in a way that in a lot of cases is not possible to distinguish with a naked eye.

Radinsky and Horvitz [29] took the next step into deep learning with predictive data-mining software by being able to predict with an accuracy of 70 to 90 percent the probability of natural disasters, disease epidemics, social unrest, and violence outbreaks. By using data gathered from Wikipedia, the 150 years of New York Times archives and web LinkedData. Her work awarded her to be in the MIT top 35 innovators under 35 [3] and was the starting point to her own venture SalesPredict [4] where massive amounts of data from inside and outside the hiring company are used, in order to improve new pipeline opportunities. SalesPredict had recently raised $1 million dollars in seed funding.

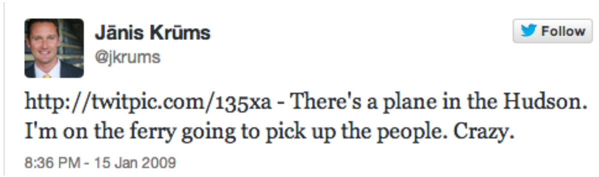
Even though a lot of solution arise in order to automate real time searches, topic categorization and many other data intensive tasks, Twitter still uses humans in order to deliver ads to trending queries, states Edwin Chen's ads quality at Twitter. On his blog post [5] Edwin describes the process of Twitter to deliver real time adds to trending queries, the main problems that arise in the Twitter platform in order to identify rising topic are mainly:

[2] http://docs.scipy.org/doc/scipy/reference/cluster.html

[3] http://www.technologyreview.com/lists/innovators-under-35/2013/

[4] http://www.salespredict.com/

[5] Edwin Chen's Blog, engineer at Twitter: http://blog.echen.me/2013/01/08/improving-twitter-search-with-real-time-human-computation/


Jānis Krūms
@jkrums

http://twitpic.com/135xa - There's a plane in the Hudson. I'm on the ferry going to pick up the people. Crazy.

8:36 PM - 15 Jan 2009

Fig. 1. Tweet by Jānis Krūms, while he is going to pick up some people

- The queries people perform have never before been seen, so it's impossible to know beforehand what they mean.
- Since the spikes in search queries are short-lived, there's only a short window of opportunity to learn what they mean.

This means that when an event happen, people immediately come to Twitter in order to know what is happening in a determined place in real time. Twitter solves this issue by monitoring which queries are currently popular in real time, using a Storm topology [6] and after the queries are identified, they are sent to a Thrift API that dispatches the query to Amazon's Mechanical Turk service where real people will be asked a variety of questions about the query. One example of this tweets that occur in a rather peculiar situation Jānis Krūms which tweets that he was on his way to the Hudson river to pick up people from a plane crash, the tweet is shown in figure .

Social Media Analytics is another raising topic which draws from Social Network Analysis, Machine Learning, Data Mining, Information Retrieval (IR), and Natural Language Processing (NLP). As stated by Melville et al. [25] 32% of the 200 million bloggers world wide blog about opinions on products and brands, 71% of the 625 million active Internet users actually read blogs and more importantly that 78% of respondents put their trust in the opinion of other consumers where only only 57% of consumers trust advertising in traditional media and even worst only 34% of consumers put their trust in such advertising. This kind of data drives companies to Social Media Analytics in a way to know what people are saying on the web about their companies and products. This new worry has brought to life a lot of new startups like Sumal[7] or ThoughtBuzz[8] but also solutions from the old players like IBM [9] and SAS [10]

Its also important to notice that in the last few years Data Science/Analysis has been a trended topic, mostly due to the fact that big dot-com companies have been making lots of money through exploiting user specific information in

[6] http://storm-project.net/
[7] https://sumall.com/
[8] http://www.thoughtbuzz.net/
[9] http://www-01.ibm.com/software/analytics/solutions/customer-analytics/social-media-analytics/
[10] http://www.sas.com/software/customer-intelligence/social-media-analytics.html

order to deliver ads and sell products. No wonder that if you look that in the top 10 ebooks sold by O'Reilly throuout 2013 four are about data science [11].

In this project we will focus on using an unsupervised learning technique based on neural networks named Self-organizing Map [20] in order to detect topics in Twitter posts, by using the Social Network users as base neurons for clustering. After the network is trained it will be possible to categorize tweets in real time. This approach will be better described in subsection 1.1.

First this report will be dedicated to explain some basics concepts like Document CLustering and specifically Self-organizing Maps in section 2. Further in, section 3 will be dedicated to the state of the art solutions related not only to topic detection but also to twitter data analysis and Self-Organizing Maps.In section 4 Architecture of the purposed solution and at section 5 it will be discussed how to evaluate results achieved. Finally we will this report by referencing some possible future work and with a brief conclusion at section 6.

1.1 Objectives

The objective of this project is clear, finding topics on Tweets by analyzing their corpus specific characteristics, like number of characters in a tweet, Hashtag, "was retweeted", etc.. And contextualize the social network evolving the person that did the tweet by retrieving user specific profile information.

After building the dataset, it will be needed to train the Self-organizing map neural network in order to be able to have the network ready for clustering classification of each future tweet that will be added to it. After the SOM training it will be necessary to categorize the clusters in order to know which topic they belong to. When this step ha ended it will be possibly to get new tweets categorized on the moment they enter the network without further delay.

In the end it will be presented a website where a user will be able to login with his twitter account and see all of his tweets being categorization in the moment they are clustered. After all the user twitts are clustered there will a graphical presentation of the user twitter usage where it will be possible to see multiple statiscal information such as the the topics a user is more interested in and his own twitter Self-organizing map network of topics with his friends, the wire-frames of the website can be seen in the attachments section 7 in figure 11.

2 Basic Concepts

In this section we will start by generally describing what Clustering is and how it works at subsection 2.1, then at subsection 3.1 it will be outlined how Self-organizing [20]maps function, which is the Document Clustering algorithm used on this project.

[11] http://shop.oreilly.com/category/deals/bestoforeillydotd.do?code=DEAL&cmp=tw-nabooksvideosinfoauthornote_best_of_2013

2.1 Document Clustering

Document clustering is an optimal division of data into cathegories without prior knowledge of the data that is being organized, based only on the similarity between documents. Due to the fact that no prior knowledge of the date has to be known Document Clustering is labeled as Unsupervised Machine Learning.

Yuan-Chao Liu et Al [23] described that Document Clustering can be used to a variety of Computer Science fields, such as:

- Natural Language Preprocessing.
- Automatic Summarization.
- User preference mining.
- Improve Text classification results.

In regard to document categorization there are two main types of Document Clustering, Hard Clustering and Soft Clustering. In Hard Clustering one document can only belong to one cluster, while in Soft Clustering one document can belong to multiple clusters.

In regard to document categorization Springorum et al. [32] performed hard and soft clustering with SOMs [20] while identifying polysemous German Propositions. They used regular SOMs to create multiple hard clusters and used Centroid-Based or Preposition-based softening to create Soft Clusters from the Hard Clusters.

The general mathematical description of Document Clustering can be seen in 2 In the first step a data set must be provided in order to cluster the documents.

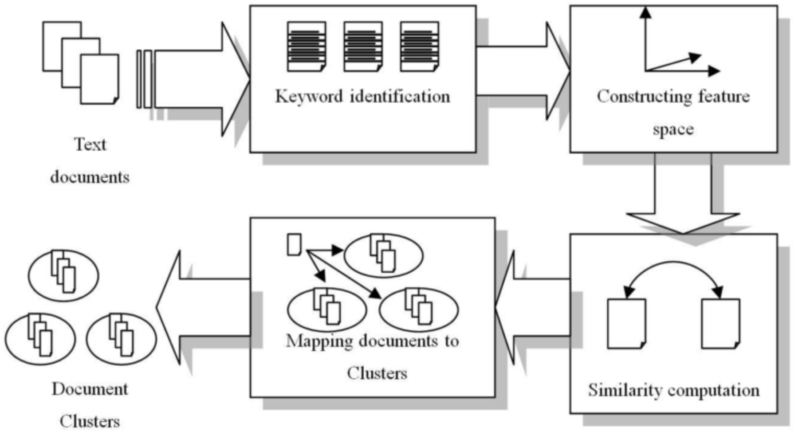


Fig. 2. Text Clustering Main Framework from Dozono [8]

The second step "Keyword identification" is where non relevant words are removed from the documents. Kang [17] proves that keyword removal improves clustering. Another way to extract features is to differentiate text features by analizing the document corpora. For example if the dataset is composed from HTML or XML documents it is possible to identify more relevante features due to the characteristics of the markup. In 3.2 it will be described twitts characteristics as a document and in 4 how feature extraction will be implemented on this project. "Constructing the Feature Space" is characterized by converting the keywords of each document into vectors, the most common algorithm used for this task is SVM (Support Vector Machines). In SVM each vector dimension means one detected key word and each document is represented by the vector of keywords in the feature space. This process and keyword removal is described in Figure 3. Due to the way documents are represented in SVM it is normal that vectors become very large and full of zeros (keywords not present) and it is needed to use sparse vectors to represent the documents in a more efficient way.

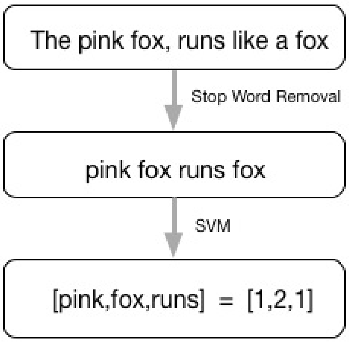


Fig. 3. Stop word removal and transformation to Vector Space Model

Dimensional reduction is done after the construction of the vector space model, in order to reduce the size of the vector space. There are two main ways to do this PCA (Principal Component Analysis) and LSI (Latent Semantic Indexing). PCA calculates the k eigenvectors of the co-variance of the document matrix, which reduces the size of the matrix to k. LSI (Latent Semantic Indexing) works just like PCA but the eigenvectors are calculated directly from the document matrix.

There are two main strategies for Document Clustering, Complete strategie where the data set does not change and Incremental where initial number of document can increase by adding new documents. After a new document is added it can be merged into a existing cluster, or can be separated as a new category. While adding new documents it might be needed to re run the clustering algorithm.

After the algorithm converges, cluster similarity can be calculated in multiple ways:

- Shortest Distance Method: Shortest distance between two members of different clusters.
- Longest Distance Method: Longest distance between two members of different clusters.
- Group Average Method: The average distance between all elements of both clusters.
- Centric Method: The distance between the center of two clusters.

There many clustering algorithms, here we will focus on the three most popular.K-means works by randomly selecting k documents as the cluster centroids, then assign each document to the nearest centroid, and finally recalculate the the centroid with new added documents. The algorithm should be executed until convergence which reflects in the centroids stop changing. K-means has the advantage that the number of centroids must be selected before starting the algorithm. AHC or Agglomerative Hierarchical Clustering is hierarquical clustering algorithm where clusters have sub-clusters which have subclusters. Like K-means it is also a simple algorithm that starts by calculating the similarity matrix, then each document is seen as a cluster and finally merge the nearest two clusters into one and update the similarity matrix. The algorithm ends when there is only one cluster or due to clustering entropy.An AHC classic example is species taxonomy where species have subspecies which have subspecies, etc. Lastly there is Self-organizing Maps introduces by [20] which will be used in the thesis and will be detailedly described in the next subsection 3.1.

2.2 The Self-organizing Map

The Self-organinzing map, or for short SOM is a kind of recurrent artificial neural network that as the desired property of topology preservation which mimics the way cortex of high developed animals brains work.

As [2] describes the basic idea behind SOM is to map the data patterns into an n-dimensional grid of neurons or units. That grid is also know as the output space, as opposed to the initial space also called input space, where the input patterns are. Both spaces can be seen in picture 5.

SOMs work similar to the way that is thought that the human brain works by having a set of neurons that through learning experience specialize in the identification of certain types of patterns. These so called neurons are responsible for categorizing input patterns for which they are responsible. Nearby neurons will be organized by similarity which will cause that similar patterns will activate in similar areas of the SOM. With a topology preserving mapping, SOM organizes the information spatially where similar concepts are mapped to adjacent areas. The topology is preserved in a sense that as far as possible neighborhoods are preserved through the mapping process. Neurons are displayed in an n dimensional grid, generally rectangular, but other dimensions are possible like hexagonal or octagonal. The grid of neurons, also called output space can be divided in neighborhoods where neurons responsible for the same kind of input reside. In SOM neurons will have the same amount of coefficients as the input patterns and can be represented as vectors through the SVM model described earlier in section 2.1.

Before describing the algorithm it is important to define three key aspects of the SOM, the learning rate and quantization error. The learning rate is a function that will be decreased in order to converge to zero, it will be applied to winning neurons and their neighbors in order for them to move toward the corresponding input pattern. Quantization Error is the distance between a given input pattern and the associated winning neuron, it describes how well neurons represent the input pattern. The radius of the neighborhood around the winner neuron is particularly relevant to the totpology of the SOM, deeply afecting the unfolding of the output space as stated by [2] The learning phase is characterized by the training algorithm, which works the following way:

- Neurons can be initialized randomly or it is possible to select initialization neurons.
- Given an input pattern, calculate the distance between the input pattern and every neuron on the network.
- The winning neuron will be the closest neuron to the input pattern.
- The neuron will move towards the data pattern at a given learning rate, in order to improve his representation as can be seen in figure 4.
- Neighbor neurons will also improve their representation in order to keep the network progressively organized as can be seen in figure 5.

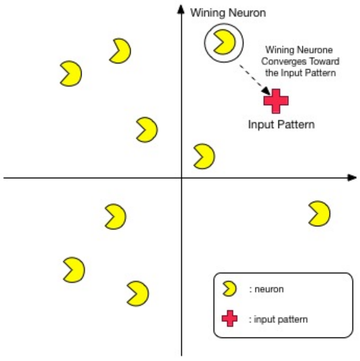


Fig. 4. Winning neuron converging at learning rate

After the algorithm converges, the prediction phase starts. On the prediction phase new input patterns can be quickly assigned to the SOM, without need to

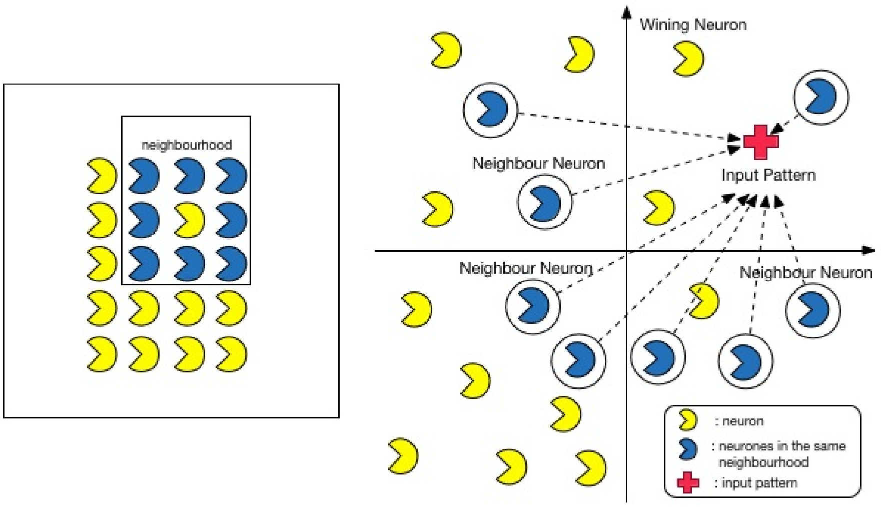


Fig. 5. On the left the output space neighbor, on the right the neighbors of the winning neuron converging

apply the learning rate to the winning neuron and his neighbors (the learning rate converged to zero), it very easy and fast to classify new data now. It is possible that during training the SOM gets stuck while unfolding, this kind of behavior might happen if the input patterns are very complex.

In order to visually interpretate the result of the SOM U-matrices may be used as stated by [2]. The U-matrix is a representation of the SOM in which distances, in the input space between neurons is represented using a gray scale.

The advantages of using SOM is data noisy immunity, easy to visualize the data and parallel processing.

3 Related Work

This section provides insight of work done in multiple research areas that are related to the project. In subsection 3.1 will be described multiple work done using Self-organizing maps. Subsection 3.2 is dedicated to work done on topic detection on the social network Twitter [12]

3.1 Self-organizing Maps

Self-organizing maps are used in a wide are of applications, from authentications systems [8], network intrusion detection [26] and speech recognition and analysis [19].

[12] http://www.twitter.com

Detecting Hidden Patterns on Twitter Usage [7] analyzed hidden patterns created buy the natural usage of twitter by its users. In its study they started by collecting dallta from the twitter API different kinds of topics like "2009 Iran Election" and "iPhone 3.0 OS launch". They made multi level signal extraction not only from information directly present on the tweet, but also by cross referencing with other social website and with the twitter user profile information. The signals retrieved from the social network can be seen in table 1.

Table 1. Twitter Signals

Twitt Corpus	**Twitter Profile**	**External Sources**
Tweet Size	Gender	Other Social Network Accounts
Replies	Number of customizations	Type of website
Re-tweets	Friends to followers ratio	
Hashtags	frequency of posts	
Presence of URIs and Type of linked content	Account Age	
Type of Device	Country	
Tweet Location		

By applying clustering algorithm of SOM, they could find 4 demographical clusters during the Iran 2009 Election. The first cluster was characterized by young web-based Iranians, with twitter accounts not older than 3 months with a high frequency of replies. The second cluster was mainly compound of web users from Iran accounts older that 3 months. The third cluster had Iranian users with mobile clients with large texts clearly trying to raise awareness. The fourth and final cluster represented the users around the world trying to raise awareness about the issue by sharing tweets with URIs. Looking at their analysis about the topic "2009 Iranian Election" it is clear to see that it was possible to describe the type of users represented in the social network and the way they interact with it.

On the iPhone 3.0 OS launch it was possible to find three main clusters. The first cluster was characterized by male users, accounts older than 90 days, coming from countries where the iPhone is marketed, with high adoption of social media clearly representing the target market of the iPhone or its customers. The second cluster had new accounts with higher rate of followers to followees, high frequency of posts per day, presence of URI linking to technology blogs or websites, no country or gender specified meaning that this cluster was clearly composed by news aggregators and technological news websites. Inside the second cluster there was a sub-cluster of Japanese users which represents the high rate of iPhone adoption in Japan. Finally the third cluster was clearly spammer accounts that where eventually deleted after a couple of months, characterized by popular social connections, posting more than 50 tweets a day with external URIs and the accounts where not older than a day or so.

In conclusion it was possible to detect Twitter usage patterns and specifically detect spammers before they where banned from the social network.

Types of SOMs Depending on the kind of data that scientist are trying to analyze and visualize, different approaches can be made the SOM algorithm in order to better adapt to the data at hand.

Weight adaptation SOMs are simple Self-organizing maps in which the weights of the vector space model are not even. For example Bação et al. [2] proposed the adaptation of the algorithm in order to better represent geographical data where more weight is given to the coordinates of the data.

First introduced by Ichiki [14], the Hierarchical SOMs are often used when it is possible to decompose on big problem into smaller problems. One or more SOMs are located at each layer usually operating in on different variables. The hierarchical SOM allows the creation of thematic classifications at lower layers which are then composed into a single one Bação et al. [2] which leads to a more specific kind of classification in the lower layers. Based on the survey made by their work Henriques and Lobo [13] suggests that there are four main types of hierarchical SOMs, Thematic Agglomerative, Agglomerative HSOM based on clusters, static devised HSOM and dynamic devised HSOM. Multi layered SOMs where used by Rauber et al. [30] on automatically detect and organize topics in order to organize bookshelves. They used the vector space model to define all books and started initially with only four cluster, that where consequently subdivided and categorized which in the end created a hierarchical tree of topics. Multi layered SOMs have been used in a wide variety of applications, such as speech recognition Ichiki [14] and learning to control a robot arm and wrist Sayers [31].

The Geo-SOM Bação et al. [2] applies the first law of geography "Everything is related to everything else, but near things are more related than distant things." to the SOM algorithm, where the winning neuron is chosen by in a radius k defined by the geo-coordinates of the data. In this way the Geo-som forces units that are close in the input space to be close in the output space. The representation of the Geo-som can be seen in figure 6.

Self organizing maps have their own limitations mainly drawing from the fact that it has a fixed number of neurons. Qiang et al. [28] cites in his survey about the state of self organizing maps, that newer algorithms namely Growing cell Structures [10] and Growing Neural Gas [9] don't have this drawback.

3.2 Topic Detection and Clustering

There have been many topic detection techniques used throughout the time. Many of them rely on the TF IDF (term frequency – inverse document frequency, based on IDF by Jones [16]) which is not particularly adequate for topic detection on Twitter due to the fact that tweets are very small, composed by typos or slang

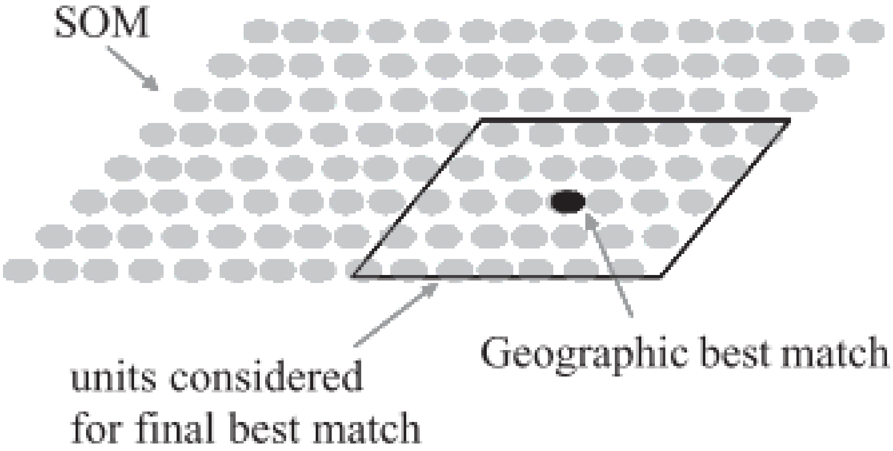


Fig. 6. Geo-SOM structure, from Bação et al. [2]

words and might be written in multiple languages, sometimes at the same time. In this subsection we will take a look at multiple methods of topic detection in general and specifically on the Twitter social network.

Topic and Trending Detection Due to the social rapid social adaptation from people to always be on-line, through the usage of cellphones on the move desktops at work and even TV at home, the increase of user generated content has increased tremendously in latest years. In 2006 35"%" of on-line adults and 57"%" of teenagers created content on the Internet [13] which in "Internet Years" was ages ago. With amount of content increasing, new real-time and scalable algorithms are needed in order to make sense of all this data. Cataldi et al. [6] proposes a new technique for emerging topic detection that permits real-time retrieval of the most emergent topics expressed by a community on Twitter. Their work applies the PageRank [27] algorithm to the users follower / followee relationship in order to find the most influential user on the network and then calculate the more trending topics by relating, social influence, word co-occurrence and time frame. In the end an interface was created where it would be possible to navigate hot topics in a given time frame. It is important to say that topic labeling was not automatic and was implicit by the time frame of an event, if two highly social events would occur in the same time frame with word relations the results could be interpreted as the same, for example if a political candidate would win the elections at the same of an important

[13] Data source: http://www.pewinternet.org/Presentations/2006/UserGenerated-Content.aspx

sport club would win it specific cup, the word win could be trending at the same time for two different topics and due to high temporal dependency they could be interpreted as the same topic. Weng et al. [35] also used the PageRank algorithm in order to find the most influential twitter users on a certain topic, but uses a different approach where they represent each twitter user as a bag of words comprising of all the tweets that they have posted, afterwards it uses Latent Dirichlet Allocation [5] in order to find the topics each user is interested in. In the end it was possible to prove that follower / followee relation on twitter was not just casual, but that people actually follow other people in which they have some resemblance or common interest, this concept is called homophily and will be further explored by this project.

Sudhof [33] presents a model to where for a given user and a certain topic, it can evaluate the user side on a determined manner of case. For example

3.3 Data Mining in Twitter

In this subsection, we will focus on work done on Twitter social network in order to leverage insights on how the public data available from the website can correlated within itself and with outside sources.

Enhancing the Tweet Tweet retrieval and analysis is a double edged problem. On one side the tweet is really small which makes it almost impossible to retrieve any actual sense from it. On the other hand the amount of tweets generated per day is around 140 million [14] wich means that it is very hard to to deep analyses the semantics and content of individual tweets, but if so is done, only the more appropriate signals should be evaluated. Tao et al. [34] evaluated how the multiple signals that could be retrieved directly or indirectly from the tweet corpus could mean that a tweet is relevant for a determined topic. In his work, Tao presents premises that seem intuitively true and proves if they actually are relevant through comparison of multiple precision and recall values. Its results on feature comparison where summarized in table 2, the first row consists of all the made hypothesis categorized by type, and the second row tells if the data used actually influenced in precision and recall results.

Tau also compared result of topic characteristics, concluding that distinction between local and global events as well as temporal persistence proved to not be relevant on relevance prediction.

McCreadie and Macdonald [24] also approached the issue of having very little content on tweets in order to categorize a tweet, and tried to solve it by applying the content of linked URIs into the tweet body in order to improve precision and recall. The best fitting approach was using Field-Based weighting where for each tweet a new document is created which contains two fields; the terms in the tweet and the terms in the linked document. Afterwards a learning to rank algorithm PL2F is used against the dataset from Microblog2011 in order to find the best

[14] https://blog.twitter.com/2011/numbers

Table 2. Tao et al. [34] resumed results

Hypotheses	**Influence of Features**
Syntatical	
Tweets that contain Hashtags are more likely to be relevant than tweets that don't	Not Important
Tweets that contain an URI are more relevant that tweets that don't	Important
Tweets that are replies to other tweets are less relevant	Important
The longer the tweet is the more relevant it is	Not Important
Semantic	
The more the number of entities the more relevant a tweet is	Important
Different types of entities are of can have different amount of interest to a give topic	Important
The greater the diversity of concepts mentions in a tweet the more likely for it to be relevant	Important
The relevance of a tweet is determined buy its polarity	Important
Contextual	
The lower the temporal distance between a query and the creation of a tweet the more relevant the tweet is	Not Important
The more the number of tweets created by a user the more relevant one of his tweets will be	Not Important

weighting that should be applied to the tweet corpus and the URI referenced page. With this trained model they where able to improve precision in an order of 0.9.

Rapidly Changing Trends Due to the real time nature of Twitter, using typical retrieval model that rely on term frequency models like BM25 or language modeling cannot be applied as stated by Lin and Mishne [22]. The study of topic perdurance on the social network proved that it is presented in bursts of queries and mentions of a topic. The typical usage of twitterr for search is not the same of Google, when user are searching in twitter they want to find out what is happening right know meaning that classification techniques based on past events cannot respond this kind problem. As stated by Lin and Mishne [22] this problem has not yet been solved at twitter (or anywhere else at the time of writing this report), and issues a new kind of data analysis approach that was not taken into consideration in the past. This effect of rapidly changing topics and queries based on real time events was named ”Churn”, and can be clearly seen in figure 7.

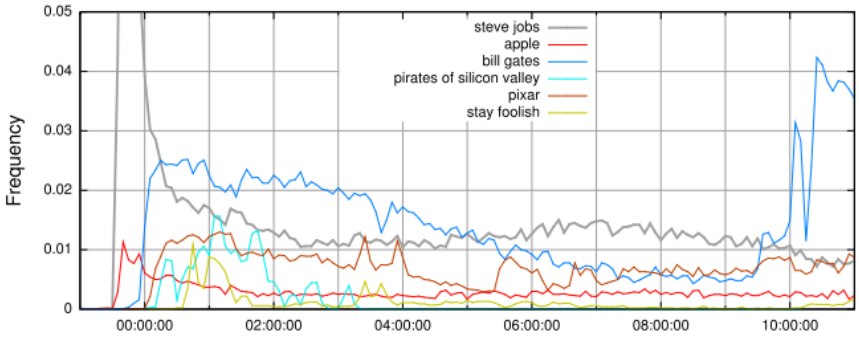


Fig. 7. The Churn effect: Frequencies of queries related to Steve Jobs death over a 12 hour period in 5-minute intervals, normalized to the total number of queries in the interval. At its peak, the query "steve jobs" reaches 0.15 (15% of the query stream); Graph taken from [22]

4 Architecture

This Section is comprised of the multiple layers needed in order to successfully be able to implement the proposed project objectives described in subsection 1.1. In subsection 4.1 will be discussed the data gathering process, the at subsection 4.2 will be detailed the SOM algorithms that will be tried in order to achieve the best results in Topic Detection on Twitter. The description of results evaluation will be described in subsection **??** and finally the Web Site architecture will be described in subsection 4.3.

4.1 Building a Dataset

In order to apply any kind of Clustering or Learn to Rank algorithm, a Data Set is needed. The two most common approaches to Dataset building are finding some Dataset that was used beforehand by someone, use the Twitter API to build your own or any combination of the two. For this project we will focus on building our own dataset for SOM training, but existing datasets might be used in order to rate results. The description of existing datasets will be presented in Subsection 5.1.

Twitter API Building your own Data Set through the Twitter API has become harder with passing years with the introduction of API limits and mandatory authentication. With these new limitations, companies like gnip [15] or [16] with licenses from Twitter are selling access to their archives of tweets.

[15] http://gnip.com/topsy/

[16] http://www.tweetarchivist.com/about/subscriptions

In order to retrieve data from Twitter is crucial to understand how their API functions. The Twitter API right now is divided into two, the REST API and the Streaming API. Both of the API's can be used at the same time, and have different types of limits. In a general way, the streaming API is used for subscriptions, where a an application can subscribe a given hashtag or User activity on the social network, and they are automatically pulled to the subscriber app. The Streaming API has no specific limit being described in the docs as "The public streaming APIs cap the number of messages sent to your client to a small fraction of the total volume of Tweets at any given moment" [17].

The REST API works by requesting resources and getting the results in a restful way. Here the limits are strict, an application cat only get a maximum number of 3200 Tweets per user and 180 calls to the API per 15 minutes, more API limits can be found on the Twitter API Documentation [18].

Crawling Twitter In order to get tweets from the Twitter API, the data will be crawled in a breath-first fashion where the first user can be selected randomly:

- Get all Tweets from the user.
- Get user profile info.
- Get list of followers/followees.
- Select a follower/followee and repeat step one.

The algorithm will stop at a given depth level, also if API limits are exceeded the algorithm will have to stop for 15 minutes and afterwards resumed. Given the API access limits, there will be no need to run the crawler asynchronously since achieving a greater level of performance will only make the algorithm achieve API limits sooner.

Storing Crawled Data While the crawler is getting data from Twitter, it will be storing it in a Redis database [19]. Given the amount of databases available in last few years, Redis was chosen for this project because it met the following criteria:

- Free.
- Simple to install and run.
- Can persist data to disk.
- Its really fast to write, by not granting data integrity (which is not a problem since this project is not dealing with sensitive information)
- Good documentation.
- Non relational, Key/Value store.
- Stores json.
- Client libraries for almost every programing language.

[17] https://dev.twitter.com/docs/faq#6861

[18] https://dev.twitter.com/docs/rate-limiting/1.1/limits

[19] http://redis.io/

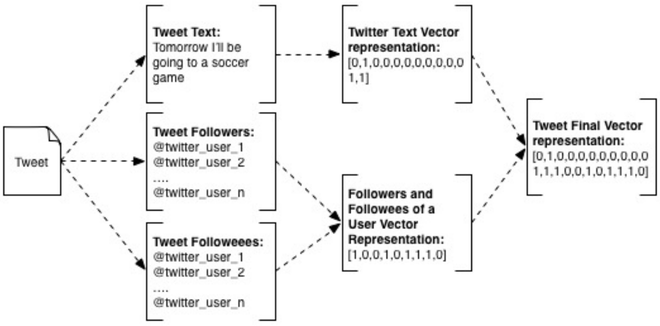


Fig. 8. Vector Space Transformation of a Tweet

– Integrated publish/subscriber.

Given the characteristics of the Redis database, there will be no need to write a schema beforehand. With this in mind, user information and Tweets will be stored directly in json into the Database.

4.2 The SOM Algorithm

The SOM algorithm in this report will have a twofold approach. Primarily it will be tried to transform the tweet social characteristics and words into a vector using the vector space model, this approach will use the default SOM implementation and will be described in detail in Subsection 4.2. The second approach will be inspired by Bação et al. [2] where the SOM algorithm will be altered in order to take in consideration the social network during the its training, this implementation will be described in Subsection 4.2.

Default SOM Approach In order to train the SOM first it will be needed to convert the tweets into the vector space model. There will be two binary vectors, the first one will represent the presence of all the words gathered in all tweets where each value 1 will represent the presence of a word. The second vector will represent the social connections between the user. On a first approach in order to give more relevance to social connections only followers that are followed back will be represented with ones, this approach will give a higher representation to the social interaction between the two users since on the Twitter social network it possible to follow someone without the followed person accept the request. In the end both vectors, the word representation vector and the social connections representation vector will be concatenated. Figure 8 shows the transformation from the tweet into the vector space model.

The SOM initialization will be tried in different ways and measured to see which gives the best results. The initialization characteristics will be the following:

- Random initial number of neurons with random content.
- Random initial number of neurons with random content evenly distributed.
- Each neuron will be the representation of each user that was crawled.
- Each neuron will be comprised of words relevant to a determined topic, and will be responsible to categorize that topic.
- Each neuron will be the representation of a user that is relevant to determined topic, for example the user *Optimus Alive* would be responsible for categorizing the topic *"Music Festivals"* while *Cristiano Ronaldo* neuron representation would be responsible for detecting the topic *Soccer*.

In the initialization ways described above, only the last two will give a SOM ready to classify. The first three options would build uncategorized clusters that would have to be classified *a posteriori*, nevertheless it would be important to look at the results given by them because they could be more interesting then the results from the last two items.

Social SOM In this approach, the use of the vector that described the followers/followees of the user that created the tweet, which is described in Subsection 4.2 will be discarded. Instead there will be a new vector that describes the number of hops between twitter users, a visual representation of this vector can be seen in Figure 9. By modifying the SOM training algorithm, input patterns will only be measured against neurons that have a determined level of social affinity. This social affinity will be defined as x which will define the number of followers/followees relations will be applied. For example if $x = 1$ only neurons that belong to a follower/followee will be selected for comparison to find which is the winning neuron. If $x = 2$ not only the followers/followees of a user will be selected for comparison, but also the followers/followees of the followers/followees. Finally if x would be equal to the number of users used in the dataset, then all neurons would be used for comparison, making this solution equal to a normal SOM.

SOMS Development The development and testing of the SOM described in Sub-subsection 4.2 and Sub-subsection 4.2 will be completely independent of the Web Site described in Subsection 4.3. The SOM training will be made using the datasets described in Subsection 4.1 and connection with components from the Web Site will be made through the Publish/Subscriber Redis interface. This approach will create an highly modular solution where it will be possible to interact with the trained SOM through terminal where it will be possible to visualize and test results.

4.3 Web Site

As described in the report objectives a web site will created to demonstrate the project categorizing tweets per topics. In order to achieve this a Web Application will be implemented that works in the following way:

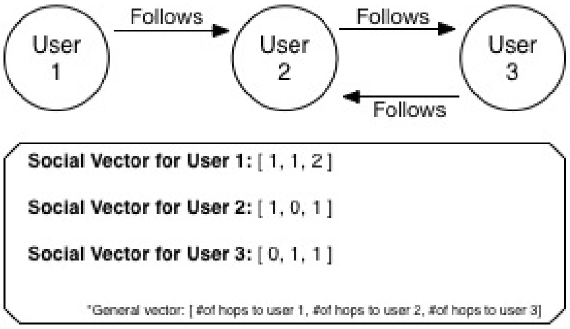

Follows
Follows
Follows
User 1
User 2
User 3
Social Vector for User 1: [1, 1, 2]
Social Vector for User 2: [1, 0, 1]
Social Vector for User 3: [0, 1, 1]
*General vector: [#of hops to user 1, #of hops to user 2, #of hops to user 3]

Fig. 9. Vector describing the number of hops between followers

- A user will be able to login with his twitter id.
- The browser will start displaying the user tweets, with the associated topic.
- After all the user tweets are displayed, some statistics about the user tweeting topics will be displayed.

The wire-frames for the application are provided as attachment, the architecture of the solution can be seen in Figure 10 and will be described in the following sub-subsections.

Client Side Application The client side application will be running on the browser of whomever connects to the website, on the client browser. This application is responsible to authenticate itself against the Twitter API through OAuth and after that will be establishing a web-socket channel that will be receiving the categorized tweets has soon as the server dispatches them. The client application will also have to display an interface to the user so he can interact with the application.

Web Server The web server will be very simple, it will only have to get the tweets from the Twitter API and publish them in a channel through the Redis Publish/Subscriber interface, on the other side the SOM machine will be receiving the Tweets and categorizing them. Also the web server will have to subscribe to the channel where the SOM Machine is publishing the categorized tweets in order to be able to push them to the client through web-sockets.

Redis Database The Redis database will be used as a middle man between the Web Server and the SOM Machine as publish/subscriber system. This is extremely useful because it will separate the login of the web server and web application from the rest of this project. In this way the SOM machine can be running the topic categorization in any programing language while the server

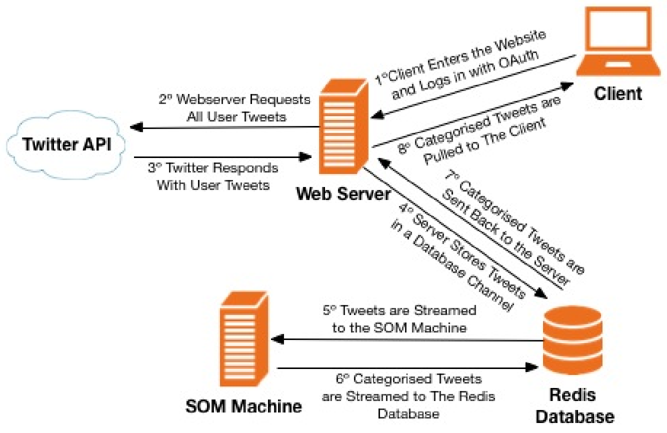

Twitter API
2º Webserver Requests All User Tweets
3º Twitter Responds With User Tweets
Web Server
1ºClient Enters the Website and Logs in with OAuth
Client
8º Categorised Tweets are Pulled to The Client
7º Categorised Tweets are Sent Back to the Server
4º Server Stores Tweets in a Database Channel
5º Tweets are Streamed to the SOM Machine
6º Categorised Tweets are Streamed to The Redis Database
SOM Machine
Redis Database

Fig. 10. Topology of the Solution

can also be working in another language and still be able to contact each other in a simple and efficient way.

SOM Machine The SOM Machine is where the Self-organizing Map trained with the crawled data described in Subsection 4.1 will be functioning. It will be subscribing to new tweets sent from the Web Server and will categorize them using the previously trained SOM. After the tweets are categorized they will be published back so the Web Server can deliver them to the client.

Solution Overview In this Sub-subsection will reside the description of how everything in this solution fits together, based on the diagram in Figure 10 and the steps described will be the same, but they will have more detail. For simplicity sake the OAuth authentication from the client to the twitter API will be omitted since its only objective is to get access to the user tweets.

- **First Step** The client connects to the website through the browser and will login with his twitter account in order for the server be able to download all of his Tweets. When the login process is over it keep an open web-socket with the server in order to receive the categorized tweets.
- **Second and Third Step** The web-server will request all the user tweets through the Twitter API. Twitter will respond with all [20] the tweets from the user that logged in.

[20] The number of tweets per user are limited to the 3200 most recent through the twitter API

- **Forth and Fifth Step** The web server will publish the tweets in the Redis database, while is subscribing to the channel where the categorized tweets will come out. On the other side the SOM Machine is subscribing to the uncategorized tweets in order to categorize them.
- **Sixth and Seventh Step** After the SOM machine categorizes the tweets it will publish them to the Redis database, which is being subscribed by the web server.
- **Eight Step** As soon as categorized tweets start hitting the server they will be immediately sent to the client through web sockets. On the client as soon as the categorized tweets hit the browser, they will be injected into the DOM and the user will start to see the tweets that he has made categorized in topics. Lastly after all the tweets have been sent to the browser, some statistics will appear with the amount of tweets per topic throughout the time.

5 Evaluation Metrics

5.1 Testing for Precision and Recall

5.2 Cluster Quality Testing

- How am I gonna evaluate my work?

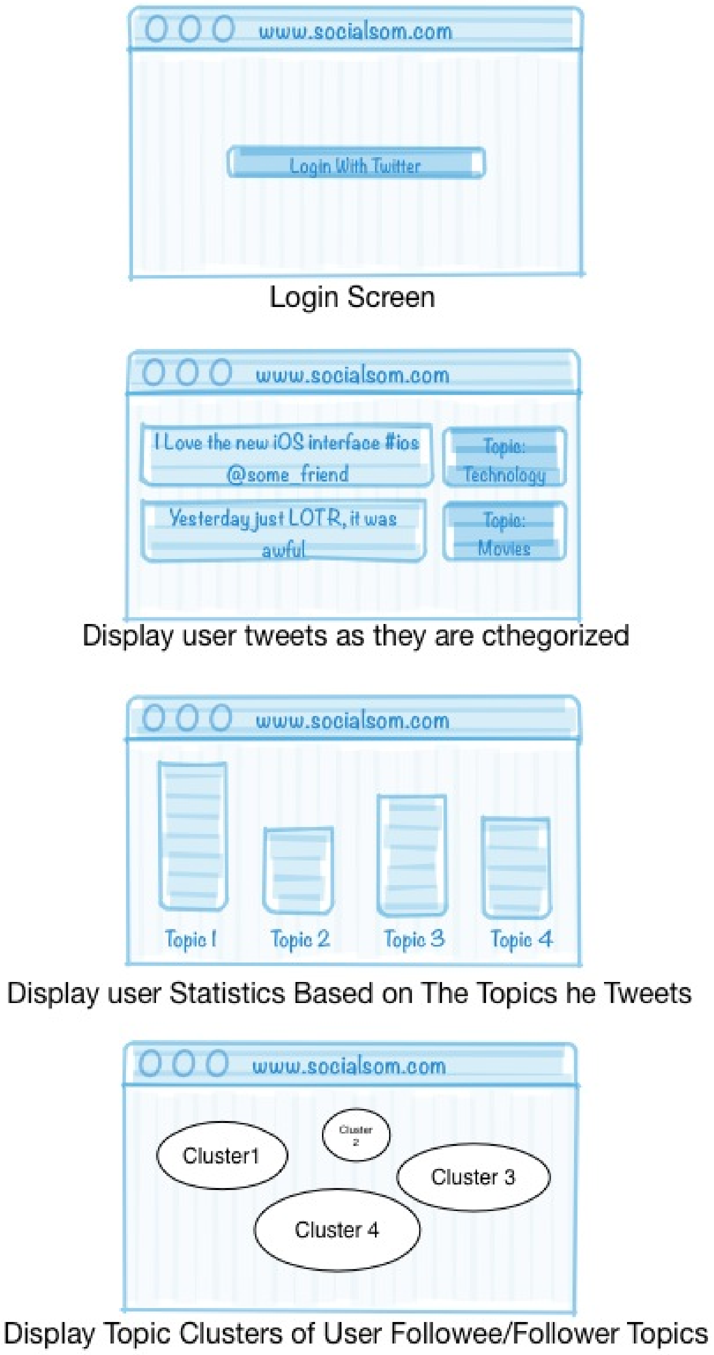
5.3 Evaluation Criteria by Teachers

- Ability to understand the research problem
- Clear and well defined goals
- Description of the different approaches explored
- Ability to relate the state-of-the-art with the research theme Work methodology and adequate planning for the next stage Organization and quality of the written document
- Inclusion and completeness of updated and appropriate references Oral presentation and discussion

6 Conclusions and Future Work

7 Attachments

7.1 Web Site Wire-frames



Login Screen

I Love the new iOS interface #ios @some_friend

Yesterday just LOTR, it was awful

Display user tweets as they are cthegorized

Display user Statistics Based on The Topics he Tweets

Display Topic Clusters of User Followee/Follower Topics

Fig. 11. Website to display user Twitter content

Bibliography

[1] Sitaram Asur and BA Huberman. Predicting the future with social media. *...Agent Technology (WI-IAT), 2010 IEEE ...*, 2010. URL `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5616710`.

[2] Fernando Bação, Victor Lobo, and Marco Painho. The self-organizing map, the Geo-SOM, and relevant variants for geosciences. *Computers & Geosciences*, 31(2):155–163, 2005. ISSN 00983004. doi: 10.1016/j.cageo.2004.06.013. URL `http://linkinghub.elsevier.com/retrieve/pii/S0098300404001918`.

[3] Michele Banko and Eric Brill. Scaling to very very large corpora for natural language disambiguation. *Proceedings of the 39th Annual Meeting on ...*, 2001. URL `http://dl.acm.org/citation.cfm?id=1073017`.

[4] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. (1993):1–34, 2013. URL `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6472238`.

[5] DM Blei, AY Ng, and MI Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003. URL `http://dl.acm.org/citation.cfm?id=944937`.

[6] Mario Cataldi, Luigi Di Caro, and Claudio Schifanella. Emerging topic detection on Twitter based on temporal and social terms evaluation. *Proceedings of the Tenth ...*, pages 1–10, 2010. URL `http://dl.acm.org/citation.cfm?id=1814245.1814249http://dl.acm.org/citation.cfm?id=1814249`.

[7] Marc Cheong and Vincent Lee. A Study on Detecting Patterns in Twitter Intra-topic User and Message Clustering. *2010 20th International Conference on Pattern Recognition*, pages 3125–3128, 2010. doi: 10.1109/ICPR.2010.765. URL `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5597282`.

[8] Hiroshi Dozono. Application of Self Organizing Maps to Multi Modal Adaptive Authentication System Using Behavior Biometrics. 2012. URL `http://www.intechopen.com/books/applications-of-self-organizing-maps/application-of-self-organizing-maps-to-multi-modal-adaptive-authentication-system-using-behavior-bio`.

[9] Bernd Fritzke. Growing cell structures—a self-organizing network for unsupervised and supervised learning. *Neural networks*, 7(9):1441–1460, 1994. URL `http://www.sciencedirect.com/science/article/pii/0893608094900914`.

[10] Bernd Fritzke. A growing neural gas network learns topologies. *Advances in neural information processing systems*, 1995. URL `http://web.cs.swarthmore.edu/~meeden/DevelopmentalRobotics/fritzke95.pdf`.

[11] Alon Halevy, Peter Norvig, and Fernando Pereira. The unreasonable effectiveness of data. *IEEE Intelligent Systems*, 24(2):8–12, March 2009. ISSN

1541-1672. doi: 10.1109/MIS.2009.36. URL `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4804817`.

[12] James Hays and AA Efros. Scene completion using millions of photographs. *ACM Transactions on Graphics (TOG)*, 1(212):1–7, 2007. URL `http://dl.acm.org/citation.cfm?id=1276382`.

[13] Roberto Henriques and Victor Lobo. Spatial Clustering Using Hierarchical SOM. 2012. URL `http://www.intechopen.com/books/applications-of-self-organizing-maps/spatial-clustering-using-hierarchical-som`.

[14] H. Ichiki. Self-organizing multilayer semantic maps. 1991. URL `http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=155203`.

[15] BJ Jansen and Mimi Zhang. Twitter power: Tweets as electronic word of mouth. *Journal of the American ...*, 60(11):2169–2188, 2009. doi: 10.1002/asi. URL `http://onlinelibrary.wiley.com/doi/10.1002/asi.21149/full`.

[16] KS Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 60(5):493–502, 1972. ISSN 0022-0418. doi: 10.1108/00220410410560573. URL `http://www.emeraldinsight.com/10.1108/00220410410560573http://www.emeraldinsight.com/journals.htm?articleid=1649768&show=abstract`.

[17] Seung-Shik Kang. Keyword-based document clustering. *Proceedings of the sixth international workshop on Information retrieval with Asian languages* -, 11:132–137, 2003. doi: 10.3115/1118935.1118952. URL `http://portal.acm.org/citation.cfm?doid=1118935.1118952`.

[18] Kevin Knight, B Megyesi, and C Schaefer. The Copiale Cipher. *ACL HLT 2011*, (June):2–9, 2011. URL `http://www.aclweb.org/anthology/W11-12#page=12`.

[19] T. Kohonen. The 'neural' phonetic typewriter, pages 11 - 22 . 1988.

[20] T Kohonen. The self-organizing map. *Proceedings of the IEEE*, 1990. URL `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=58325`.

[21] QV Le, MA Ranzato, R Monga, and Matthieu Devin. Building high-level features using large scale unsupervised learning. *arXiv preprint arXiv: ...*, 2011. URL `http://arxiv.org/abs/1112.6209`.

[22] Jimmy Lin and Gilad Mishne. A Study of" Churn" in Tweets and Real-Time Search Queries (Extended Version). *arXiv preprint arXiv:1205.6855*, 2012. URL `http://arxiv.org/abs/1205.6855`.

[23] YC Liu, Ming Liu, and XL Wang. Application of Self-Organizing Maps in Text Clustering: A Review. 2012. URL `http://www.intechopen.com/books/applications-of-self-organizing-maps/application-of-self-organizing-maps-in-text-clustering-a-review`.

[24] R McCreadie and Craig Macdonald. Relevance in microblogs: enhancing tweet retrieval using hyperlinked documents. *...Research Areas in Information Retrieval*, 2013. URL `http://dl.acm.org/citation.cfm?id=2491787`.

[25] Prem Melville, V Sindhwani, and R Lawrence. Social media analytics: Channeling the power of the blogosphere for marketing insight. *Proc. of the WIN*, pages 2–6, 2009. URL `http://people.cs.uchicago.edu/~vikass/sma-win09.pdf`.

[26] Kulkarni P. Nikam D.M. Pachghare, V.K. Intrusion Detection System using Self Organizing Maps. 2009.

[27] The Pagerank, Citation Ranking, and Bringing Order. 1 Introduction and Motivation 2 A Ranking for Every Page on the Web. pages 1–17, 1998.

[28] Xinji Qiang, Guojian Cheng, and Zhen Li. A survey of some classic self-organizing maps with incremental learning. *2010 2nd International Conference on Signal Processing Systems*, pages V1–804–V1–809, July 2010. doi: 10.1109/ICSPS.2010.5555247. URL `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5555247`.

[29] Kira Radinsky and Eric Horvitz. Mining the web to predict future events. *Proceedings of the sixth ACM international conference on Web search and data mining - WSDM '13*, page 255, 2013. doi: 10.1145/2433396.2433431. URL `http://dl.acm.org/citation.cfm?doid=2433396.2433431`.

[30] A Rauber, M Dittenbach, and D Merkl. Automatically Detecting and Organizing Documents into Topic Hierarchies: A Neural Network Based Approach to Bookshelf Creation and Arrangement. *Research and Advanced Technology . . .*, 135476:1–5, 2000. URL `http://medcontent.metapress.com/index/A65RM03P4874243N.pdfhttp://link.springer.com/chapter/10.1007/3-540-45268-0_37`.

[31] Craig Sayers. Self Organizing Feature Maps and their Applications to Robotics. *Technical Reports (CIS)*, 1991. URL `http://repository.upenn.edu/cgi/viewcontent.cgi?article=1406&context=cis_reports`.

[32] Sylvia Springorum, SS im Walde, and Jason Utt. Detecting Polysemy in Hard and Soft Cluster Analyses of German Preposition Vector Spaces. *aclweb.org*, 1998. URL `http://www.aclweb.org/anthology/I/I13/I13-1072.pdf`.

[33] Moritz Sudhof. Politics, Twitter, and information discovery: Using content and link structures to cluster users based on issue framing. *stanford.edu*, pages 67–76, 2011. URL `http://www.stanford.edu/group/journal/cgi-bin/wordpress/wp-content/uploads/2012/09/Sudhof_Eng_2012.pdf`.

[34] Ke Tao, Fabian Abel, Claudia Hauff, and GJ Houben. What makes a tweet relevant for a topic? *Making Sense of Microposts (# . . .*, pages 49–56, 2012. URL `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.232.2188&rep=rep1&type=pdf#page=54`.

[35] Jianshu Weng, EP Lim, Jing Jiang, and Q He. Twitterrank: finding topic-sensitive influential twitterers. *. . . conference on Web search and data . . .*, 2010. URL `http://dl.acm.org/citation.cfm?id=1718520`.