HOMOPHILIC SELF ORGANIZING FEATURE MAPS: FINDING TOPICS ON SOCIALY CONNECTED DATA, USING SOCIAL NETWORK RELATIONS

Bernardo Simões

Dissertação para obtenção do Grau de Mestre em

Engenharia de Redes de Comunicações

Júri

Presidente:	Professor Doutor Paulo Jorge Pires Ferreira
Orientador:	Professor Doutor Pável Calado
Vogais:	Doutor whatever full name 2
	Doutor whatever full name 3

Outubro de 2014

Acknowledgments

Remember that your parents have paid for the last 20 something years of your studies and that your advisor had to read this document.

Abstract

Clustering is a widely used technique in data analysis. In this thesis, a generically lgorithm used for clustering is modified in order to enhance the value of socially connected ententies.

To achieve this, we present RubySOM. A framework for easy construction of custom Self-Organizing Maps. With it, it is possible to dinammically change multiple parts of the algorithm, making it extremlly flexible solution to create, train and run custom implementations of the algorithm.

With RubySOM, a relational aware version of the SOM algorithm was created in order to better identify topics on the social network twitter.

Keywords

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

Resumo

Palavras Chave

detecção de tópicos, twitter, mapas auto organizados, classificação, agrupamento

Contents

List of Figures

List of Tables

List of Acronyms

SOM Self-Organizing Maps

NLP Natural Language Processing

U-Matrix Unified Distance Matrix

TF-IDF Term Frequency–Inverse Document Frequency

LDA Latent Dirichlet Allocation

TDT Topic Detection and Tracking

VSM Vector Space Model

IR Information Retrieval

ML Machine Learning

ANN Artificial Neural Network

MDS Multi Dimensional Scalling

PCA Principle Component Analysis

URL Uniform Resource Locator

JSON JavaScript Object Notation

CSV Comma Separated Values

1
Introduction

Contents

With the evolution of social network websites like Facebook and Twitter, the amount of pertinent content about a specif issue is increasing dramatically, which calls for new ways to make catalog this sense of and data. The usage of social networks for branding quality and on-line marketing is specially compelling since 19% of all tweets [14] and 32% of blog posts [27] are about brands or products. Nevertheless, finding topic sensitive information on social networks is extremely complicated due to the fact that documents have very little content, slang vocabulary ,orthographic mistakes and abbreviations. Asur, Sitaram and Huberman [2] successfully predicted box-office revenues by monitoring the rate of creation of new topics based on debuting movies. Their work outperformed some traditional market-based predictors.

Thus academic and enterprise worlds started looking at Machine Learning for new ways to achieve revenue or simply explore and discover patterns in data. As a consequence, the Machine Learning course at Standford is the one with more students enrolling in the year of 2014 [1] with more than 760 students enrolled.

Using unsupervised Machine Learning (ML) Le et al. [21] was able to achieved 81.7% accuracy in detecting human faces, 76.7% accuracy when identifying human body parts and 74.8% accuracy when identifying cats. He used a 9-layered locally connected sparse auto-encoder 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). This dataset was trained using model parallelism and asynchronous SGD on a cluster with 1,000 machines (16,000 cores) during three days. Even though the amount of computing power used in this project was of several order of magnitude, it is remarkable how an unsupervised algorithm could achieve such results.

Even though a lot of solutions have arisen in order to automate real time searches, topic categorization and many other data intensive tasks are still done manually. Twitter still uses humans to deliver ads to trending queries, states Edwin Chen's Data Scientist responsible for ads quality at Twitter. On his blog post [2], Edwin Chen describes the process of delivering real time adds to trending queries at Twitter. The main problems that arise in the Twitter platform in order to identify rising topics are:

- The queries people perform have never before been seen, so it is 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 happens, people immediately come to Twitter in order to know what is happening in a determined place. Twitter solves this issue by monitoring which queries

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

[2] http://blog.echen.me/2013/01/08/improving-twitter-search-with-real-time-human-computation/

are currently popular in real time, using a Storm topology [3]. After the queries are identified, they are sent to a Thrift API [4] that dispatches the query to Amazon's Mechanical Turk service [5] where real people will be asked a variety of questions about the query.

Social Media Analytics is another raising topic that draws from Social Network Analysis [17], ML, Data Mining [37], Information Retrieval (IR) [31], and Natural Language Processing (NLP). As stated Melville et al. [27], 32% of the 200 million active bloggers, write about opinions on products and brands, while 71% of 625 million Internet users read blogs and 78% of respondents put their trust in the opinion of other consumers. In comparison, traditional advertising is only trusted by 57% of consumers. This kind of data drives companies to Social Media Analytics as 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[6] or ThoughtBuzz[7], but also solutions from the old players like IBM[8] and SAS[9]

Its also important to notice that in the last few years Data Science/Analysis has been a trending topic, mostly due to the fact that big dot-com companies have been having high revenues by exploiting user specific information in order to deliver ads and sell products. Not surprisingly that if you look that in the top ten ebooks sold by O'Reilly throughout 2013, four are about data science [10].

1.1 Motivation

Clustering analysis has been widely used throughout the times, from its first occurrence in England, where John Snow was able to map a wider amount of people infected with cholera to a well in the center of London. Nowadays the applications are endless, and fields where it is applied are quite vast.

Specifically with a greater amount of people describing events around them, and their lives on social media, it is increasingly more challenging to categorize this data, due to its sparsity and volume.

However, data generated on social networks, have more information than simply written text, on a tweet, or a photo published on Facebook. Data generated in social network is connected by entities, and these entities tend to be closer to other entities with the same kind of interests [26]. In this thesis we will explore how a clustering algorithm can be altered in order to add social relevance to its fed data. By focusing on using an unsupervised learning technique based on

[3] http://storm-project.net/
[4] http://thrift.apache.org/
[5] https://www.mturk.com/mturk/
[6] https://sumall.com/
[7] http://www.thoughtbuzz.net/
[8] http://www-01.ibm.com/software/analytics/solutions/customer-analytics/social-media-analytics/
[9] http://www.sas.com/software/customer-intelligence/social-media-analytics.html
[10] http://shop.oreilly.com/category/deals/bestoforeillydotd.do?code=DEAL&cmp=twnabooksvideosinfo-authornote_best_of_2013

neural networks named Self-Organizing Maps (SOM) [19] 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.

1.2 Objectives

The main objective of this project is to find topics on tweets by contextualizing the social network involving the person that authored the tweet in the clustering process.

We start by building a dataset, in order to train the SOM, that will later classify each future tweet that arrives on the network without further delay.

After creating the dataset, we will try to find clusters of topics using the default SOM approach, converting each tweet to Vector Space Model (VSM). After analyzing the results from the default SOM approach, the algorithm will be changed in order to give relevance to the fact that there is a relationship between authors of tweets.

1.3 Contributions

The main contributions of this work are as follow:

- A method to enhance topic discovery on data with small text corpus and high social significance.
- A framework to easily edit multiple parts of a SOM algorithm, by passing high order functions as configuration.
- To develop highly customizable infinite tweeter crawler that preserves the social network.

1.4 Dissertation outline

Explain how did you organized your thesis.

2

Background

Contents

In this section, we will start by generally describing what clustering is and how it works. We will then outline how SOM [19] perform, which is the document clustering algorithm used on this thesis.

2.1 Document Clustering

Document clustering is an optimal division of documents into categories without prior knowledge of the data that is being organized, based only on the similarity between them. Due to the fact that no prior knowledge of the data has to be known, document clustering is labeled as Unsupervised ML [11].

Liu et al. [23] asserted that document clustering can be used in a variety of Computer Science fields, such as:

- Natural Language Preprocessing.
- Automatic Summarization.
- User preference mining.
- Improving text classification results.

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.

The clustering process usually works as described in Figure 2.1. In the first step, a data set must be provided with the documents to be clustered. The second step is where non relevant words are removed from the documents, to improve clustering quality [15].

The third step is characterized by converting the keywords of each document into vectors. The most common model used for this task is VSM. In VSM, each vector dimension represents one detected keyword and each document is represented by the vector of keywords in the feature space. This process is illustrated in Figure 2.2 and works in the following way:

- **First step:** string tokenization, and token selection. In this case, stop words and repeated words will be removed.
- **Second step:** string to VSM conversion. Each different word will correspond to a position in the array, and its value will correspond to the number of occurrences.

There are many clustering algorithms. In the following section we will describe the particullar case of the SOM algorithm, the solution used in our work.

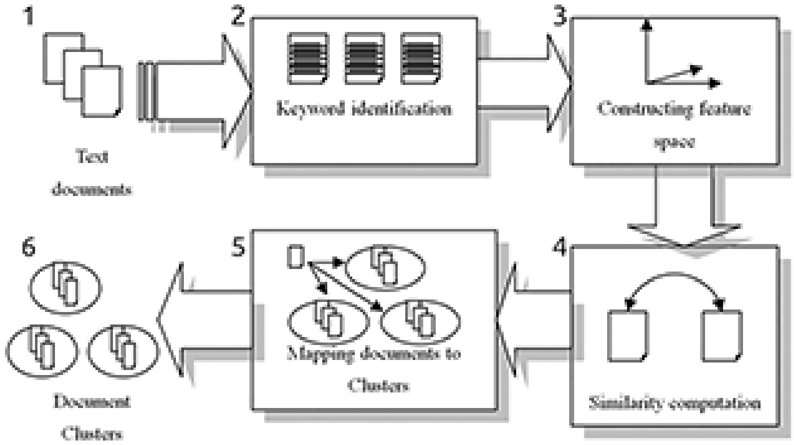


Figure 2.1: Text clustering main framework [9]

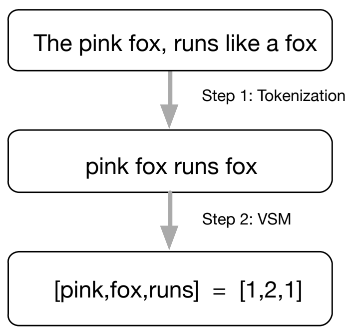


Figure 2.2: Text tokenization and transformation to Vector Space Model.

2.2 The Self-Organizing Map

SOM are a two layer recurrent Artificial Neural Network (ANN) that has the desired property of topology preservation, thus mimicking the way the cortex of highly developed animals brains work. SOM allow cluster visualization of multi-dimensional data, similar to methods such as Multi Dimensional Scalling (MDS) [20] and Principle Component Analysis (PCA) [13] .

Bação et al. [3] described the basic idea behind SOM as a mapping between input data patterns into a n-dimensional grid of neurons, or units. That grid is also know as the output space, as opposed to the initial space — input space — where the input patterns reside. An illustration of both spaces can be seen in Figure 2.4.

SOMs work in a similar way as is thought the human brain works. Analogously to the human brain, SOMs also have a set of neurons that, through learning experience, specialize in the identification of certain types of patterns. These neurons are responsible for categorizing the input patterns for which they are responsible to identify. Nearby neurons will be organized by similarity, which will cause similar patterns to activate similar areas of the SOM. With this topology preserving mapping, the SOM organizes 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 throughout the mapping process. Output neurons are displayed in an N dimensional grid, generally rectangular, but other structures are possible, such as hexagonal or octagonal. The grid of neurons, in the 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.

Before describing the algorithm, it is important to define two key aspects of the SOM: the learning rate and the quantization error. The learning rate is a function that will be decreased 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 in progressively smaller steps. 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 winning neuron is also particularly relevant to the topology of the SOM, deeply affecting the unfolding of the output space as stated by Bação et al. [3].

The learning phase is characterized by the Algorithm 1, which works the following way:

- **On line 1:** neuron closer to the input pattern is selected. The euclidian distance (Eq. 2.1) is generally used.

$$Dist = \sqrt{\sum_{i=0}^{i=n}(V_i - W_i)^2} \tag{2.1}$$

- **On line 2:** the winning neuron (p) previously selected on line 1 is updated, in order to better represent the inputt pattern — this process is represented on Figure 2.3. Also, all

Algorithm 1: Self-Organizing Map [19]

Data: Input patterns $X = \{\overrightarrow{x_1}, \ldots, \overrightarrow{x_N}\}$, number of iterations t_{max}, neighborhood function $\sigma(t)$, learning rate $\epsilon(t)$

Result: Trainned map and clustered input patterns

Randomly initialize neurons, $w_i \in \mathbb{R}^D, \forall i$

for $t = 1 \; to \; t_{max}$ **do**

```
    Randomly draw an input pattern, x_d
1   p = arg min_i{||x_d - w_i||}
2   w_i = w_i + ε(t) · h_ip(t) · (x_d - w_i), ∀i
3   σ(t) = σ_0(σ_f/σ_0)^(t/t_max)
4   ε(t) = ε_0(ε_f/ε_0)^(t/t_max)
5   t ← t + 1
```

Randomly draw an input pattern, $\overrightarrow{x_d}$

1 $p = \arg min_i\{\|\overrightarrow{x_d} - \overrightarrow{w_i}\|\}$

2 $\overrightarrow{w_i} = \overrightarrow{w_i} + \epsilon(t) \cdot h_{ip}(t) \cdot (\overrightarrow{x_d} - \overrightarrow{w_i}), \forall i$

3 $\sigma(t) = \sigma_0(\sigma_f/\sigma_0)^{t/t_{max}}$

4 $\epsilon(t) = \epsilon_0(\epsilon_f/\epsilon_0)^{t/t_{max}}$

5 $t \leftarrow t + 1$

other neurons inside a specific radius will also be updated — this process is described in Figure 2.4. Each neuron is updated with a different rate of influence determined by how far away it is from the winning neuron, which is defined by the neighborhood influence function $h_i p(t)$, the Gaussian (Eq. 2.2) is often used.

$$h_{ip}(t) = \exp - \frac{|\overrightarrow{a_i} - \overrightarrow{a_p}|^2}{\sigma^2(t)} \tag{2.2}$$

- **On line 3:** the size of the radius will be updated.
- **On line 4:** the learning rate is updated.
- **On line 5:** the number of iterations is incremented.

In order for the algorithm to converge, the learning rate and the radius of the neighborhood need to decrease at a given rate. Generally exponential decay is used.

The prediction phase can start after the model is learned. On the prediction phase, new input patterns can be quickly assigned to the SOM, without need to apply the learning rate to the winning neuron and his neighbors. Due to the fact that the input pattern will be assigned to the cluster that is mapped by the nearest neuron. Thus, it is very easy and fast to classify new data now. As stated by Liu et al. [23], the advantages of using SOM are: data noise immunity, easy to visualize data, and parallel processing.

In order to visually interpret the result of the SOM, Unified Distance Matrix (U-Matrix) method may be used [3]. The U-Matrix is a representation of the SOM, in which, the average topological error between a neuron and all the input patterns he represents is displayed in a color scale proportional to the maximum and minimum values obtained in the SOM.

Computing a U-Matrix is done by the Algorithm 2. Essentially, each neuron is responsible for representing a cluster of input patterns. The better a neuron represents an input patter and the smaller the distance between them both. Therefor, the better a neuron represents a group of input patterns, the smaller the average distance between himself and all the input patterns it represents. For representation purposes, after all the averages are computed a lighter color is assigned to the lower average, and the darkest color to the highest. Thus, all the values in between must have a color proportional to its own value, in reference to the edge values. An example of an U-Matrix can be seen in Figure 2.5(b).

2.2.1 Quantization Error

SOM training is always subject to some variability due to multiple causes, like the sensitivity of initial conditions, convergence to local minima and sampling variability, as stated by Bodt et al. [6]. This subsection will present statistical tools to measure the quality of the SOM, by measuring its quantization error.

The SOM Quantization Error is the mean of all Euclidean distances between the observed data points and their corresponding winning neuron. This value might vary depending on the initialization neurons or the order of the input data fed into the SOM while the training is occurring. When applied to an individual input data, represents how well a neuron is representing input data. Since the SOM Quantization Error represents the mean of all quantization errors from all the input data, generally, the lower the error is the best the SOM was trained.

No general formula exists to minimize quantization error [6] . What is generally done is just to change the number and values of the starting neurons and the order of the input data in order to train multiple SOMs. In the end the SOM with the lowest quantization error is chosen. In this project since multiple approaches to the SOM algorithm and data representation will be tested, as described in Section **??**,and the ones having the lower quantization error will be selected for the prototype.

Algorithm 2: U-Matrix

Data: $W = \{\overrightarrow{w_{0,0}}, \ldots, \overrightarrow{w_{n,n}}\}$ are the trained neurons

$D_{i,j}$ be the input patterns represented with neuron $w_{i,j}$

U is an empty matrix with size $2n-1.2n-1$

Result: U-Matrix

/* Initialize U by adding the trained neurons */

for $w_{ij} = \overrightarrow{w_{00}}$ to $\overrightarrow{w_{n,n}}$ **do**
  $U_{i*2,j*2} \leftarrow w_{i,j}$

/* Calculate the distance between every adjacent neurons, and apply it to the square between them */

for $i = 0$ to U_{max} **do**
  for $j = 0$ to U_{max} **do**
    if $l+1 < m || j+1 < m$ **then**
      $U_{i+1,j} = \|u_{i,j} - u_{i+2,j}\|\ U_{i,j+1} = \|u_{i,j} - u_{i,j+2}\|$
      $U_{i+1,j+1} = \frac{\|u_{i,j} - u_{i+2,j+2}\| + \|u_{i+2,j} - u_{i,j+2}\|}{2}$
    $j \leftarrow j+1$
  $i \leftarrow i+1$

/* Substitute the neurons for an average of surrounding distances */

for $i = 0$ to U_{max} **do**
  for $j = 0$ to U_{max} **do**
    $u_{ij} = avg(Adj[u_{ij}])$
    $j \leftarrow j+1$
  $i \leftarrow i+1$

/* convert the distances to color */

$WHITE = 255$

$BLACK = 0$

$u_{max} \leftarrow max(U)$

$u_{min} \leftarrow min(U)$

for $u_{ij} = u_{00}$ to $u_{n,n}$ **do**
  $U_{i,j} \leftarrow (1 - \frac{u_{i,j} - u_{min}}{u_{max} - u_{min}}) * WHITE$

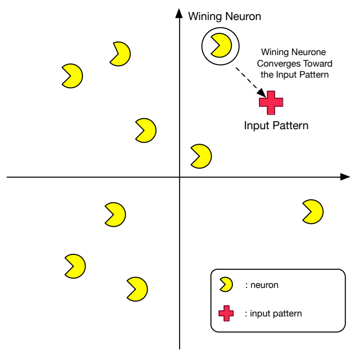


Figure 2.3: Winning neuron converging at learning rate

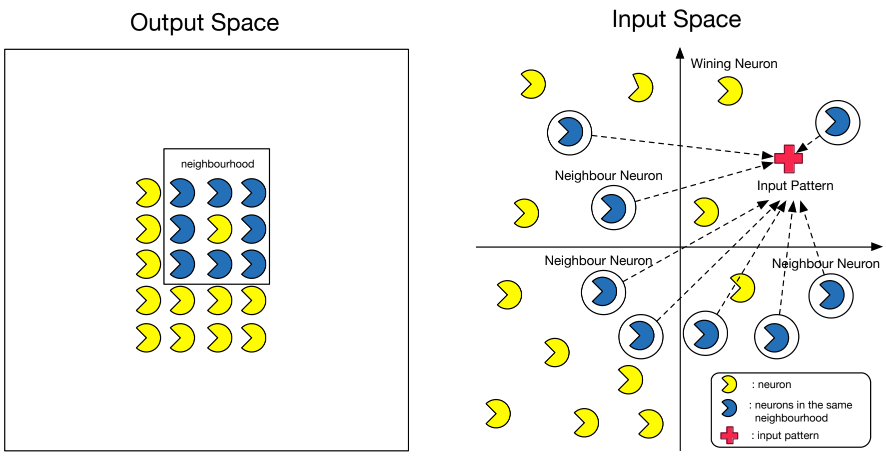
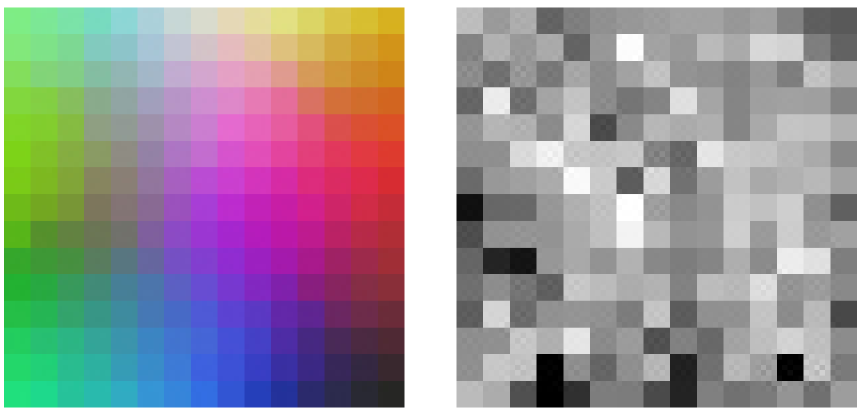


Figure 2.4: On the left the output space neighborhood, on the right the neighbors of the winning neuron converging in the direction of the input pattern



(a) SOM ouputspace trained with random RGB vectors

(b) U-Matrix representing how well each neuron maps to input patterns it represents

Figure 2.5: U-Matrix and SOM output space, computed by training the SOM during 400 epochs, with 1500 random input patterns representing an RGB color.

3

State of the art

Contents

This section provides insight of work done in several research areas that are related to the project. In section 3.1, work done using SOM maps will be described. Section 3.2 and 3.3 are dedicated to data clustering and to data mining specifically on Twitter [1].

3.1 Self-Organizing Maps

SOM are used in a wide area of applications, from authentication systems [9] through network intrusion detection [32] and speech recognition and analysis [18]. In this section we highlight some of their applications.

3.1.1 The Geo-SOM

The Geo-SOM, by Bação et al. [3], applies the first law of geography "Everything is related to everything else, but near things are more related than distant things." [35] to the SOM algorithm. In this case, the winning neuron is chosen in a radius defined by the geographic-coordinates of the data, forcing units that are close geographically to be close in the output space.

The algorithm works by defining a variable k which is used as a "geographical tolerance" that forces the winning neuron to be geographically near the input pattern. When $k = 0$, the winning neuron is forced to be the unit geographically closest to the input data, whilst k increases, the tolerance for data with further geographic coordinates, increases as well. k is a geographic radius applied in the output space. When the radius exceeds the size of the output space, every unit is eligible to be the winning neuron, and therefor, we have a regular SOM.

The selection of the winning neuron is done in two steps. First, geographic neurons inside the tolerance k with the input data as a center are selected. Only after that, comparisons are made with the rest of data present in the input data. The representation of the Geo-SOM can be seen in Figure 3.1, where the units considered for the best match are defined by a sort of geographic radius defined by k, whilst in the original SOM, the winning neuron could have been any of the units presented on the figure.

The Geo-SOM approach to the alteration of the default SOM algorithm is specially interesting due to the fact that this thesis objective is also to give relevance to data patterns that are not located in the same space as the trained data. In a way, what we are trying to achieve is similar to the work by Bação et al. [3] but changing the geographic relevance in data by a social relevance.

3.1.2 Detecting Hidden Patterns on Twitter Usage

Cheon and Lee [8] analyzed hidden patterns in the usage of twitter. In their study, they started by collecting data from the twitter API of 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

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

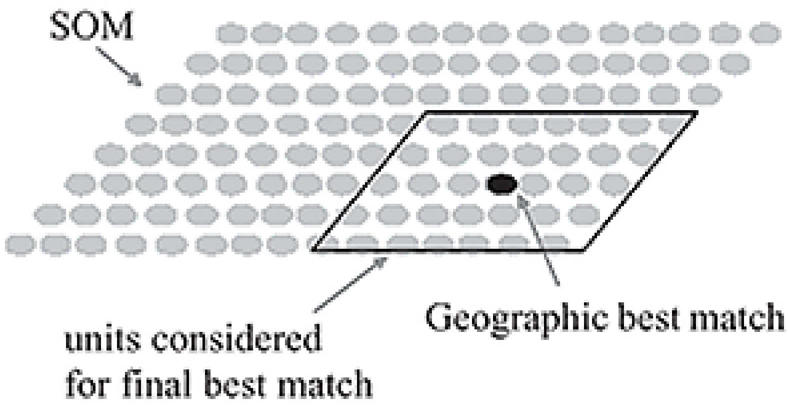


Figure 3.1: Geo-SOM structure, where the units considered to be the winning neuron are constrained by the geographic coordinates of the data, from Bação et al. [3]

present on the tweet, but also by cross referencing with other social websites and with the twitter user profile information. The signals retrieved from the social network can be seen in Table 3.1.

Table 3.1: Twitter Signals

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

Parece me interessante descrever quais as caracteristicas diferenciadoras dos clusters. By applying a SOM, they could find four demographical clusters during the Iran 2009 Election. The first cluster was characterized by young web-based Iranians, with twitter accounts not older than three months with a high frequency of replies. The second cluster was mainly compound of web users from Iran accounts older that three 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 fifty tweets a day with external URIs and the accounts were not older than a day or so.

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

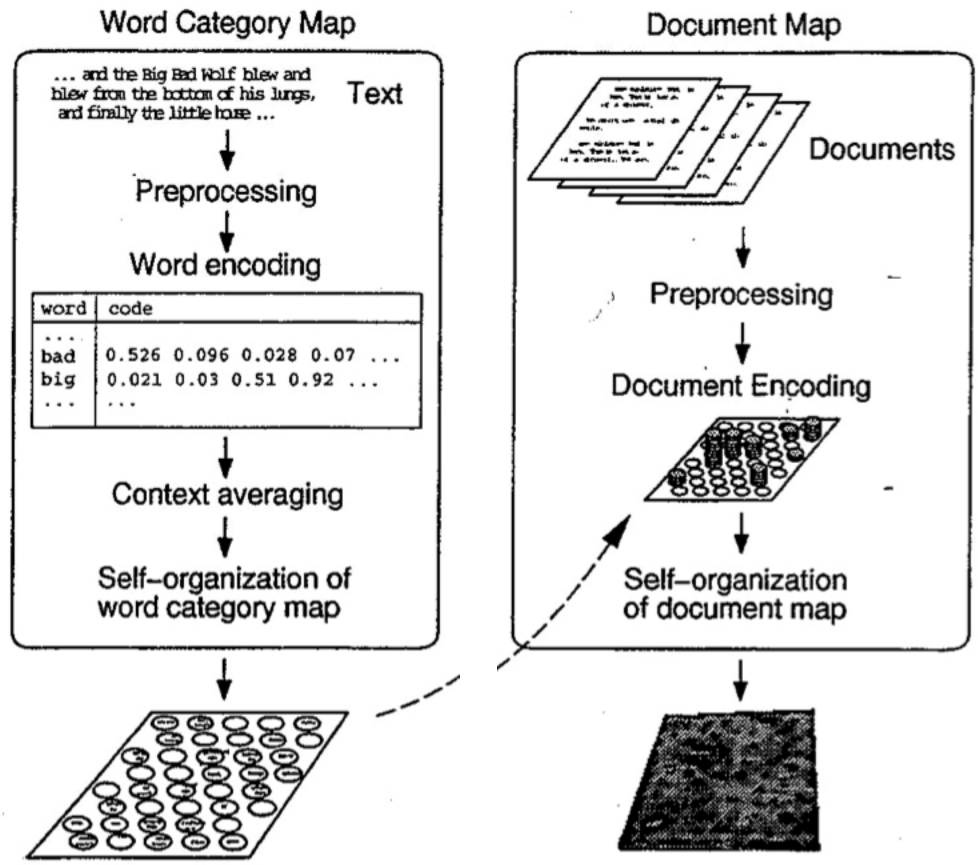
3.1.3 WEBSOM

Honkela et al. [12] developed a new approach to automatically order arbitrary, free from textual, document collections, using two different SOMs. The first SOM is called word category map and its used to find words that have similar meaning, while the second SOM, called document map, is the one actually used to cluster the documents.

The WEBSOM was not based on keywords and boolean expressions, instead, words with the same meaning are encoded in a word category map (Fig 3.2(a)), where placement and frequency in documents is taken into account. This way it is possible to remove words with similar meaning — greatly reducing the VSM size making it possible to train the document map in a scalable way.

3.2 Topic Detection and Tracking with Clustering

Allan [1] defined Topic Detection and Tracking (TDT) as: “TDT research begins with a constantly arriving stream of text from newswire and from automatic speech-to-text systems that are monitoring selected television, radio, Web broadcast news shows. Roughly speaking, the goal of TDT is to break the text down into individual news stories, to monitor the stories for the events that have not been seen before, and to gather stories into groups that each discuss a single news topic”.



(a) The word cathegory map, a SOM where word frequency and placement is used for encoding text

(b) The document map, organized based on documents enconded with the word cathegory map.

Figure 3.2: Basic architecture of the WEBSOM method, from [12]

Nowadays, due to the rapid adaptation of 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 [2], which in "Internet Years" was ages ago.

The challenge of TDT is evermore focused on online generated documents, and in new forms to be able to track and categorize all the information that is continuously being generated. Many TDT techniques have been proposed, a significant amount of them rely on the Term Frequency–Inverse Document Frequency (TF-IDF) [4]. Because tweets are very small, often with typos or slang words, and because the same tweet might be written in multiple languages, TF-IDF is not particularly adequate for topic detection on twitter. In this subsection, we will take a look at multiple methods of topic detection in general, and also specifically on the Twitter social network.

3.2.1 Topic and Trending Detection

Cataldi et al. [7] proposed 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 algorithm [29] to the users follower/followee relationship, in order to find the most influential users on the network. Then, the most trending topics are calculates, by relating social influence, word co-occurrence and time frame. In the end, an interface was created where it would be possible to navigate, through hot topics in a given time frame. Topic labeling was not automatic and was implicit by the time frame of an event.

Weng et al. [36] also used the PageRank algorithm to find the most influential twitter users on a certain topic. However, using a different approach, they represent each twitter user as a bag of words comprising of all the tweets that they have posted, and applied Latent Dirichlet Allocation (LDA) [5] in order to find topics in which users are interested in. Finally, it was possible to prove that follower/followee relations on twitter are not just casual, but that people actually follow other people to whom they have some resemblance or common interest. This concept is called homophily and will be further explored on this thesis.

3.3 Twitter Data Mining

In this subsection, we will focus on work done on the Twitter social network in order to leverage insights on how the public data available from the website can be explored.

3.3.1 Tweets Implicit Data

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

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

Table 3.2: Tao et al. [34] tweet characteristics hypothesis versus influence

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

amount of tweets generated per day is around 140 million[3], which means that it is very hard to do a deep analysis of the semantics and content of individual tweet, and that only the more appropriate signals should be evaluated. For this reason, 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 their work, they present premises that seem intuitively true and proves they actually are relevant through a comparison of multiple precision and recall values. Their results on feature comparison are summarized in Table 3.2. The first column consists of all the made hypothesis categorized by type, and the second column tells if the data used actually influenced in precision and recall results. Tao et al. [34] also compared results, 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 [25] also approached the issue of having very little content on tweets in order to categorize them, and tried to solve the problem 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 called PL2F [24] is used against the dataset from "Trec Microblog2011" in order to find the best weighting. With this model they were able to improve precision in an order of 0.9, over only analyzing the text contained in the tweets.

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

3.3.2 Tweeter Natural Language Processing

Using standart NLP tools on tweets has been extremely unreliable, due to the fact that microbloging text tends to be full of abbreviations, emojis and smiles . Recently, Owoputi et al. [28] published a NLP library, specific for twitter. As shown in Figure 3.3, ARK Tweet NLP can tag words that are only used in social networks. The tagger was built using maximum entropy Markov model, where a tag is assigned to a word based on the entire tweet text, and the tag assigned to the word to its left. Owoputi et al. [28] state that the tagger has a 93.2% accuracy. By using NLP tools, it is possible to reduce the dimension of VSM space by only choosing words that are relevant, like common nouns, hashtags and proper nouns. This will not only yield better results by removing tweets that have no content, and therefor, cannot be categorized, but will also increase performance during training due to the reduced dimensions caused by less use of words.

ikr	smh	he	asked	fir	yo	last
!	G	O	V	P	D	A
name	so	he	can	add	u	on
N	P	O	V	V	O	P
fb	lololol					
^	!					

Figure 3.3: Tweet automatically tagged with ARK Tweet NLP. ! stands for interjection, while V stands for verbs and D for determiner. The full table of tags can be found in [28].

3.3.3 Rapidly Changing Trends

Due to the real time nature of Twitter, using typical retrieval model, that relies on term frequency models like Okapi BM25 or language modeling cannot be applied, as stated by Lin and Mishne [22]. The study of topic endurance on the social network proved that topics are presented in bursts of queries and mentions. In addition the typical usage of twitter for search is not the same of Google. When users are searching on twitter, they want to find out what is happening in that moment, meaning that classification techniques based on past events cannot respond this kind of 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 3.4.

By including social features into clustering algorithms, it might be possible to discover interesting rising topics to a specific specific user, by categorizing them through a trained SOM.

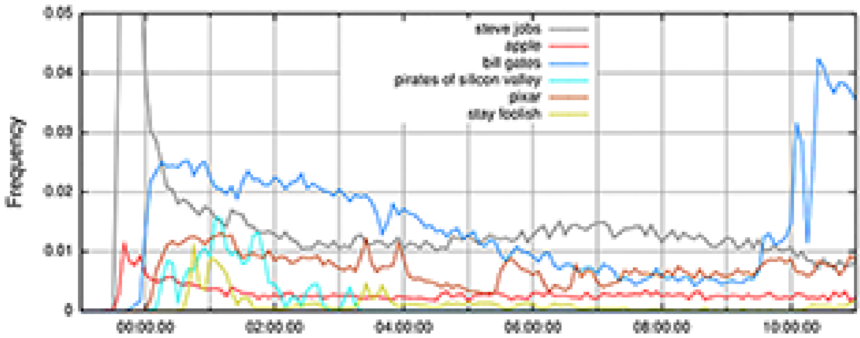


Figure 3.4: The Churn effect: Frequencies of queries related to Steve Jobs death over a 12 hour period in 5-minute intervals, normalized to the total number of queries in the interval. At its peak, the query “steve jobs” reaches 0.15 (15% of the query stream); Graph taken from [22]

3.4 Summary

Ending section summarizing the chapter is typically a good idea.

Ensure that the next chapter starts in a odd page

4

Clustering Tweets with Self Organizing Maps

Contents

4.1 Adapting the SOM to the Social Web

transform tweets in binary matrix's and train them Tweets categorization with **SOMs!** (**SOMs!**) requires a dataset to work on. Due to systematic API restrictions, which where implemented in order to protect Twitter business model, gathering a dataset that directly maps to the social network is a great challenge by itself. In order to categorize tweets with **SOMs!** first we used a dataset provided by INESC-ID. The dataset had almost 1TB of data in JSON format.

4.2 Crawling Twitter for Social Relations

The INESC twitter dataset, was a good dataset to start analyzing the complexity of clustering tweets. On Chapter 5.1 we deeply analyzed the characteristics of the INESC twitter dataset, and concluded that it was not optimal to the problem we are trying to solve since it doesn't have the social connections between the authors of the tweets.

4.3 SOM Framework

Resume this section

4.3.1 Motivation

When researching ways to extend the SOM algorithm, in order to add social features to the learning process. I found that the number of SOM libraries was not very extense. Even though, programing languages often used in ML and Data Mining, such as Python or C++, have their how implementation of the SOM algorithm. I've found that most of these libraries are made in such a way to be extremely fast, in order to take as much advantage from the hardware they are running on as possible. They often lack the modularity needed to adapt the SOM algorithm to specific problems.

The SOM algorithm has been changed many times in order to better categorize data with specific features, for example Geo-SOM was described in Subsection 3.1.1, the Growing Hierarchical SOM [30], the time adaptive SOM [33], the Ontological SOM [10], and the list goes on…

In order to create the homophilic SOM, described in Section 4.5 we first created a SOM framework that is easy to extend due to be fully object oriented, scripted — even though it can be compiled to run on the JVM — and without C extensions.

4.4 Clustering Socially Connected Data

The default SOM algorithm has no idea whatsoever of the social connections between the tweets, it simply looks at the binary vectors that represent sentences and assigns it to the most similar neuron.

In order to better categorize socially connected data, we propose some alterations to the SOM algorithm in order to make it aware of the social connections between the tweets, and therefor better represent the homophilic behavior present on social networks.

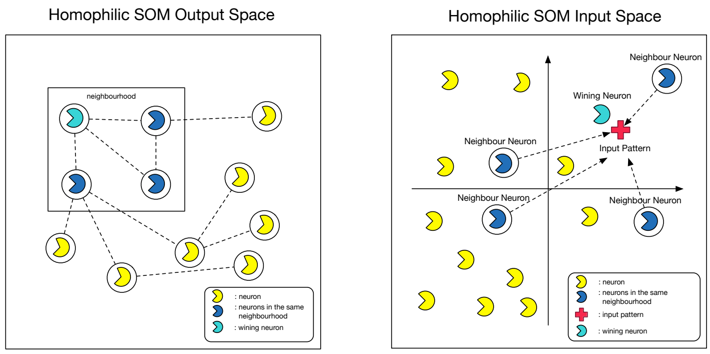
insert homophilic som algorith here

4.5 Homophilic SOM Definition

4.5.1 Output Space

The outputs space is the zone on the SOM algorithm where the neurons reside. It works like a cortex where neurons are scattered in a geometric fashion, generally a square. The output space is generally initialized with random values, with a relatively high learning rate, and also a relatively high number of epochs. The algorithm is made this way in order to be able to identify any type of data that can be represented as vectors.

First we will try to change the output space to better resemblance the social network. In order to do this, the squared grid that defines the output space was changed by the social network connections, and the neurons, are represented by a social network user. This changes are applied in the following way:



(a) The neighbourhood is defined by the relations of followers/followees between the winning neuron and the other neurons

(b) Homophilic input space works in the same way as a normal input space

Figure 4.1: Homophilic SOM output and input space during the learning phase.

- Each neuron is comprised of the text from all the tweets that he authored.
- Each neuron has a unique id, and stores the ids of his followers and followees that are present in the output space.

- During the learning phase, the radius will be defined as the maximum number of hops separating the winning neuron and followers/followees of followers/followees.

insert image of the output space with social features vrs tipical output space

4.5.2 Learning Phase

Like in the default SOM the learning phase is where the output space is trained in order to organize the input data into clusters. Since this algorithm is specific to categorize tweets using social network features, the learning rate, radius and number of epochs used can be greatly reduced in order for the algorithm to converge. The learning phase operates in the following way:

- The distance between the input pattern and all the neurons is calculated. The neuron closest to the input pattern is considered the winning neuron.
- When the winning neuron is selected, he and his social neighbors within k hops, update their representations in the input space, and move closer to the input patter. The Gaussian function (Func. 2.2) is also used in here in order for the neighbors that are closer to the input pattern be significantly more influenced by the input pattern, while the neurons further away are less influenced.
- This process is repeated for a predefined number of epochs. While the number of epochs increases, the learning rate, and number of hops that defines the neighborhood decreases in order for the algorithm to converge.

Just like the default SOM algorithm, after the map is trained, input patterns can be fast assign to the nearest neuron since the neuron positions in the output space are no longer updated.

Link to the learning phase in the algorithm on the main chapter, add images of the training model

4.6 Social Clusters

resume what is written in this chapter

4.6.1 Training

In order to train the Homophilic SOM, we used the crawler defined in Section 3.3. The dataset had the following characteristics: add table with number of users, tweets, tags, on the dataset show amount of time it took to train the SOM show umatrixes of the trainne compare clusters/time and umatrixes of the default SOM and the Homophilic SOM show the tweets in some clusters

5

Evaluation Metrics

Contents

```
{
  "_id" : { "$oid" : "4fa14bc97e5617025fb14787" },
  "text" : "RT @FastCoDesign: A Paintbrush That Works On The iPad
     http://t.co/eWjEZAga (@sensubrushman)",
  "id_str" : "197701817864421376",
  "coordinates" : null,
  "in_reply_to_screen_name" : null,
  "in_reply_to_user_id" : null,
  "possibly_sensitive" : false,
  "favorited" : false,
  "in_reply_to_status_id" : null,
  "source" : "<a href=\"http://www.flipboard.com\" rel=\"nofollow
     \">Flipboard</a>",
  "possibly_sensitive_editable" : true,
  "contributors" : null,
  "retweet_count" : 0,
  "truncated" : false,
  "in_reply_to_status_id_str" : null,
  "geo" : null,
  "in_reply_to_user_id_str" : null,
  "entities" : { Enteties Object },
  "user" : { User object },
  "retweeted" : false,
  "id" : 197701817864421376,
  "place" : null,
  "created_at" : "Wed May 02 14:59:21 +0000 2012" }
```

Figure 5.1: JSON representation of a Tweet.

5.1 Clustering Tweets with Self-Organizing Maps

5.1.1 Twitter Dataset

Each tweet is comprised of multiple parameters. Figure 5.1 shows how a tweet is represented in JSON format. Inside the tweet there is also information about the user whom created the tweet, shown in Figure **??** and entities shown in Figure **??**.

As can be seen in Figure **??** no information about the social relations of the user which emitted the tweet are present. Therefor in order to retrieve the social network in which a user is contained, it will be necessary to connect to the Twitter API. Crawling twitter is discussed in further depth in Chapter **??**.

In order to better understand the dataset at hand, all the JSON files where converted into CSVin a way to reduce the size of the dataset. While tweets where being converted, Uniform Resource Locator (URL) where removed — since most of them where minified in order to fit in less that 140 characters, without translating the minified URL, not a lot of information can be gathered. Also, all tweets that where not identified as being in English where also removed. The tweet shown in JSON format in Figure 5.1 is converted to CSV in Figure 5.2.

```
karmadabaghi,A Paintbrush That Works On The iPad @sensubrushman
```

Figure 5.2: CSV representation of a Tweet. The username is present in the first column and the tweet text on the last.

Add table with dataset characteristics

5.1.2 SOM training

Refer dataset info on Describe naive approach to SOM training in R Describe amount of weird words

5.1.3 Reducing SOM vector size

5.1.3.A Identify Tweets language

Identifying tweets that where not in the English language was done through the usage of Ruby library called whatlanguage [1], which tries to identify one language through Bloom Filters. Inside the tweet there is a field which identifies the user language, we found that x is not acurate. Removing tweets that weren't in the english language reduced the amount of different words in x and therefor will reduce the dimensional size of the SOM.

5.1.3.B Text Manipulation for VSM reduction

all the text techniques used show amount of words reduction with the introduction of a new technique and the amount of time that it takes to be applied

Work done on the INESC twitter dataset with SOMs. SOM implementations used, what where their strong points and weaknesses

5.1.3.C Clustering with Word Selection

Results of SOMs using selected words based on occurrence after applying SVM reduction Results where pretty bad

5.1.4 Clustering with NLP selected words

results of som using arktweet NLP types of tags selected show tweets that had no representation review clusters and results

[1] https://github.com/peterc/whatlanguage

5.1.5 Conclusions

5.2 Twitter Crawler

This problem could be solved in any of the following ways:

- **First aproach:** For each tweet, fetch the user information including the users he is connected to.
- **Second aproach:** Create our own crawler, where the social connections, tweets and users are saved.

When designing the twitter crawler, we took into consideration that it had to be extremely resilient in order to be able to be left alone, crawling the twitter, until told to stop. Also if anything happened to the machine where the crawler was running it would be necessary to return to some previous crawling state, with minimum data loss. Algorithm I NEED TO WRITE THIS ALGORITHM shows the crawler algorithm, which works in the following way:

- **Step 1:** Choose some seed users to start crawling or deserialize a serialized version of the crawler if available.
- **Step 2:** For each seed user get all of his followers, and add them to an array if they haven't yet been crawled.
- **Step 3:** Repeat step one with random users taken from the array on step 2, until API limit is reached.
- **Step 4:** When API limit is reached, print the state of the crawled network, serialize the current state, and wait 15 minutes until it is possible to resume crawling.

5.2.1 Crawler Performance

show number of tweets per second, number user per second, and users reached per second. compare the with the theoretical maximum compare size of the serialized social network with the amount of tweets and users it is getting compare time taken to deserialize with amount of tweets crawled compare time to serialize with amount of tweets crawled

5.3 SOM Framework

The SOM framework was developed in the Ruby programing language [2] due to the desired characteristic of allowing great levels of introspection and being an almost pure object oriented programing language. Due to this characteristics making modifications to core parts of the algorithm is fairly easy.

[2] https://www.ruby-lang.org/en/

The SOM Framework was developed in a test driven fashion, having 100% of its public methods tested and documented for expected behavior. These characteristics, associated with the fact that was published under an open source license, makes it available for other researchers to implement their own SOM variants.

By default, the base SOM algorithm is implemented as described by the Algorithm 1 in Section 2.2.

5.3.1 Clustering Color Vectors

Out of the box, the SOM Framework implements a squared output space, where all residing neurons are manipulated as arrays. It is possible ate any given moment of the training to export the output space to JSON, CSV or to visualize its current U-Matrix. Also during training a progress bar is displayed in order to know how much time will be needed for the training to end.

Due to the features described above, it is possible to train a SOM to identify random colors — RGB vectors — while printing the results. In order to do this we will start by:

- Initializing a SOM object with an output space size of 15 by 15 neurons, which will yield a total of 255 neurons — and directly maps to the maximum number of clusters — and 700 epochs.

- Create 1500 input patterns with size 3 and random values between 0 and 255.

- Tell the SOM to print its state at the end of each epoch.

The machine used for training had the hardware specifications outlined in table 5.2

Table 5.1: Test machine one specs

Operative System	OSX 10.9.5
Memory	8 GB, 1067MHz DDR3
Processor	2,4GHz Intel Core 2 Duo
Hard Drive	128GB SSD

A summary of the training is specified in Table **??**

Table 5.2: SOM trainning resumed

Number of Neurons	225
Output Space Size	15x15
Number of Input Patterns	1500
VSM size of Input Patterns and Neurons	3
Number of Epochs	600
Training Duration	14 hours
Type of Train	print each epoch training
Initial learning rate	0.6
Initial Radius	8

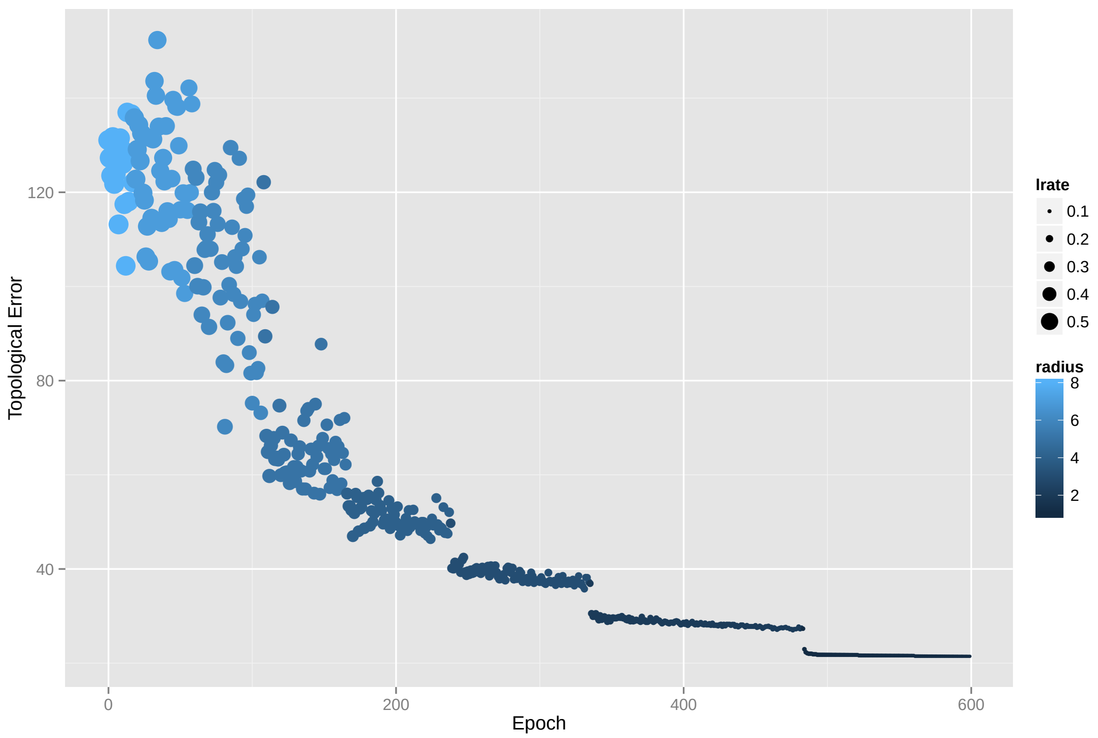


Figure 5.3: Changes in topological error throughout the SOM training, lrate stands for learning rate, and radius for radius applied to the winning neuron

5.3.2 Benchmarking

5.4 Homophilic SOM

5.5 Conclusions

Only these methods are not enough to cluster tweets

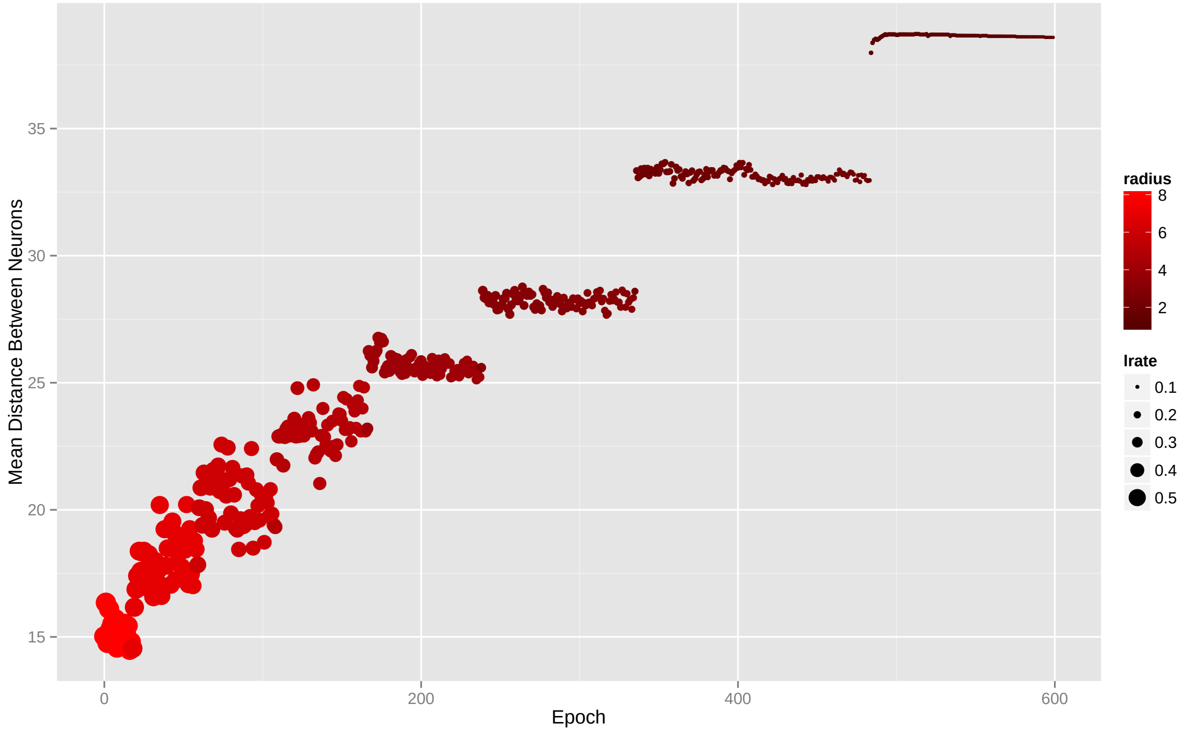
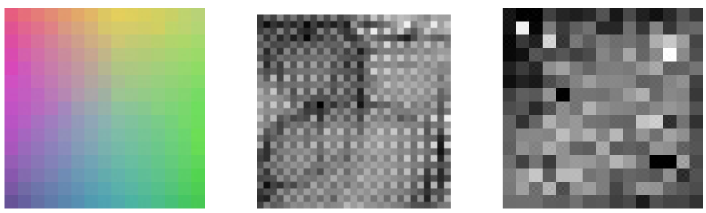
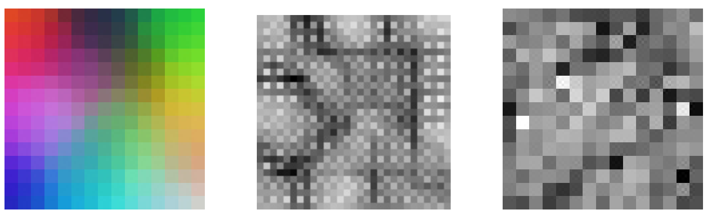


Figure 5.4: Changes in the average distance between neurons, throughout the SOM training



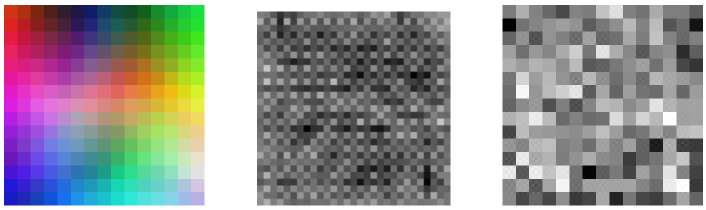
(a) Output Space **(b)** U-Matrix **(c)** Q-Matrix

Figure 5.5: SOM state after first epoch of training. Its learning rate is at 0.598, and radius at 8.



(a) Output Space **(b)** U-Matrix **(c)** Q-Matrix

Figure 5.6: SOM state after second epoch of training. Its learning rate is at 0.22, and radius at 3.



(a) Output Space **(b)** U-Matrix **(c)** Q-Matrix

Figure 5.7: SOM state after third epoch of training. Its learning rate is at 0.081, and radius at 1.

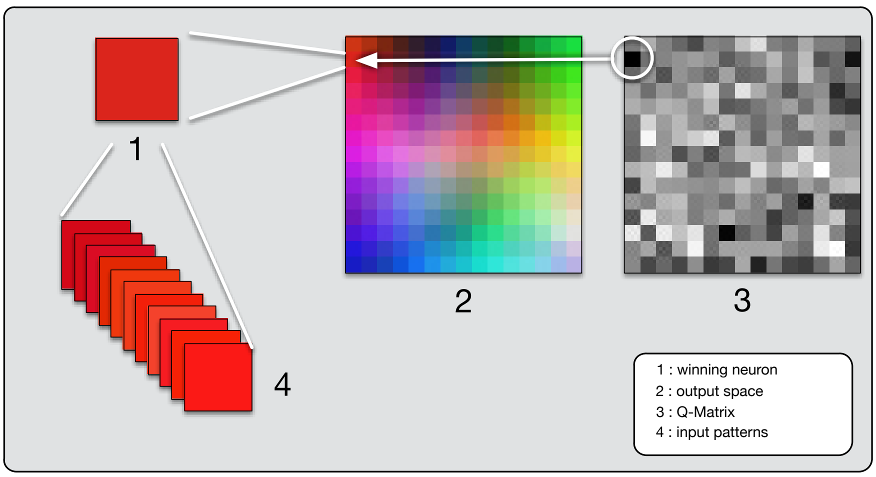


Figure 5.8: Input patterns associated with the neuron with maximum topological error –31. Even though the neuron has the biggest topological error of all neurons, it still has a good representation of the input patterns. The colors in this image are not figurative, and represent the entities at the end of trainning

6

Conclusions and Future Work

Draw your conclusions here and sell your work. Trasmit to the juri how hard it was to develop the presented work.

A future work section is usually here.

Bibliography

[1] Allan, J. (2002). Topic detection and tracking: event-based information organization, volume 12. Springer.

[2] Asur, Sitaram and Huberman, B. A. (2010). Predicting the future with social media. Proceedings of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, 01:492–499.

[3] Bação, F., Lobo, V., and Painho, M. (2005). The self-organizing map, the Geo-SOM, and relevant variants for geosciences. Computers & Geosciences, 31(2):155–163.

[4] Baeza-Yates, R. A. and Ribeiro-Neto, B. (1999). Modern Information Retrieval. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

[5] Blei, D., Ng, A., and Jordan, M. (2003). Latent dirichlet allocation. the Journal of machine Learning research, 3:993–1022.

[6] Bodt, E. D., Cottrell, M., and Verleysen, M. (2005). Statistical tools to assess the reliability of self-organizing maps. Neural networks, 15(8-9):967–978.

[7] Cataldi, M., Di Caro, L., and Schifanella, C. (2010). Emerging topic detection on twitter based on temporal and social terms evaluation. In Proceedings of the Tenth International Workshop on Multimedia Data Mining, MDMKDD '10, pages 4:1–4:10, New York, NY, USA. ACM.

[8] Cheong, M. and Lee, V. (2010). A Study on Detecting Patterns in Twitter Intra-topic User and Message Clustering. 2010 20th International Conference on Pattern Recognition, pages 3125–3128.

[9] Dozono, H. (2012). Application of Self Organizing Maps to Multi Modal Adaptive Authentication System Using Behavior Biometrics. Applications of Self-Organizing Maps, pages 120–141.

[10] Havens, T., Keller, J., and Popescu, M. (2010). Computing with words with the ontological self-organizing map. Fuzzy Systems, IEEE Transactions on, 18(3):473–485.

[11] Hinton, G. E. and Sejnowski, T. J. (1999). Unsupervised learning: foundations of neural computation. MIT press.

[12] Honkela, T., Kaski, S., Lagus, K., and Kohonen, T. (1997). Websom - self-organizing maps of document collections. In Neurocomputing, pages 101–117.

[13] Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components.

[14] Jansen, B. J., Zhang, M., Sobel, K., and Chowdury, A. (2009). Twitter power: Tweets as electronic word of mouth. J. Am. Soc. Inf. Sci. Technol., 60(11):2169–2188.

[15] Kang, S.-S. (2003). Keyword-based document clustering. Proceedings of the sixth international workshop on Information retrieval with Asian languages -, 11:132–137.

[16] Kiviluoto, K. (1996). Topology Preservation in Self-Organizing Maps. Neural Networks, 1996., IEEE International . . . .

[17] Knoke, D. and Yang, S. (2008). Social network analysis, volume 154. Sage.

[18] Kohonen, T. (1988). The 'neural' phonetic typewriter. Computer, 21(3):11–22.

[19] Kohonen, T. (1990). The self-organizing map. Proceedings of the IEEE.

[20] Kruskal, J. and Wish, M. (1978). Multidimensional Scaling. Sage Publications.

[21] Le, Q. V., Ranzato, M., Monga, R., Devin, M., Corrado, G., Chen, K., Dean, J., and Ng, A. Y. (2012). Building high-level features using large scale unsupervised learning. In ICML. icml.cc / Omnipress.

[22] Lin, J. and Mishne, G. (2012). A Study of" Churn" in Tweets and Real-Time Search Queries (Extended Version). arXiv preprint arXiv:1205.6855.

[23] Liu, Y., Liu, M., and Wang, X. (2012). Application of Self-Organizing Maps in Text Clustering: A Review. Applications of Self-Organizing Maps, pages 205–219.

[24] Macdonald, C., Plachouras, V., He, B., Lioma, C., and Ounis, I. (2006). University of glasgow at webclef 2005: Experiments in per-field normalisation and language specific stemming. In Peters, C., Gey, F., Gonzalo, J., Müller, H., Jones, G., Kluck, M., Magnini, B., and de Rijke, M., editors, Accessing Multilingual Information Repositories, volume 4022 of Lecture Notes in Computer Science, pages 898–907. Springer Berlin Heidelberg.

[25] McCreadie, R. and Macdonald, C. (2013). Relevance in microblogs: Enhancing tweet retrieval using hyperlinked documents. In Proceedings of the 10th Conference on Open Research Areas in Information Retrieval, OAIR '13, pages 189–196, Paris, France, France. LE CENTRE DE HAUTES ETUDES INTERNATIONALES D'INFORMATIQUE DOCUMENTAIRE.

[26] McPherson, M., Smith-Lovin, L., and Cook, J. (2001). BIRDS OF A FEATHER : Homophily in Social Networks. Annual review of sociology.

[27] Melville, P., Sindhwani, V., and Lawrence, R. (2009). Social media analytics: Channeling the power of the blogosphere for marketing insight. Proc. of the WIN, pages 2–6.

[28] Owoputi, O., Dyer, C., Gimpel, K., Schneider, N., and Smith, N. A. (2013). Improved part-of-speech tagging for online conversational text with word clusters. In In Proceedings of NAACL.

[29] Page, L., Brin, S., Motwani, R., and Winograd, T. (1999). The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab. Previous number = SIDL-WP-1999-0120.

[30] Rauber, A., Merkl, D., and Dittenbach, M. (2002). The growing hierarchical self-organizing map: exploratory analysis of high-dimensional data. Neural Networks, IEEE Transactions on, 13(6):1331–1341.

[31] Salton, G. and McGill, M. J. (1983). Introduction to modern information retrieval.

[32] Samarjeet Borah, A. C. (2013). Intrusion detection system using self organizing map (som): A review. SCIENCE PARK, 1(2).

[33] Shah-Hosseini, H. and Safabakhsh, R. (2003). Tasom: a new time adaptive self-organizing map. Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on, 33(2):271–282.

[34] Tao, K., Abel, F., Hauff, C., and Houben, G.-J. (2012). What makes a tweet relevant for a topic? In Rowe, M., Stankovic, M., and Dadzie, A.-S., editors, #MSM, volume 838 of CEUR Workshop Proceedings, pages 49–56. CEUR-WS.org.

[35] Tobler, W. R. (1970). A computer movie simulating urban growth in the Detroit Region. Economic Geography, 46:234–240.

[36] Weng, J., Lim, E.-P., Jiang, J., and He, Q. (2010). Twitterrank: Finding topic-sensitive influential twitterers. In Proceedings of the Third ACM International Conference on Web Search and Data Mining, WSDM '10, pages 261–270, New York, NY, USA. ACM.

[37] Witten, I. H. and Frank, E. (2005). Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann.

A

Appendix A

B

Appendix A