Universität Passau
Fakultät für Informatik und Mathematik

Masterarbeit

Semantic enrichment and data filtering in social networks for subject centered collection.

Masterarbeit zur Erlangung des akademischen Grades
Master of Science (M.Sc.)

am Lehrstuhl für Verteilte Informations- und Multimedia-Systeme
der Fakultät für Informatik und Mathematik
der Universität Passau

Name:	Anthony FARAUT
Matrikelnummer:	76205
Fachbereich:	Informatik
Studiengang:	Master Informatik
Schwerpunkt:	Informations- und Kommunikationssysteme
Studienjahrgang:	2015-2016
Erstprüfer:	Prof. Dr. Michael GRANITZER
Zweitprüfer:	Dr. Habil. Elöd EGYED-ZSIGMOND

Contents

List of Figures **3**

List of Tables **5**

1 Introduction **9**

1.1 Motivation 9

1.2 Research questions 10

1.3 Organization of the document 11

2 Overall idea **12**

2.1 Problem statement 13

3 Understanding the data **16**

3.1 The corpus collected 16

3.2 Sample of tweets 16

3.2.1 Sample of tweets about "Fête des lumières" 17

3.2.2 Sample of tweets about other subjects 17

3.3 Some facts about the data collected 18

3.3.1 Handmade clustering 18

4 Approach **23**

4.1 Data collection and data storage 23

4.1.1 Sources 23

4.1.2 Tools 24

4.1.3 Social network collectors 24

4.2 Data loading 26

4.3 Data pre-processing 27

4.3.1 Removing stage 28

4.3.2 Cleansing stage 29

4.3.3 Enrichment stage 30

4.3.4 Example of pre-processing on tweets 31

4.4 Data processing 32

4.4.1 Word2Vec 32

4.4.2 Word2Vec vectors . . . 34
4.4.3 Doc2vec vectors . . . 35
4.4.4 TF-IDF vectors . . . 35
4.5 Data clustering . . . 36
4.6 Data extraction . . . 38
4.7 Data visualization . . . 39

5 Experimentation 42
5.1 Knime . . . 42
5.2 Support vector machine . . . 43
5.3 Backtracking process . . . 44

6 Evaluation and validation 45
6.1 Evaluation measures . . . 45
6.1.1 Precision, Recall, F1 . . . 45
6.1.2 Rand index . . . 47
6.1.3 Normalized Mutual Information (NMI) . . . 47
6.2 Models generation . . . 48
6.3 Validation methods . . . 49
6.3.1 Classification - Support Vector Machine . . . 49
6.3.2 Clustering . . . 51

7 Results 52
7.1 Classification - Support Vector Machine . . . 53
7.1.1 Classification on 700 tweets . . . 53
7.1.2 Classification on 7000 tweets . . . 55
7.1.3 Classification on 31000 tweets . . . 58
7.1.4 Discussion about the classification of the whole corpus . . . 60
7.2 Clustering - Kmeans . . . 62
7.2.1 Clustering on 700 tweets . . . 62
7.2.2 Clustering on 7000 tweets . . . 64
7.2.3 Clustering on 31000 tweets . . . 67
7.2.4 Discussion about the clustering of the whole corpus . . . 69

8 Perspectives 72

9 Conclusion 74

10 Acknowledgements 76

11 Bibliography 77

List of Figures

2.1 Overall view of the entire project. 12
2.2 Overall idea of the entire project. 14
2.3 Illustration of the "Fête des lumières". 15

3.1 Curves of corpus per time window of 4 hours. 21
3.2 Curves of the most active time period per time window of 30 minutes. 22

4.1 Data collection architecture. 25
4.2 API architecture. 26
4.3 Collectors inputs. 26
4.4 Data loading architecture. 27
4.5 Country and capital vectors projected into a vector space (Source [24]). 33
4.6 Word2Vec + TF-IDF weighting. 34
4.7 Word2Vec + TF-IDF weighting formula. 34
4.8 Example of the Word2Vec + TF-IDF weighting formula. 35
4.9 Exemple of a Word2vec vector. 35
4.10 Exemple of a Doc2vec vector. 35
4.11 TF-IDF formula. 36
4.12 Exemple of a TF-IDF vector. 36
4.13 Data extraction from clustering. 38
4.14 Movement of the users who posted a tweet in the neighborhood in red. 40
4.15 Movement of the users who posted a tweet during our collect period. 40
4.16 Positioning all tweets in France. 40
4.17 Positioning "Fête des lumières" tweets in France. 41
4.18 Positioning all tweets with a zoom in Lyon. 41
4.19 Positioning "Fête des lumières" tweets with a zoom in Lyon. . . 41

5.1 SVM partitioning. 43
5.2 Backtracking architecture. 44

6.1 Twitter and Instagram logos. 45
6.2 Ways of being right or wrong. 46
6.3 Precision formula. 46
6.4 Recall formula. 46
6.5 F1 formula. 47
6.6 Rand index formula. 47
6.7 Normalized Mutual Information formula. 48
6.8 Mutual information formula. 48
6.9 Entropy formula. 48
6.10 SVM process 50
6.11 Example of a problem of discrimination in two classes. The problem is linearly separable. 51

7.1 Curves of the best and the worst representations with the classification according to the F1 measure. 60
7.2 Curves of the best representations with the clustering according to the F1 measure. 69

List of Tables

3.1 Languages present in tweets 18
3.2 Some of the more interesting clusters from the handmade clustering 19
3.3 More important languages present in tweets talking about the event 20
3.4 Table of corpus per time window of 4 hours. 21
3.5 Table of the most active time period per time window of 30 minutes. 22

4.1 Example of top 15 hashtags and words after the extraction step. 39

6.1 Doc2vec parameters which varied. 49

7.1 Results of the classification on 700 tweets. 53
7.2 Confusion matrix on 700 tweets. 54
7.3 Top 15 hashtags extraction from the classification on 700 tweets. 54
7.4 Results of the classification on 7000 tweets. 55
7.5 Confusion matrix on 7000 tweets. 56
7.6 Top 15 hashtags extraction from the classification on 7000 tweets. 56
7.7 Results of the classification on 31000 tweets. 58
7.8 Confusion matrix on 31000 tweets. 58
7.9 Top 15 hashtags extraction from the classification on 31000 tweets. 59
7.10 Table of the best and the worst representations with the classification according to the F1 measure. 60
7.11 Results of the clustering on 700 tweets. 62
7.12 Confusion matrix on 700 tweets. 63
7.13 Top 10 hashtags extraction from the clustering on 700 tweets . . 63
7.14 Results of the clustering on 7000 tweets. 65
7.15 Confusion matrix on 7000 tweets. 65
7.16 Top 10 hashtags extraction from the clustering on 7000 tweets. . 65
7.17 Results of the clustering on 31000 tweets. 67
7.18 Confusion matrix on 31000 tweets. 67
7.19 Top 10 hashtags extraction from the clustering on 31000 tweets. 68

7.20 Table of the best representations with the clustering according to the F1 measure. 69

Abstract

In recent years, social networks have become an important source of information, connecting people all around the world in almost real-time. Accordingly, demands for extracting meaningful and interesting information from them have dramatically increased. A lot of research has turned towards analyzing the social media content for tracking real time events, for example politics, sports, natural disasters, group or community discovery. Social networks can be queried through their API (Application programming interface) with a finite set of keywords and they return the posts containing exactly the searched terms. However, social media remain large, noisy and constantly evolving.

Heterogeneity is particularly a problem, as users are free to post messages in the form they want. Indeed, there are no grammatical or spelling rules to follow on social networks. In this context, the problem is to be able to group messages talking about similar subjects even though they use different words or representations. It is also important to keep the real time aspect of this research in mind. The main topic of this work is to follow real events in real time through social networks by improving an initial query with #hashtags discovered in the clusters made by the process. It is a technique that has been chosen to use among others to follow a subject. Here, the goal is to answer the question "how to group messages from social network correctly in real time" by trying several approaches that will be presented.

In this research, the focus will be made on several parts of the project. First of all, a phase of pre-processing that will reduce the heterogeneity of data by cleaning, removing and enriching the raw text. Then, a step of processing which will transform messages into a numerical vector structure which allows a clustering. Three representations are used (Word2vec, Doc2vec and TF-IDF). Then, the clustering will cluster the data into groups based on the numerical vector representation of tweets. For grouping messages talking about the same subject together, two methods will be used: a classification with Support vector machine and a clustering with some clustering algorithms as Kmeans, DBScan, Optics and Hierarchical.

Finally, in this work, a state of the art of the main parts of the process and the methods used to identify the discussion flows on social networks will be presented. The focus was defined mainly on Twitter about real events as the "Fête des lumières" in Lyon. A model for the query enrichment for dynamic monitoring of an event over an extended period with keywords that evolve over time will be presented. This project is part of a larger project entitled IDENUM [8].

Keywords: Social networks, Data science, Machine learning, Neural networks (Word2vec, Doc2vec), Support Vector Machine (SVM), Classification, Clustering

1 Introduction

1.1 Motivation

Nowadays, a huge amount of data is being generated by social media networks in real time. Accordingly, demands for extracting meaningful and interesting information from them have dramatically increased. We can see that a lot of research has turned towards analyzing the social media content for tracking real time events, for example politics, sports, natural disasters, group or community discovery. However, it is difficult to apply data mining techniques directly without an automated data gathering and filtering system because of the main characteristics of social media: the data is large, noisy and constantly evolving.

The starting point of this project is based on the API of social networks. Indeed, the only way to easily communicate with social networks is to go through their API. The API returns documents containing some data, as the message posted on the social network, the date, the user and also the initially inputs as keywords or the geographic location position. So the whole architecture of the process have been created by taking into account these constraints.

Besides, it's often difficult to identify correctly the subject of a social network post because users write them using unstructured language with many typographical errors. A lot of research attempt to overcome this issue in finding structured data in the post's text in order to provide additional context. They use, for example, query expansion and relationships defined on the semantic web like in [34].

Meanwhile enriching the set of words with other words directly related with the topic, arises several problems. How to know if the enrichment is correct that is to say, it has no noise and is therefore directly related to the subject? The process have to filter all the new information in order to keep all those which are desired. Furthermore, mobile devices used by most of the population allow them to publish data in real time. The real time provides an additional problem. Indeed, a word can be pertinent during a period of time and be stale

two hours after. So, the process has to be dynamic and keep a set of relevant words spatially and temporally. The main goal is to enrich the initial request with relevant keywords in order to extend the search field.

1.2 Research questions

In this work, several research topics are tackled as textual data cleaning, textual data enrichment, textual data representation and textual data clustering.

Social networks can be queried through their API (Application programming interface) mainly with a finite set of keywords and they return the posts containing exactly the searched terms. However, as keywords quickly change over time, it is difficult to maintain an effective querying throughout the follow-up event. The process implemented in this project is trying to solve this problem by trying to keep a bag of words that evolves over time. The evolution of the bag of words will reduce the noise it could exist with a constant bag of words. All the following research topics try to improve this step of keeping a bag of relevant words over the time.

In this project, a phase of data pre-processing is presented in order to reduce the heterogeneity in the corpus as presented before. Moreover, like most messages on social networks are hollow and has only a short context, an enrichment phase is also proposed based on the entire corpus. As data clustering is not possible on textual data, a numerical representation of the textual data is necessary. For this part, three representations of the textual data are presented. First of all two representations based on neural networks (Word2vec and Doc2vec) and a representation based on the statistic context (weighting method) which is TF-IDF. The three representations are numerical vectors based. The goal is to find which is the best textual data representation for posts from social networks.

Finally, as the main goal of the entire process is to cluster the data into groups talking about the same topic, it is interesting to look at all the algorithms that exist to meet these needs. For this step, two comparisons will be made. A classification with support vector machine technique and with a clustering algorithm. The classification and clustering do not act the same way. It will be interesting to see which one performs the best on messages from social networks.

1.3 Organization of the document

The document is organized as follows. First of all, the overall idea of the entire project (chapter 2) is presented in order to help the reader to have a global view of the project. Then, as understanding the data was a very important step in the project, some elements about the corpus made will be presented. The corpus, some examples of messages from social network and some facts about the data will be presented. It is really interesting to know the corpus on which the project will be realized. Furthermore, the entire approach will be presented.

As the "data collection phase" is the first stage of the process, it will be discussed in the section 4.1. Moreover, the collectors made for the project will be presented. Following this step of Data collection, a brief phase of Data loading (section 4.2) will be presented. The raw data cannot be used without any transformation, so the following stage will be introduced: "Data pre-processing" (section 4.3). This stage will explain methods in order to clean, remove some issues in the raw data and enrich the entire corpus. After the combination of those stages, the data need to be converted in a way to facilitate the rest of the work. Here comes the Data processing stage (section 4.4), which will transform the raw text data into numerical vectors. At this moment, everything is ready to begin the clustering step, with the "Data clustering" (section 4.5) stage. This step will allow to get all the discussion flows. Finally in this last step "Data extraction" (section 4.6), the process will select discussion flows (clusters) related to the subject, and get the X top tokens (words), #hashtags and re-inject them in the beforehand query. Throughout the state of the art, social networks related problems will be discussed such as the fact that the data is large, noisy, constantly evolving and unstructured. After the presentation of the approach, some data visualization on the data will be presented.

Finally, the experiments made and the evaluation with all the measures will be presented. It's important to know beforehand how to validate the approach. Then, the results will be presented with perspectives view in order to try to improve the results.

2 Overall idea

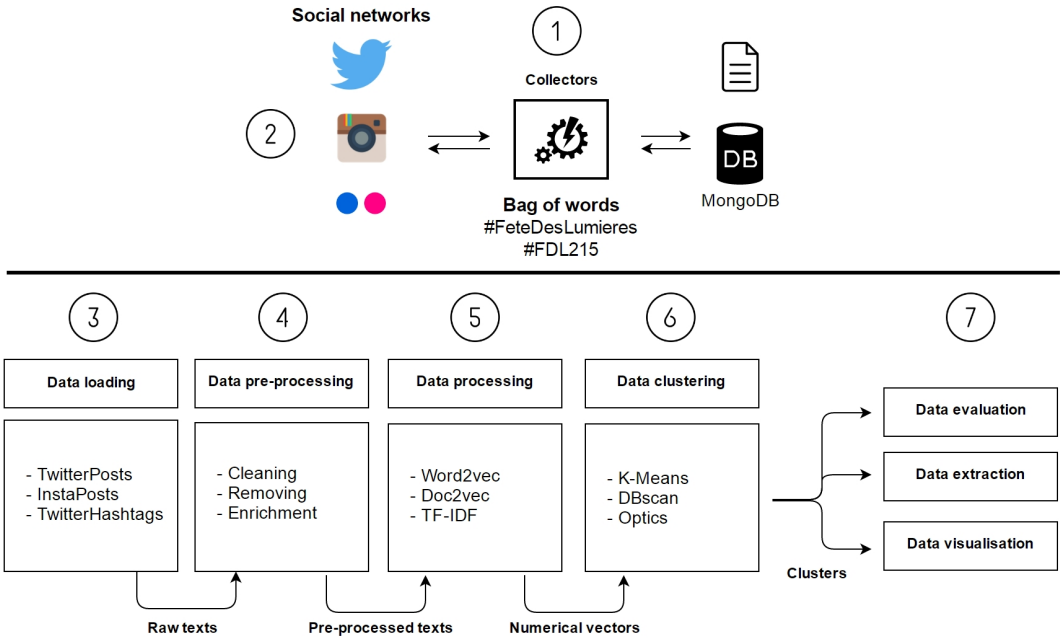


Figure 2.1: Overall view of the entire project.

This section will present an overall idea of the entire project in order to ensure a broad understanding before going into details. There are several steps in the process that should be kept in mind throughout the reading of this master thesis. Most of these steps are presented in the Figure 2.1.

It is important to understand that there are two distinct parts in this project. Indeed, the top part of the Figure 2.1 as well as the bottom part. The top part is represented by the numbers (1 and 2), and the bottom one is represented by the numbers (3, 4, 5, 6 and 7). It is why there is a black line between them.

Firt of all, as it will be presented in the subsection 4.1.3, collectors have been developed in order to collect the data from the social networks using their API (Application programming interface (communication interface)) (number 2 on the picture 2.1). These collectors (number 1 on the picture 2.1) are daemons and work independently of the entire process. They are the first stage of the process, in order to get data to work with in the following steps. This step has been developed because it's more interesting for the process to collect what is

needed and because it will be re-used for other projects.

As explained before, the rest of the process have been developed in a layer based architecture in order to get a clear separation between layers. As you can see on the right side of the figure 2.1, there are 4 main layers as "Data loading" (number 3 on the picture 2.1), "Data pre-processing" (number 4 on the picture 2.1), "Data processing" (number 5 on the picture 2.1) and "Data clustering" (number 6 on the picture 2.1).

The data loading layer allows to make an abstraction for the database which is hidden and to get the data easily just by calling a java method. Then the data pre-processing step is present in order to make some modification on the raw texts. These modifications are important to improve the following stages. There are three main subsections in the data pre-processing step as cleaning (subsection 4.3.2), removing (subsection 4.3.1) and enrichment (subsection 4.3.3) which it will be explained in the section 4.3. Concerning the data processing step, the main goal is to change the representation of the messages form social network in a comprehensible format for the following stage. This step makes numerical vectors from the texts. Indeed, it will be easier for the clustering stage to have some numerical representation of texts. Finally, there is a data clustering step, as indicated by its name, to cluster the data.

As the main goal of the project is to enrich the beforehand set of words, and not just cluster the data, there is a data extraction step (number 7 on the picture 2.1). This stage allows to extract the X top words and or #hashtags of each (selected) cluster, in order to enrich the initial set of words. It's important to understand that here, a choice between clusters has to be done. The process will give some clusters from the clustering step and it has to keep the ones which are related to the event followed. And then, from these selected clusters, get the X top words and or #hashtags.

The data evaluation and data visualization stages are not necessary for the process but are there to test and evaluation all the process with some measures (as explained in the section 6.1), and to visualize some results.

2.1 Problem statement

Messages posted on social networks (such as Twitter, Instagram, etc.) reflect user interaction on real events taking place in the world in real time. Topics discussed on social networks are various like elections, sports, cultural events,

natural disasters, etc. The nature of these events have a direct impact on the amount of messages posted on social networks.

Monitoring these events is a bold challenge for researchers, first because a subject is characterized by several terms (these terms may be hashtags) that can change dynamically over time. Some may disappear over time and others may occur.

So it is worth asking how is it possible to cover all these terms used during the analysis process. This is one of the objectives in this work. However, before seeking the evolution of these terms, the need is to identify sets of tweets that talk about the same subject and representing a discussion flow, which defines the main objective of the work.

To summarize, the main goal of this master thesis is to “Collect the most information on an event described beforehand as a set of words while being robust (i.e. eliminating noise) in real time." A practical example is to begin with a bag of words set according to the subject (number 1 on the picture 2.2). Retrieve data from social networks (number 2 on the picture 2.2) before the date of the event, pre-process the data (number 3 on the picture 2.2) in order to clean everything wrong (that is to say, all the mistakes made by the users: the misspellings, the grammatical alterations such as incomplete sentences and word distortions etc.). Then process the data (number 4 on the picture 2.2) for a numerical representation of texts, cluster the data (number 5 on the picture 2.2) in order to regroup posts whose speak about the same topic in the same cluster. Finally get the X top tokens (words and or #hashtags) from each cluster in order (number 6 on the picture 2.2) to re-inject those words in the beforehand bag of word (number 7 on the picture 2.2) and relaunch the query (enriched).

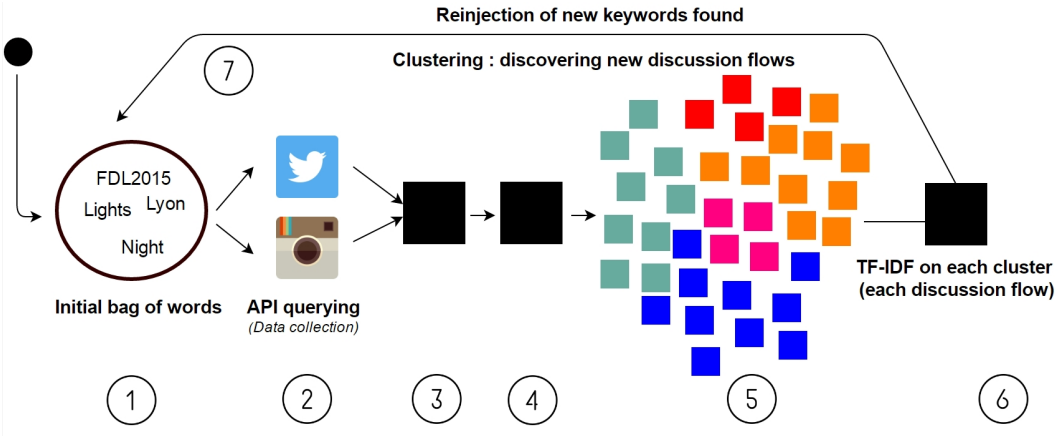


Figure 2.2: Overall idea of the entire project.

This whole process seems simple, however the amount of data published online every day on social networks makes the task difficult. For example, in Twitter there are 313 millions monthly active users, 1 billion unique visits monthly to sites with embedded Tweets, 82% active users on mobile, over 40 supported languages. These data were retrieved from [1] and all numbers are approximate as of June 30, 2016.

Furthermore, the process must be able to process these data in a continuous way. The processing difficulty is due to several reasons. First, the data is large, noisy and constantly evolving. In addition, for example Twitter, there is a major constraint in the number of character which involves fragmented and ambiguous messages. There is also the lack of context words (#hashtags) in some messages which is a rich source of information.

For the experiments, the “Fête des lumières 2015” in Lyon has been chosen as the event to focus on.

Here is an explanation of the Festival of Lights (in french: "Fête des lumières") in Lyon from Wikipedia. "France expresses gratitude toward Mary, mother of Jesus on December 6 to December 9 of each year. This uniquely 'Lyonnaise' (from Lyon) tradition dictates that every house place candles along the outsides of all the windows to produce a spectacular effect throughout the streets. The festival includes other activities based on light and usually lasts four days, with the peak of activity occurring on the 8th."

Figure 2.3: Illustration of the "Fête des lumières".

3 Understanding the data

Before beginning any research on the "Masterarbeit" it was important to understand the content returned by the API of social networks. In fact, the need was to understand what was available in the Json in order to work with. As it will be presented in the following subsection 4.1.3, some specific collectors which communicate with the APIs were developed.

3.1 The corpus collected

Thanks to the collectors developed, several corpus for several real events were generated. The focus was on the "Festival of Lights 2015", which unfortunately has not occurred, but has allowed to recover a lot of messages from Twitter and Instagram.

As it will be explained later, the collectors developed take as input a list of keywords to search for and / or a rectangle coordinates, to target the research. In this corpus, the initial request was with these following keywords "Lyon", "FeteDesLumieres2015", "FDL2015", "candle" and the geographical coordinates of "le grand Lyon" which is : *the Metropolis of Lyon, also known as Grand Lyon (i.e. "Greater Lyon"), is a French territorial collectivity with special status located in the central Eastern region of "Auvergne-Rhône-Alpes", encompassing the city of Lyon and most of its suburbs* (Wikipedia definition[1]).

The collection was done in a continuous manner over several days (with the twitter collector in real time) and the data was injected directly into a database (details will be explained in the subsection 4.1.3).

3.2 Sample of tweets

The content of the tweets collected through the developed collectors are dense and various. As explained before, the data is large, noisy and constantly evolving, so in the following subsections some tweets will be presented in order

[1] https://en.wikipedia.org/wiki/Metropolis_of_Lyon

to see that there is no pattern, no rules, everybody wrote what he wanted in his specific manner. We can see in the data, tweets containing only hashtags, only user mentions, some links and some non existing words as "RT" which means "retweet" [2].

3.2.1 Sample of tweets about "Fête des lumières"

- #Lyon #8decembre #hommageauxvictimes https://t.co/eWFVmChqU8
- I'm at Place de la République in Lyon https://t.co/kbYxy0p8Pk
- #8decembre aux lyonnais #FeteDesLumieres https://t.co/ojgYDNkq8n
- À Lyon, ils ont osé mettre un # à côté de "Merci Marie". Avec un peu de chance elle va RT ! #FeteDesLumieres #Lyon https://t.co/wlXqDSrMNt
- 8 décembre traditionnel à #Lyon : les lumignons sont aux fenêtres. https://t.co/ZiuMzHirko
- Petit tour en ville comme chaque année pour ce 8 décembre particulier. Place des Jacobins #FdL2015 https://t.co/WxFP7xFi9S
- #Lyon #FeteDesLumieres une fête particulière cette année #Paris https://t.co/oyJgvZpBqb
- Fêtes des Lumières #8decembre #lyon #parisattacks #werenotafraid #forabetterworld #PrayForParis... https://t.co/t0tLug7XhM

3.2.2 Sample of tweets about other subjects

- @mgrgiraud @ldevilloutreys C'est la génération #2013 #LMPT
- #FF @berniezinck for a good music
- @aegyottae everyday i listen this song (every morning not exception) https://t.co/FasYu9FI1B #np @BTS_twt
- @BroleoJ mañana voy a tu país..! A ver qué tal es eso por allá en el norte...
- Yo les twittos bien ou bien la famille ?
- Sie sind #endlich wieder da ! ???? @phillaude @derTC @oguz @Y_Titty @PatrickBuenning
- Winner Disrupt London 2015 Jukedeck https://t.co/omYaFCcjHH et aussi par @leweb en 2014 :) https://t.co/hGIAFJMakv
- #XF9 Cesare ti voglio bene !!!!! Mio cantante preferito veramente. #skyuno

[2] https://support.twitter.com/articles/77606

3.3 Some facts about the data collected

The corpus of the "Fête des lumières" was built from December 7, 2015 at 01pm until December 11, 2015 at 09am. The corpus has a size of 90.3 MB (uncompressed json without the media). It contains:

- 31 006 tweets;
- 1559 tweets with a specific geolocation;
- 3796 tweets with at least one media (photo/video);
- 83 tweets with at least one video;
- 126 tweets with at least one gif;
- 3587 tweets with at least one photo;
- 4136 tweets with at least one link;
- 6354 tweets with at least one #hastags;
- 15838 tweets with at least one user mention.

In the corpus, 38 languages are represented. According to the subject (which takes place in France), 21925 tweets are in French. For the rest of the languages, we can see (for the more commons) :

Table 3.1: Languages present in tweets

	Language	Number of tweets in this language
Language 1	French	21925
Language 2	English	4479
Language 3	Undefined	2284
Language 4	Spanish	611
Language 5	Arabic	293
Language 6	Indonesian	194
Language 7	Turkish	184
Language 8	Portuguese	130
Language 9	Italian	113
Language 10	German	66

3.3.1 Handmade clustering

A handmade clustering have been made by Mrs. Oriane PIQUER-LOUIS (who is a PhD student in communication science, working on the IDENUM project[1]), on the entire corpus, that is to say on 31000 tweets. This is a binary clustering, with "Fête des lumières" cluster and "NOT Fête des lumières". A more precise clustering have been also made by Mrs. PIQUER-LOUIS on 700 tweets representing one hour collection from December 08, 2015 at 07pm until December 08, 2015 at 08pm. From this clustering, there are 43 clusters, with one cluster related to the main topic (which is the “Fête des lumières 2015”).

In this cluster there are 122 tweets for the event. This clustering has been made in order to see the dispersion of tweets during one hour. Furthermore, this small corpus has been selected during this period because it was during a peak (as presented in the Table 3.4).

Table 3.2: Some of the more interesting clusters from the handmade clustering

	Subject of the cluster	Nb elem in the cluster
Cluster 1	Unclassifiable	283
Cluster 2	Fête des Lumières	122
Cluster 3	#TPMP (a French live TV show)	33
Cluster 4	Politics	24
Cluster 5	#cavous (a TV show)	21
Cluster 6	Student life	21
Cluster 7	Music	16
Cluster 8	@Daouda95100, who talks to himself	15
Cluster 9	Food	14
Cluster 10	#YR	14
Cluster 11	iPhone	12
Cluster 12	Spanish/politics	12
Cluster 13	Sex	10
Cluster 14	OL (football club based in Lyon)	8
Cluster 15	#SS9 (a French reality show)	7
Cluster 16	Snapchat failure	7
Cluster 17	FN (a French political party)	6
Cluster 18	TheVamps	6
Cluster 19	Foot	5
Cluster 21	#LPLDA3 (a French reality show)	4
Cluster 22	Cinema	4
Cluster 23	Selfies	4
Cluster 24	Fashion	4
Cluster 26	Cécifoot	4
Cluster 28	Louane (a French singer)	3
Cluster 29	Code	3
Cluster 30	Harry Potter	3
Cluster 31	BTS	2
Cluster 33	Trends	2
Cluster 34	#lgj (a French live TV show)	2
Cluster 36	Religion	2
Cluster 37	#COP21	2
Cluster 38	Trump	2

In the entire corpus, there are 1048 tweets talking about the "Fête des lumières" (this is not a lot) and 29960 tweets which don't talk about the previous cited event.

Concerning the cluster talking about the "Fête des lumières" which contains 1048, there are :

- 423 tweets with at least one media (photo/video);
- 4 tweets with at least one gif;
- 15 tweets with at least one video;
- 404 tweets with at least one photo;
- 150 tweets with at least one user mention.
- 462 tweets with at least one link;
- 435 tweets with a specific geolocation;
- 561 tweets with at least one #hastags;

Table 3.3: More important languages present in tweets talking about the event

	Language	Number of tweets in this language
Language 1	French	771
Language 2	English	122
Language 3	Undefined	118
Language 4	Spanish	13
Language 5	Japanese	5
Language 6	Norwegian	2
Language 7	Russian	2

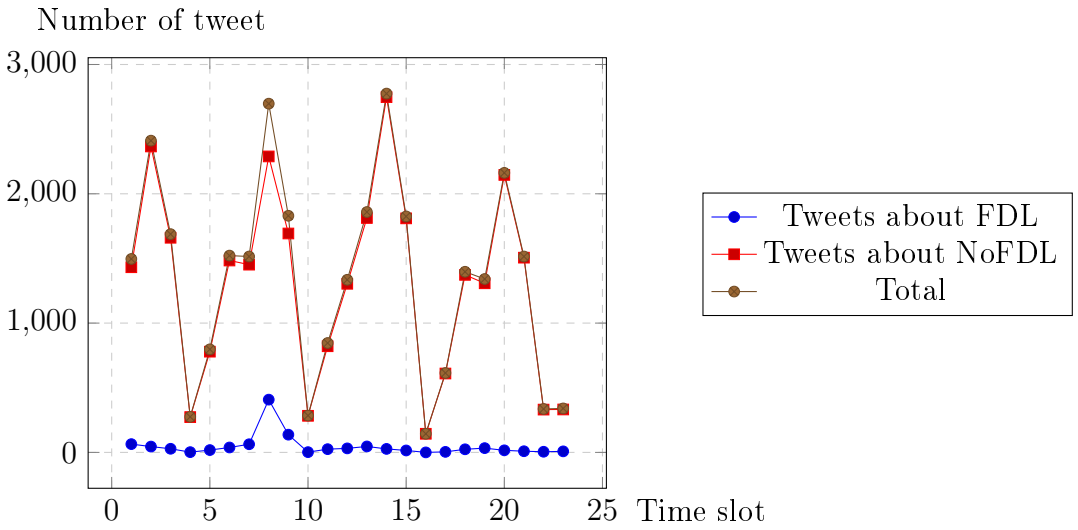
According to the table 3.3, we can suppose that the language is particularly discriminant in this kind of event which takes place in France.

Finally, it is interesting to see the distribution of tweets talking about the "Fête des lumières" over the time. In the Figure 3.1, the progression of tweets during the days is shown. Indeed, for example, for the periods 4, 10, 16, 22, (Table 3.4), which is during the night, there are not many message. Otherwise, for the periods 2, 8, 13, 14, 19, 20, which is during the afternoon or the evening/night, there are more messages. This shows that the message publication on social networks follows more or less the period of day and night. It is likely to be different for an event in New York (for example), since it is a city that never sleeps.

It is also interesting to see that in the Figure 3.2, there are two message peaks between 6:00pm and 6:30pm and between 9:00pm and 9:30pm December 8th. However there is only one peak for the "Fête des lumières" tweets between 6:00pm and 9:30pm. The curves presented come from the both tables (Table 3.4, Table 3.5).

Step	Dates	FDL	NoFDL	Percent	Total
1	07 dec 13:00 - 07 dec 17:00	64	1,432	4.5%	1,496
2	07 dec 17:00 - 07 dec 21:00	45	2,366	1.9%	2,411
3	07 dec 21:00 - 08 dec 01:00	28	1,660	1.7%	1,688
4	08 dec 01:00 - 08 dec 05:00	2	273	0.7%	275
5	08 dec 05:00 - 08 dec 09:00	18	779	2.3%	797
6	08 dec 09:00 - 08 dec 13:00	38	1,484	2.6%	1,522
7	08 dec 13:00 - 08 dec 17:00	63	1,452	4.3%	1,515
8	08 dec 17:00 - 08 dec 21:00	408	2,289	17.8%	2,697
9	08 dec 21:00 - 09 dec 01:00	137	1,693	8.1%	1,830
10	09 dec 01:00 - 09 dec 05:00	2	283	0.7%	285
11	09 dec 05:00 - 09 dec 09:00	25	821	3.0%	846
12	09 dec 09:00 - 09 dec 13:00	31	1,304	2.4%	1,335
13	09 dec 13:00 - 09 dec 17:00	46	1,813	2.5%	1,859
14	09 dec 17:00 - 09 dec 21:00	27	2,748	1.0%	2,775
15	09 dec 21:00 - 10 dec 01:00	15	1,810	0.8%	1,825
16	10 dec 01:00 - 10 dec 05:00	0	144	0%	144
17	10 dec 05:00 - 10 dec 09:00	4	610	0.7%	614
18	10 dec 09:00 - 10 dec 13:00	24	1,373	1.7%	1,397
19	10 dec 13:00 - 10 dec 17:00	33	1,308	2.5%	1,341
20	10 dec 17:00 - 10 dec 21:00	16	2,146	0.7%	2,162
21	10 dec 21:00 - 11 dec 01:00	9	1,506	0.6%	1,515
22	11 dec 01:00 - 11 dec 05:00	5	331	1.5%	336
23	11 dec 05:00 - 11 dec 09:00	8	332	2.4%	340

Table 3.4: Table of corpus per time window of 4 hours.

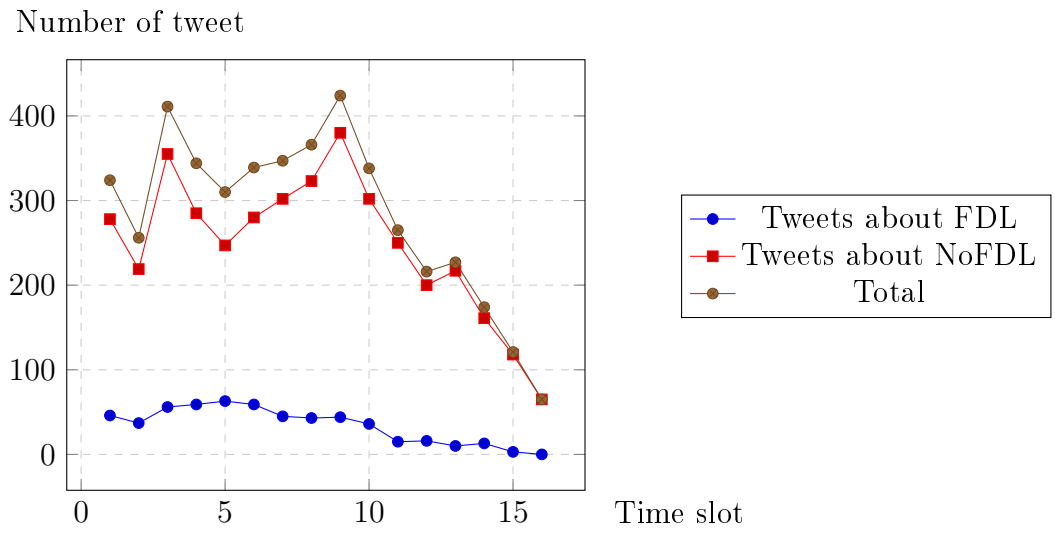


The chart shows therefore the number of tweets per time slots of 4 hours for the entire corpus.

Figure 3.1: Curves of corpus per time window of 4 hours.

Step	Dates	FDL	NoFDL	Percent	Total
1	08 dec 17:00 - 08 dec 17:30	46	278	16.5%	324
2	08 dec 17:30 - 08 dec 18:00	37	219	16.9%	256
3	08 dec 18:00 - 08 dec 18:30	56	355	15.8%	411
4	08 dec 18:30 - 08 dec 19:00	59	285	20.7%	344
5	08 dec 19:00 - 08 dec 19:30	63	247	25.5%	310
6	08 dec 19:30 - 08 dec 20:00	59	280	21.1%	339
7	08 dec 20:00 - 08 dec 20:30	45	302	14.9%	347
8	08 dec 20:30 - 08 dec 21:00	43	323	13.3%	366
9	08 dec 21:00 - 08 dec 21:30	44	380	11.6%	424
10	08 dec 21:30 - 08 dec 22:00	36	302	11.9%	338
11	08 dec 22:00 - 08 dec 22:30	15	250	6.0%	265
12	08 dec 22:30 - 08 dec 23:00	16	200	8.0%	216
13	08 dec 23:00 - 08 dec 23:30	10	217	4.6%	227
14	08 dec 23:30 - 09 dec 00:00	13	161	8.1%	174
15	09 dec 00:00 - 09 dec 00:30	3	118	2.5%	121
16	09 dec 00:30 - 09 dec 01:00	0	65	0%	65

Table 3.5: Table of the most active time period per time window of 30 minutes.



The chart shows therefore the number of tweets per time slots of 30 minutes during 8 hours.

Figure 3.2: Curves of the most active time period per time window of 30 minutes.

4 Approach

In this section, the models made to respond to the problem, will be presented. It is divided into several distinguishable layers with a brief state of the art and the development made for each step. Most of the stages have been inspired from the papers read and the others have been created to meet the needs of the project. Some layers are essential for the process as the "data loading" (section 4.2), the "data processing" (section 4.4) and the "data clustering" (section 4.5). The layer "data pre-processing" (section 4.3) is not. However, this layer is still necessary to improve results. Indeed, as explained in the section 4.3, this layers improve the quality of the corpus by making some modification on it.

4.1 Data collection and data storage

The first step which has to be carried out concerning the data search on the social network events follow-up is to retrieve the data. Without data the process cannot do anything. A lot of social media services are available on the web and the person who wants to work with have to learn to access them in order to retrieve all the information he needs. Most of the social networks provide APIs (Application Programming Interface) in order to access data (usually user posts). In the following sections, the sources and the tools present in the networks and the literature will be presented. All information which will be presented come from the online API documentations.

4.1.1 Sources

Nowadays, there are a lot of social networks which provide to share a lot of things through different medias as text, pictures and sometimes videos. Some of them are real-time which means that there are a lot of users that publish on it in real time.

In Twitter [19], users can post a tweet (which is the name of the text publication on this social network) with text (limited to 140 characters), photos and/or videos. Furthermore, they can localize their posts in order to prove

their location in real-time and so on like user-mention, #Hashtags and RT (Retweet) in order to answer to or complete a selected tweet.

In Instagram [9], users can post photos or videos with a text describing the media, with user-mention, #Hashtags as in Twitter. Instagram is very famous, and maybe the first one in the share of media as photo and video in real-time.

There are a lot of other famous social networks which are not really real-time. More precisely, there are real-time but the use made of these social networks by users, makes these social networks not real-time. For example, there are Pinterest [17] or Flickr [6], which are famous but not real-time. Indeed, Flickr and Pinterest are used by computer users and not by smart-phone users generally and serve as an archive, more than real time information exchange.

The most popular social network in Europe and more precisely in France is Facebook [5]. However, according to the privacy settings it is difficult to access any posts. In the papers read Facebook is not used. The use of Facebook is more for sharing life events / chatting with friends and not with unknown users as Twitter. So, generally, user accounts in Facebook are totally "closed" (If you're not friend with the person, you cannot see anything).

4.1.2 Tools

As mentioned before, Twitter provides an API in order to communicate with it, and it is available here [20]. However, as the use of the API is quite complex, there are several libraries in order to facilitate the work of the developers. For example, in [29] the authors used Twitter4J API [22]. On the website you can read: "With Twitter4J, you can easily integrate your Java application with the Twitter service." This is exactly what is it for. This allows developers to create applications communicating with the Twitter API, quickly and effortlessly.

Helps are provided on the Twitter website in the developer section, for example, here [21] you can see the JSON format on the Twitter response from the API. In the paper [28] in order to extract real-time knowledge from Social big data using distributed architecture, they use a lot of tools for the data collection stage.

4.1.3 Social network collectors

According to the state of the art made for this section, the project was focused on Twitter and Instagram because these social networks are in real-time

as explained before. Furthermore, data collectors were developed for Twitter and Instagram, they are available online on GitHub [12]. The fact of realizing some data collectors allows to have more flexibility in how to manage the collection process. The choice was made to use the Mongo database [11] in order to store the data collected as the API respond with JSON data, and Mongo database deals well with it. *"Classified as a NoSQL database, MongoDB avoids the traditional table-based relational database structure in favor of JSON-like documents with dynamic schemas (It calls the format BSON), making the integration of data in certain types of applications easier and faster."* (Wikipedia[1])

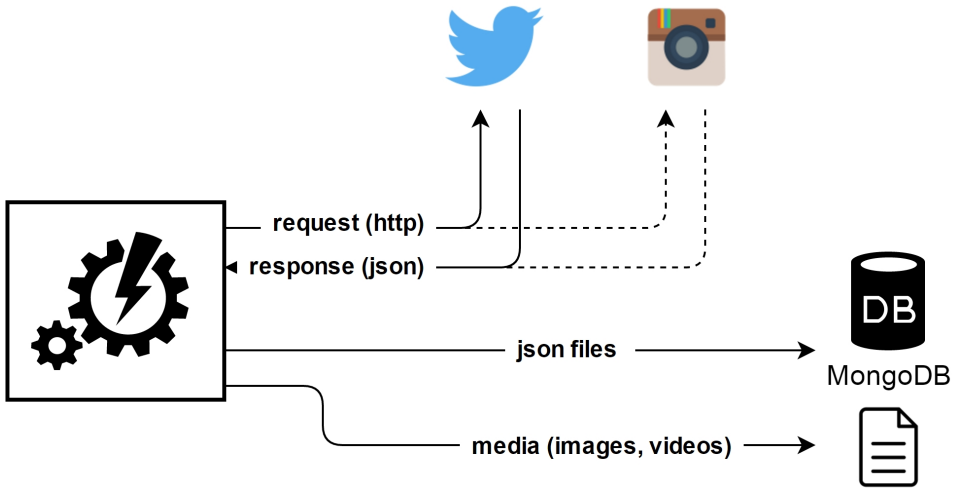


Figure 4.1: Data collection architecture.

For both of the social networks (Twitter and Instagram), 2 collectors were developed. One collector in real time with the Streaming API and one in order to collect data in the "past" from now with the Rest API. For Twitter, the both collectors are in Java, and for Instagram the real time collector is in nodejs and the other one in Java. As explained in the figure 4.2, the collectors are the client (number 1 on the picture 4.2) and communicate with the web services (number 3 on the picture 4.2) which are (Twitter and Instagram) throughout internet (number 2 on the picture 4.2).
The collectors can take different inputs for the requests. Indeed, a point with a radius (number 1 on the picture 4.3), some keywords (number 2 on the picture 4.3) and or a geographical zone with 4 points (number 3 on the picture 4.3) can be given.
There is also the possibility of combining the inputs in order to reduce the search space and to eliminate the noise that there might be without such restriction settings. For example, you can specify 4 #hastags as ("#dult", "#passau", "#dirndl", "#lederhosen") and a specific geographical zone as

[1] https://en.wikipedia.org/wiki/MongoDB

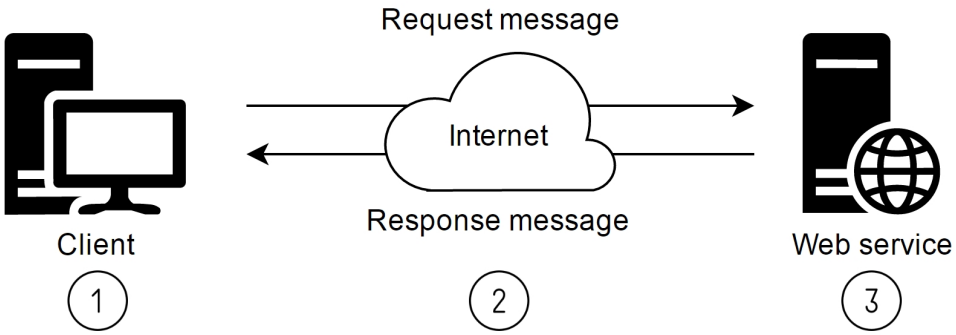


Figure 4.2: API architecture.

Figure 4.3: Collectors inputs.

the "Passauer dult" with 4 geographical points.

The collectors made also allow saving the media, as images and videos. For each message published on Twitter or Instagram, each time there is an available media, it's directly downloaded by the collector. These media are not used in this project, but can be used for upcoming works as explained in the chapter 8. Indeed, the media could be used in order to get external information.

Finally, these collectors work in the background and require no assistance of a human.

4.2 Data loading

Once the data is stored (without any modification by the collectors), the process have to load the data in order to work on it. As explained before, the process have been developed in successive layers in order to have a logic flow. The data loading stage is the first stage of the process because the data collection stage is a "background" daemon. Indeed, as explained in the subsection 4.1.3, the collectors work in the background and require no assistance of a human. The collection phase is launched before the event followed and stopped shortly after the end of the event. This first layer allows to make an abstraction layer for the database. Some methods have been developed as :

- “get the twitter posts from a timestamp to another timestamp”;
- “get the X first following twitter posts from a timestamp”;
- “get all the hashtags present in the entire corpus”;
- “get all the languages present in the entire corpus”;
- “get all the languages present in the "Fête des lumières" cluster”;
- “get all the twitter posts containing media";
- “get all the twitter posts containing at least one hashtag";
- etc...

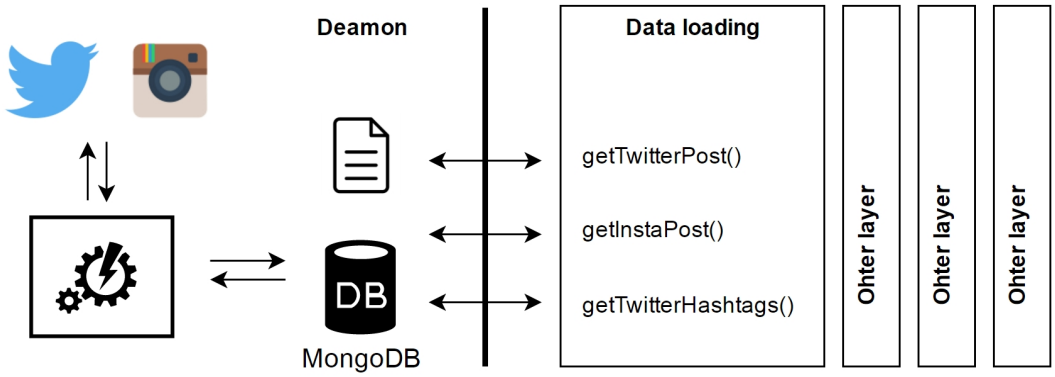


Figure 4.4: Data loading architecture.

4.3 Data pre-processing

Before starting any data mining work on posts from social networks, it is important to prepare the data. In the social networks we can see non-standard terms or symbolic expressions, written by the users. They have the freedom to express their opinions, their moods in words with no syntactic or semantic control. Some social media services such as Twitter limit the length of each message and that challenge the users to say what they want in few words. That results in creativity that would be considered as noise if nothing is made in order to remove the issues. Besides, we can see grammatical alteration such as incomplete sentences and word distortion modifying the original meanings from the real terms (as an example “Looooooooooooooove" in order to mean “much love" or “brb" in order to mean “be right back"). These non-standard terms cause problems for the semantic interpretation of the words or the sentences. In all the papers read, most of them pre-process the data pretty much in the same way.

As explained before, there are so many grammatical alterations in the raw data from social networks, that it is important to clean the data as much as possible in order to ease the work of the following stages. In order to clean the

raw text [37] pre-process the data as follows: The textual information in each tweet is first lowercased, then usernames, stopwords, punctuations, numbers, and URLs are removed.

In the paper [29], they present their method for cleansing data from social networks. For each post, they pre-process the text in order to only keep what they want. They normalize the text as follows: Remove urls, user mentions and hashtags as well as digits and other punctuation. Once done with this stage, they split the text with white spaces and remove stop words. This makes some tokens (one word, one token). In order to prepare the tweet corpus, in each time window, for each post, they first append the user mentions, the hashtags and the resulting clean text tokens.

We can see in the paper [33], that they are less selective. Indeed, they tokenized the text contained on the tweet contents into words using space and punctuation as separators as [29]. For each word in tweets, they applied stemming and stop word elimination and removed tweets with non-alpha numeric characters and ones that indicate location check-ins. Finally, in their tweet cleaning process, they removed URLs and repeating characters that appear more than twice in a word consecutively, for example "gooooal" into "goal".

In the paper [31], they split into three parts the cleansing part. 1. Segments containing well-known slang words, for example "lol". They use a compilation of Internet slangs provided by this online dictionary. 2. Segments containing words with consecutive repeating characters, for example "hahahaha" and "goooooood". The use of these words are frequent in tweets to represent exaggerative emotions. 3. Segments containing hashtags (words with "#" as prefix). Created by users to highlight keywords/topics, they are not considered named entities for the paper, and can be easily removed by locating the "#" prefix with for example regular expressions as well.

For the preprocessing, the stages were divided into three steps that are:

4.3.1 Removing stage

As indicated by its name, this step will lose information (that is not very useful for the project). Indeed, there are a lot of things in the messages from social network that are not useful for the current version of the project. For example, the user mentions are not used by the process because it just refer another user without adding more information about the event followed. Moreover, as explained before, the data is very noisy. Each person writes his message

as he thinks without worrying about grammar or spelling. For example, for the French language which has accents, it is possible to see two messages with the same words but not spelled in the same way. For example, with the words "Fête des lumières", it is possible to see "Fête", "Fete", "Lumières", "Lumière", "Lumiere", etc.

- Removing the line breaks: Useless for the following work.

- Removing the usernames (user-mentions): Usernames in posts allow to make reference to another user (like @user), but don't give us more information about the meaning of the post, so it's not very necessary to keep them.

- Removing the uri: Some people want to link some website thanks to its URI, but it's also made by spam user or ad account. The choice was made to remove them in order to reduce the noise produced. Furthermore, most of the time users use URI shortening, so the process cannot get meaning of words present in the URI. Wikipedia definition for ("URL shortening"): "URL shortening is a technique on the World Wide Web in which a Uniform Resource Locator (URL) may be made substantially shorter in length and still direct to the required page."

- Removing accents: As explained before, we can see non-standard terms or symbolic expressions, written by the users. They express their posts in a specific way without respecting the usual rules. We can find for example for French phrases two parts of a sentence in the same post as "à la mer", "a la montagne" and for those parts of a sentence the accent is mandatory. In removing the accents, the process lose precision but reduce the variety of tokens (word = token). That will facilitate the linking of words for the further work.

4.3.2 Cleansing stage

This step of cleansing will remove anything like the previous one, but clean the token in order to improve the further token connections. Indeed, as the tokens will be split with the space, some cleansing are necessary. The most important stage is to try to leave only the words in the tokens in order to create links between messages. That is to say (for example), to separate all the punctuation with the words.

- Clean the following points: On the data, we can see, that user can write "?????", "!!!!!!" or also "....." and that makes noise for the data mining stage. Thus, the process will remove this succession of punctuation signs and replacing them will one of them.

- Clean space between punctuations: This is a very important part for the process. Indeed, the process will in the data mining stage split the post according to the blank space. Without cleaning the punctuations the process will have tokens as "Hello," for the phrase "Hello, I'm Anthony". At the same time, we will have a sentence like "Hello you I'm Paul", and it will not be easy for the process to make a relation between the two tokens "Hello," and "Hello" if this stage is not accomplished.

- Lowercase: As red in many papers and explained before, lowercase the posts is very important in order to remove the meaningless differentiation between "usa" and "USA" (for example).

4.3.3 Enrichment stage

After removing some noise and cleaning the tokens, the time is to enrich the data in order to improve the relevance of the entire corpus, and reduce the weakness of messages from social networks. In this section, the enrichment is only made by internal resources, as the #hashtags already present in the entire corpus. The goal is to improve the relation between message by trying to have same tokens in the near messages (near in meaning).

- Enrich raw post with hashtags: The main goal of this enrichment is to help the clustering step to make links between words and text. In fact, thanks to the "Data loading" stage and its methods, all the unique hashtags present in the entire corpus are retrieved. After that, the process will check all the tokens in the entire corpus in order to see if a token can match with a #hashtag just got from the database. If there is a match, the process will add a "#" before the token (word), in order to make a #hashtag from the token.

- Enrich raw post with hashtag from at least X user(s): The previous method is interesting but doesn't take in account the relevance of the hashtag. In this methods, the hashtags are kept only when they have been posted at least by X different users in the entire corpus. This allows to remove irrelevant hashtags posted by only one user as for example "#AnthonyIsTheBestOneInTheWord".

As shown in the section 3.3, there were 6354 tweets containing at least one #hashtag. After this step of enrichment (which add the #hashtags already present in the corpus in order to ease the relation links between tweets) there are 10649 tweets containing at least one #hashtag. This step had a big impact on the corpus as it changes 4295 tweets (by adding # before words).

4.3.4 Example of pre-processing on tweets

For this example of pre-processing on tweets, there are several removing, cleansing and enrichment. There is a enrichment with the unique hashtags already present in the entire corpus. First of all, the line breaks have been removed, then the usernames (user mentions), the uri (links) and the accents. After the stage of removing cames the step of cleansing. The following points as ".....", "!!!!!!" have been cleaned and a space between the punctuation and the previous have been added. The folliwing space have been removed and all the text have been lowercase.

Before the pre-processing :

- #PROCOLD https://t.co/MdDWM0FQmW https://t.co/LBgQZyCKss
- Gluhwein at Christmas market PlaceCarnot https://t.co/M78JcGZi7o
- @miko_make @Tiloa_ @FleuranceNature J'ai testé la gamme entière pour peaux mixtes à grasse et elle est top !
- Mal gut, dass es draufsteht... @ Confluence https://t.co/qMmHetjiOj
- @itele La #RUSSIE ???? FAIRE LA PAIX ????? MDRRRRR ..... surtout quand c'est Poutine qui commence ...
- Le PS = Le temple Solaire ou le suicide collectif.
- #Regionales2015 Quand vont-ils comprendre que ce ne sont pas les investissements qui poses problèmes, mais les charges de fonctionnement
- @NeverJokeFlo je sais merci
- Even though it wasn't cold at all, gluhwein is always a good idea! @ Place Carnot https://t.co/MFIpjfphA0
- @DamienRieu Jean Moulin était 1 Patriote qui a dit non au envahisseurs, que de désillusions s'il revenait !

After the pre-processing :

- #procold
- gluhwein at #christmas market placecarnot
- j'ai teste la gamme entiere pour peaux mixtes a grasse et elle est top !
- mal gut , dass es draufsteht . @ #confluence
- la #russie ? faire la #paix ? mdrrrrr . surtout quand c'est poutine qui commence .
- le #ps = le temple solaire ou le suicide collectif .
- #regionales2015 quand vont-ils comprendre que ce ne sont pas les investissements qui poses problemes , mais les charges de fonctionnement
- je sais #merci
- even though #it wasn't cold at all , gluhwein is always a good idea ! @ place carnot

- jean moulin etait 1 patriote qui a dit #non au envahisseurs , que de desillusions s'il revenait !

4.4 Data processing

All the changes made so far, have cleaned and enriched raw text from social networks. However, most of the time, once the two previous stages are done, the raw text is transformed into a structure which makes the rest of the work simpler. In most of the cited papers, they transformed the text into numerical vectors which contain some formulas as TF-IDF. In the paper [32], they used a method which is substantially identical. In fact, the idea and the results are comparable. Word2Vec [23] is as explained on the website "a tool which provides an efficient implementation of the continuous bag-of-words and skip-gram architectures for computing vector representations of words. These representations can be subsequently used in many natural language processing applications and for further research." There is a tool implemented in java, that facilitates the work with Word2Vec [3]. The main problem of word2vec in this case is that is doesn't take into account tweet borders and mixes words from different tweets (according to its window size).

There is another tool which is named Doc2Vec [4]. In this case the conversion step is not performed word by word. Indeed, this tool allows to consider a text fragment from social network (for example, for Twitter a tweet) as a Document. This tool makes it possible to overcome the problem with word2vec on window size.

For this stage, 3 representations have been done, (Word2vec + TF-IDF, Doc2Vec, TF-IDF) in order to see which one deals the best with texts from social networks. These approaches will be presented in the following sections.

4.4.1 Word2Vec

Word2vec is a powerful statistical model for Natural language processing. The input of Word2vec is a text corpus and its output is a set of vectors which are feature vectors for words in that corpus. Word2vec can be applied to every corpus in which patterns may be discerned. It computes vector representations for words which are called word embeddings. Word2vec detects similarities mathematically thanks to its vector representation of words. The purpose of Word2vec is that it is able to group vectors of similar words together in vector space. Word2vec is able to work without the intervention of an human being.

Furthermore, Word2vec includes two different methods to compute word embeddings: continuous bag of words (CBOW) and skip-gram. For CBOW, a word is predicted given its context while for skip-gram the context is predicted given the word. CBOW seems to be faster while skip-gram seems to be slower but does a better job for infrequent words.

The output of Word2vec is a vocabulary set in which each word has a numerical vector representation. So one word is represented as one numerical word. These numerical vector representations allow to detect relationships between words. Indeed, word vectors are positioned in a way that words that share common contexts in the entire corpus are located in the same area in the vector space. Using the vector representation allows to establish word association with other words as for example: (e.g. "Paris" is to "France" what "Berlin" is to "Germany"). The cosine similarity is used in order to detect similarities between words as a 0 degree angle between two vectors represents the best similarity (1) and 90 degree angle represents no similarity at all.

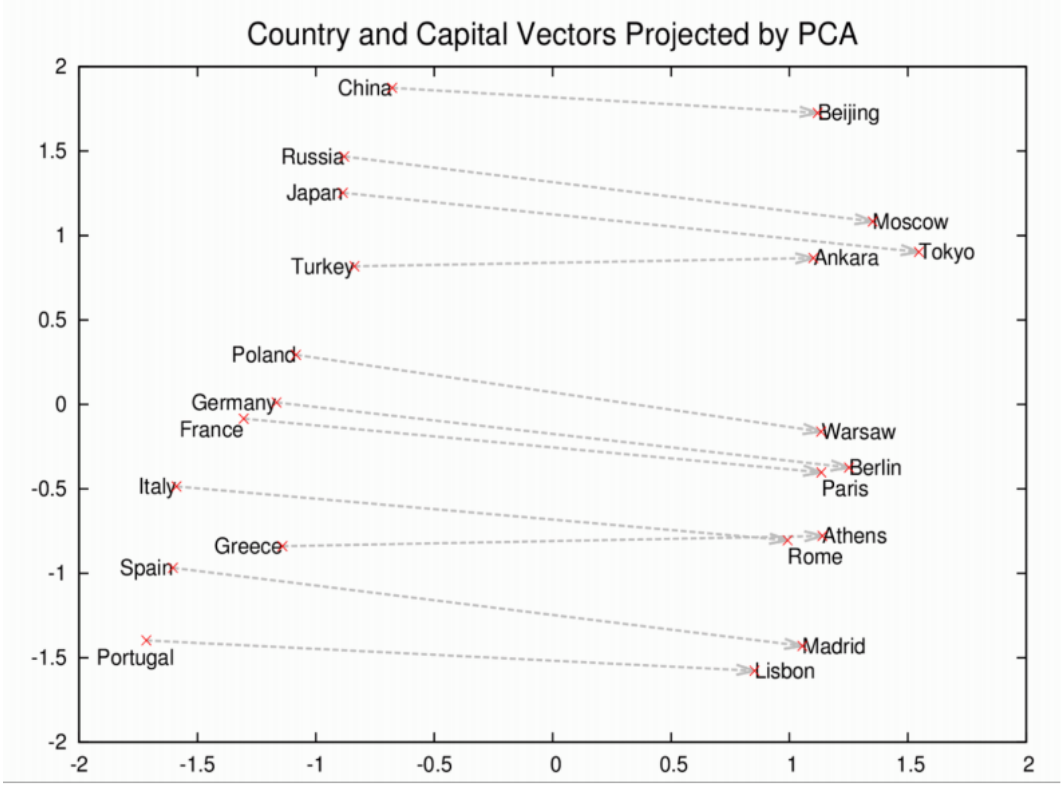


Figure 4.5: Country and capital vectors projected into a vector space (Source [24]).

4.4.2 Word2Vec vectors

As explained before word2vec allows to represent words with numerical vectors. More precisely, Word2vec allows to detect similarities mathematically in grouping vectors of similar words together in vector space. However, the focus have made on tweets from Twitter, so in order to make the possibility of using this method, each “words” numerical vectors will be combined into one “tweet” numerical vectors. Brief schema (Figure 4.6) of what it’s made in order to aggregate all the numerical vectors from work in one numerical vector for tweet.

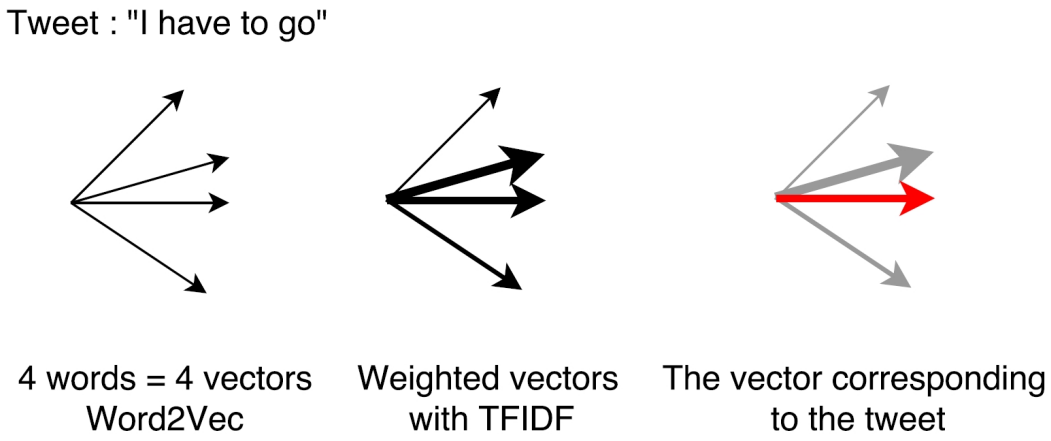


Figure 4.6: Word2Vec + TF-IDF weighting.

However, some words are more relevant than others. To end this difference, the TF-IDF value of each word is used in order to weight each word’s numerical vector. This will make more important the values of the vectors for an important word.

$$\sum_{i=0}^{X}\sum_{j=0}^{Y} TFIDF\left(word\left[j\right]\right) * w2vVector\left(i\right) \tag{4.1}$$

X : Word2vec vector length;
Y : Number of words in the text.

Figure 4.7: Word2Vec + TF-IDF weighting formula.

The following Figure 4.8 explains the previous formula as it’s quite confusing. For the explanation, the following tweet is represented "Bellecour #8Décembre #FDL2015". There are 3 words, so it will have 3 values for the TF-IDF values and 3 word2vec vectors. On the Figure 4.8, the explanation is only based on the word "#FDL2015" to simplify the whole process. However, keep in mind that the formula has to been computed for each word which composed the message (in this case the tweet). In this case here, there are 3 words, so it will be calculated 3 times for the words ("Bellecour" "#8Décembre" "#FDL2015").

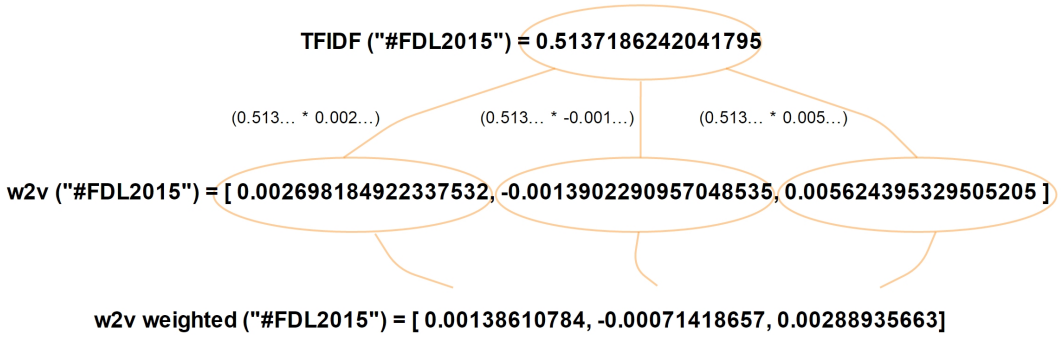


Figure 4.8: Example of the Word2Vec + TF-IDF weighting formula.

$$\#f\vec{dl}2015 = [0.20301395654678345, 0.17443618178367615, ... \\ ..., 0.3125198185443878, -0.010524170473217964] \quad (4.2)$$

Figure 4.9: Exemple of a Word2vec vector.

4.4.3 Doc2vec vectors

As seen with word2vec, it focuses on words and transform a "word" to a numerical vector. However, as in the corpus there are texts, tweets from social networks, a method in order to take in account the entire phrase is wanted. For that Doc2vec [30] which provide a "phrase, tweet" numerical vector is used. So one tweet is represented as one numerical vector. Doc2vec is based on word2vec (Doc2vec is an extension of word2vec) and do almost the same work but for sentences. Doc2vec returns numerical vectors that represent the "meaning" of a document. Doc2vec if for unsupervised learning of continuous representations for larger blocks of text, such as sentences, paragraphs or entire documents.

$$\vec{doc} = [-0.02930622175335884, -0.010446332395076752, ... \\ ..., 0.005861464887857437, 0.00598028302192688] \quad (4.3)$$

Figure 4.10: Exemple of a Doc2vec vector.

4.4.4 TF-IDF vectors

The TF-IDF formula is quite simple, and provides a “relevant measure” value for each word from an entire corpus. In fact, the TF-IDF value is between 0 and 1. When the value is close to 0, it means that the word is common to the overall corpus (it’s maybe a stop word, or a very used word). When the value is close to 1, it means that the word is specific to a given document with

respect to the other documents in the collection. The length of the vectors is the number of unique words in the entire corpus (without duplicates). That makes hollow vectors (in opposite of dense vectors). More the corpus is big, the more chance there is of having large TF-IDF vectors (this represents a considerable problem in practice to use this representation).

$$TFIDF\,(t) = \left(\frac{a}{b}\right) * log\left(\frac{c}{d}\right) \tag{4.4}$$

a : The number of occurrences of the word "t" in the tweet;
b : The length of the tweet (number of word);
c : The total number of analyzed tweets;
d : The number of tweets where the term "t" is present.

Figure 4.11: TF-IDF formula.

$$\vec{tweet} = [0.5137186242041795, 0.7955335394712545, ... \\ ..., 0.21327605555690626, 0.6717281421807599] \tag{4.5}$$

Figure 4.12: Exemple of a TF-IDF vector.

4.5 Data clustering

According to the papers cited before, the clustering algorithms used by researchers in related works are the hierarchical clustering and the clustering based on the density. Most of the time they use the hierarchical clustering, DBSCAN and/or Optics. It was decided to use these three algorithms on the workflow and add the Kmeans algorithm that belongs to the partitioning algorithm family.

This choice is linked with an interest to take advantage of the strengths of each algorithm. These points are mentioned in the comparative study that follows. In the prospect of spending towards clustering in real time where the most optimal algorithm have to be chosen, that is to say having a good compromise between quality results and execution time.

The following table presents a comparative study between the four clustering algorithms.

	Strengths	Weaknesses
Kmeans [10]	- Extensible for processing large data sets. - Faster than hierarchical, DBSCAN and OPTICS.	- Sensitive to outliers. - The specification of the number of clusters beforehand. - Modifying the clusters on the fly not possible (must redo) the calculation.
Hierarchical [7]	- No need to specify the number of clusters. - Level partitioning (you can stop at a desired level). - Applicable to plain text.	- Sensitive to outliers. - Very greedy in terms of memory and it becomes slow with large sets. - The algorithm makes a single pass through the dataset. Individuals who are erroneously assigned will not be reassigned later.
DBSCAN [2]	- Eliminating outliers. - No need to specify the number of clusters. - The possibility of reinstating the new data into clusters without repeating any calculation.	- Unable to build clusters of different densities. - Specification of maximum radius and the minimum points of clusters beforehand. - Relatively slow compared to Kmeans.
OPTICS [15]	- Eliminating outliers. - Construction of clusters of different densities. - No need to specify the number of clusters. - The possibility of reinstating the new data into clusters without repeating any calculation.	- Specification of the minimum points of clusters beforehand. - Relatively slow compared to Kmeans.

4.6 Data extraction

After the clustering step, the process had already group tweets into discussion flows it discovered from the data (that is to say that the process had already separate properly the data). However, the final goal of the entire process is not to cluster the data but, to cluster the data (number 1 on the picture 4.13) in order to detect some discussion flows (some clusters) related to the subject (number 2 on the picture 4.13).

Once the clustering is made, the process have to select the clusters talking about the "Fête des lumières" or more generally about the event followed. These clusters are useful for the enrichment of the beforehand query because they probably contain new hashtags or new words. These tokens can serve for enriching the beforehand bag of words.

Once the process has kept clusters (the discussion flows) that can be used, the process concatenates all the elements together for each cluster in order to do a big document for each cluster containing all the messages. The result is a document per cluster with the concatenated messages. This step is necessary for the TF-IDF formula. In fact, it allows to group the messages talking about the same subject (here the event followed) for extracting the important words.

The process extracts then the X top tokens (words) and Y top hashtags from the clusters selected (number 3 on the picture 4.13) according to the TF-IDF formula. Those words are then re-injected in the beforehand bag of words. This re-injection of words allows to request the API to collect more information related to the subject.

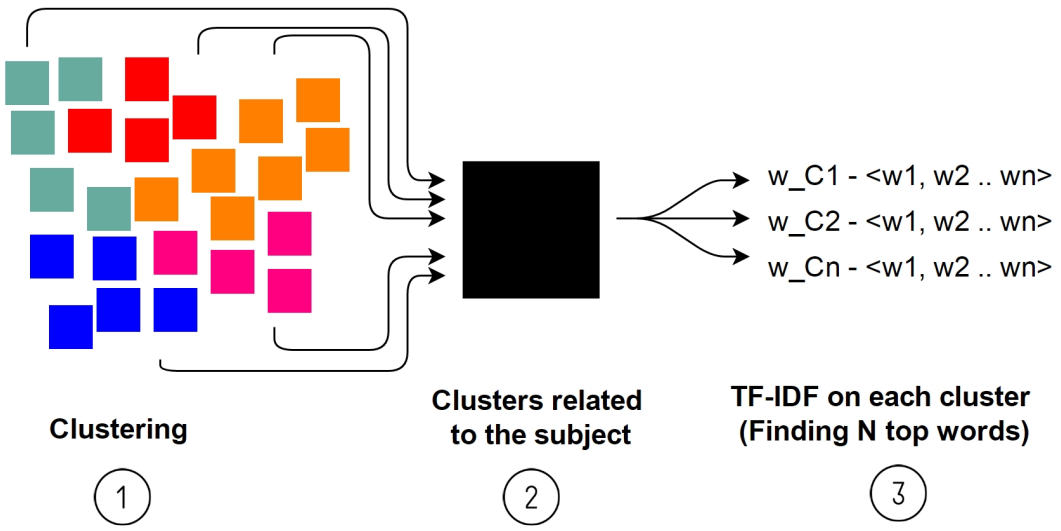


Figure 4.13: Data extraction from clustering.

Table 4.1: Example of top 15 hashtags and words after the extraction step.

	Hashtags	words
1	#lumignon	illuminons
2	#incity	illumine
3	#festivaloflights	recueillement
4	#attentats	basilique
5	#attentatsparis	solidaires
6	#iphone	particuliere
7	#fetedeslumieres...	rebord
8	#tourincity	tok
9	#lpadllyon	halles
10	#igarchitecture	s'illuminent
11	#prayforabetterworld	defilent
12	#lyon-centre	l'immaculee
13	#placebellecour	illuminer
14	#lugdunumsuum	barbarie
15	#fetesdeslumieres	irakiens

4.7 Data visualization

For the data visualization step, a visualization with a map has been chosen because it allow to see the location of each message posted on social network. Furthermore, Google maps API[2] has been chosen because it is easy to use.

Here, the maps allow to see the users movements easily thanks to a real map (Figure 4.14 and Figure 4.15). Indeed, as there are (for some of the tweets) the geo location, a point can be put on the map for each tweet. The points with the same color are tweets that have been posted by the same person (the same user). Furthermore, as the timestamp is available for each tweet, it is possible to follow the user in the city.

As explained before in the subsection 4.1.3, the collectors can take different inputs as keywords and/or geo location. For the corpus made for the "Fête des lumières 2015", the request has collected messages from keywords OR from location. In the Figure 4.16 and the Figure 4.17, it is easy to see that the "Fête des lumières" tweets (orange points on the picture) are mostly present in the region of Lyon. The blue points are tweets that do not talk about the "Fête des lumières".

Finally, in the Figure 4.18 and the Figure 4.19, the dispersion of tweets is shown. It is also possible to see that for the orange points on the Figure 4.19, there is a concentration of points where demonstrations were located.

[2] https://developers.google.com/maps/?hl=fr

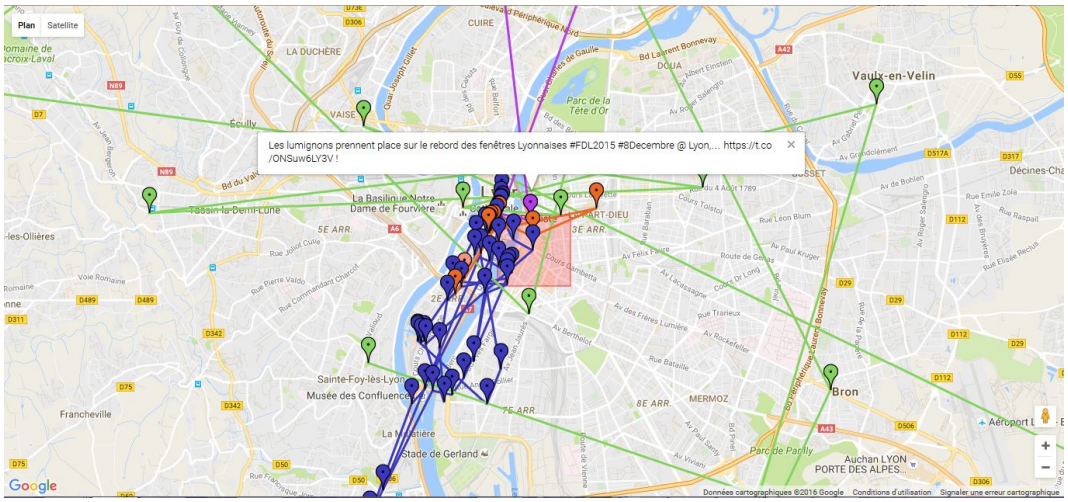


Figure 4.14: Movement of the users who posted a tweet in the neighborhood in red.

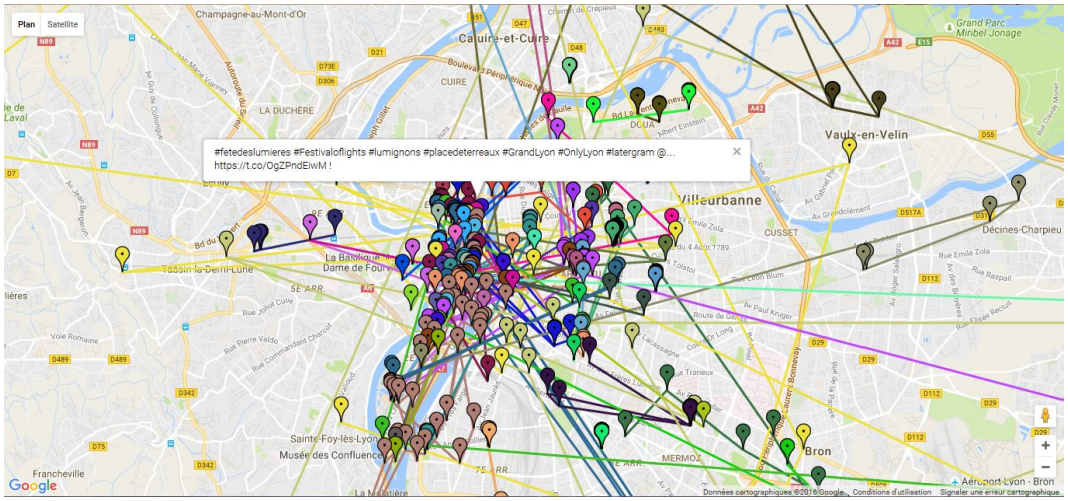


Figure 4.15: Movement of the users who posted a tweet during our collect period.

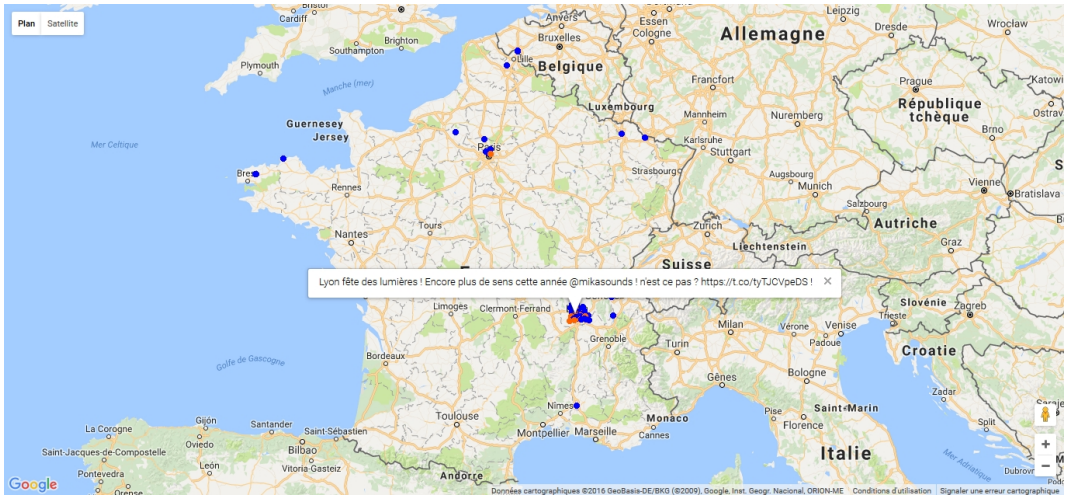


Figure 4.16: Positioning all tweets in France.

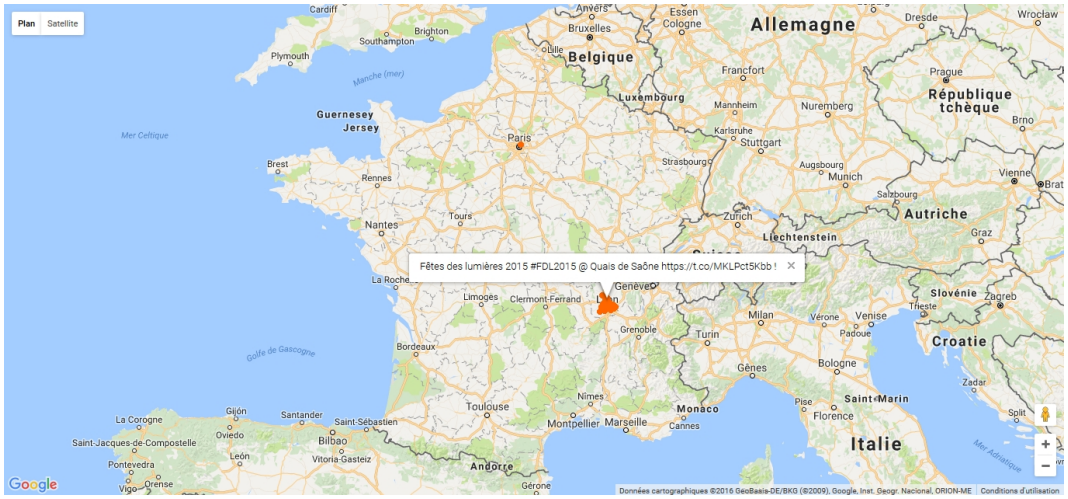


Figure 4.17: Positioning "Fête des lumières" tweets in France.

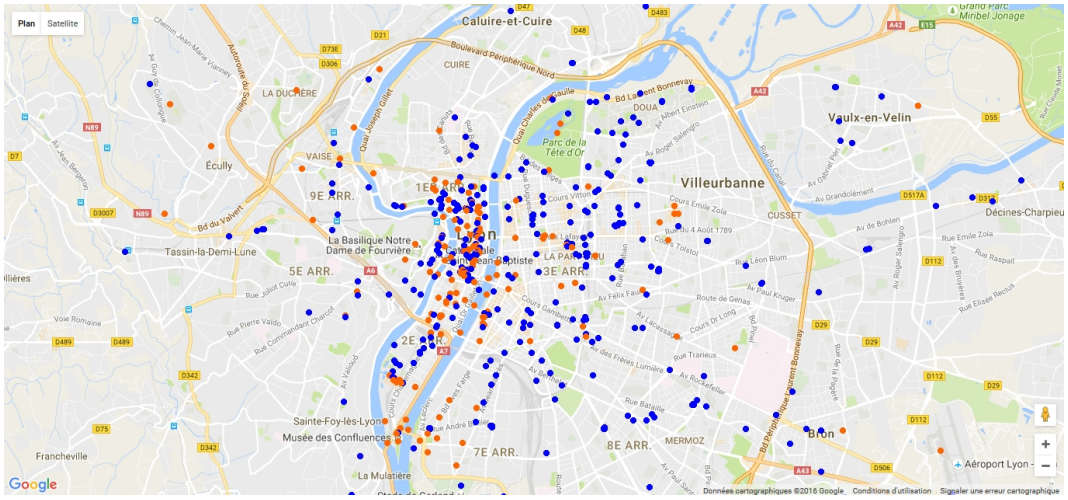


Figure 4.18: Positioning all tweets with a zoom in Lyon.

Figure 4.19: Positioning "Fête des lumières" tweets with a zoom in Lyon.

5 Experimentation

First of all, in order to understand the content of the data collected with the collectors made (subsection 4.1.3) and the habits of the users from social networks, Knime[1] were used. Furthermore the support vector machine (SVM) algorithms were used to see if the clustering with some clustering algorithms was possible. Finally, for the experiments, a backtracking process (section 5.3) have been developed in order to overcome the time consuming problem.

5.1 Knime

The main goal of using Knime was to try to do some modifications on the raw data in order to see what was possible without any knowledge very easily and rapidly. Knime allows to move on very quickly through its system of boxes and arrows. In a short time, a preliminary idea of what it was possible to do with the raw data was discovered.

Concerning the development made on Knime : some layers in order to properly cut the steps were made. Indeed, a pre-processing stage, a clustering stage and a visualization layer were made. Furthermore, thanks to the "Text processing" features[2], a lot of modification on the raw text were done as for example, remove punctuation, remove #hastags, remove word under X characters, remove the stop words, etc.

Finally, the map representation in Knime have been used in order to visualize the geo-location of each tweet. Indeed, it's more interesting if the message has been posted in the location of the event.

Therefore, Knime allowed to move very fast to understanding the data. Moreover, Knime allowed to see very quickly the necessary steps for the pre-processing stage of data. The number of message per user and the frequency of message per user were discovered. Finally, it was also possible to find from where came the messages around the world.

[1] https://www.knime.org/
[2] https://tech.knime.org/knime-text-processing

5.2 Support vector machine

The Support Vector Machine allows to know whether the clustering stage can have good results or not. Indeed, if the supervised learning doesn't work well, the clustering won't give good results.

Concerning the data, as presented in the figure 5.1, the "Fête des lumières" tweets have been taken as positive class and all others as negative class. That is to say that the tweets talking about the event followed have been labeled with the "True" class, and all the other tweets talking about several subjects have been labeled with the "False" class. As presented in the chapter 3, there are more tweets which are not talking about the event in all the corpus retrieved for the project.

The supervised learning serves to find the best representation (representation for the data) for which it is possible to separate both classes in a good way. If there is no good representation, clustering will not find anything useful for the process. For that, two kernels have been tried as :

- Linear kernel (which is default);
- Radial Basis Function kernel (RBF kernel).

If RBF works well than density based clustering might work well too (and other clustering approaches that model mixtures of Gaussians). If linear works well than K-means might work well too. The linear kernel try to separate the data with a linear curve while the rbf kernel try to separate the data in a multidimensional space. The rbf kernel is very useful when the classification of the data is not easy.

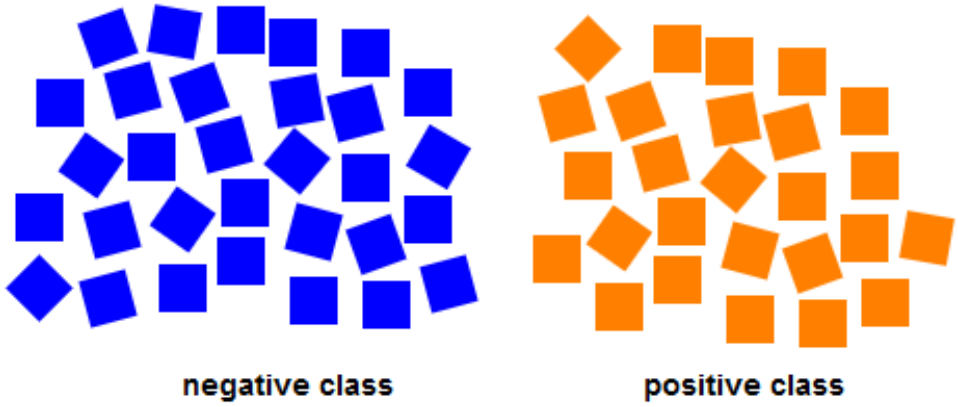


Figure 5.1: SVM partitioning.

5.3 Backtracking process

The backtracking process is an ingenious and very useful process. For the validation, the process have to loop a lot of times for each step in order to try a huge amount of parameters to detect the best ones. However, this validation process is time consuming. Indeed, for each step of the entire process, there are a lot of possible values for the parameters. It would be a shame not to keep the work made a the step n-1. In fact, at a time T, only the parameters of the step "n" are changing, so all the work made for the previous stages can be kept.

For that, the architecture developed is presented in the figure 5.2, which store for each step the result of this one in a serialization format. For example, after the "Data pre-processing" step, the corpus pre-processed is stored, in order to not recalculate it. That is to say, all the tweets which have been cleaned and enriched are serialized. A this stage, a loop for the data processing stage can be made, without having to recompute all the previous stages since they will not change. The binary serialization has been chosen because it is faster to load and because it is easy to do in Java.

Furthermore, the process has been made in a certain way that allow to stop the process at any step wanted. This allows to evaluate a precis layer in the process. For example, this feature has been very useful for the generation of Word2vec and Doc2vec models (section 6.2). Moreover, this feature has also been very useful for the development step in order to evaluate the methods.

Finally, for each step, a pattern file name has been used in order to have consistency in the serialization files. First of all, the number representing the method followed by all the parameters for the method.

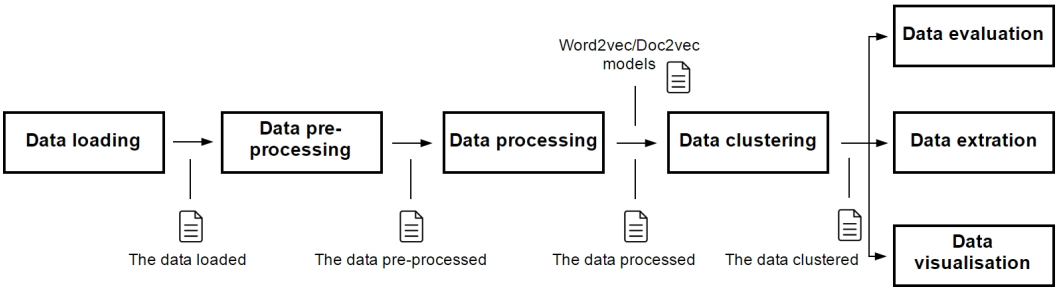


Figure 5.2: Backtracking architecture.

6 Evaluation and validation

In order to constitute the dataset for the “Fête des lumières 2015” in Lyon. The Twitter and the Instagram collectors were launched during 5 days with real-time tools with a set of keywords as input.

Twitter : 31000 messages - 3800 contain images (approximately 12%);
Instagram : 11600 messages;

Figure 6.1: Twitter and Instagram logos.

As there are more messages retrieved from Twitter (31000), for the rest of the work the focus was made on the Twitter dataset.

6.1 Evaluation measures

The main goal of the evaluation step is to fine the best parameters for the previous stages and the best method(s) which overlap the best it can the handmade clusters. For the evaluation the following formulas are used :

6.1.1 Precision, Recall, F1

The precision, the recall and the f1 formulas are widely used in information retrieval in order to evaluate the work. There are measures of the quality of a data classification mechanism. The accuracy of a method of classification or clustering corresponds to its success (it is in percentage because it is a number between 0 and 1), while the recall is its effectiveness. More simply, the precision is "how many selected items are relevant" and the recall is "how many relevant items are selected".

Before presenting the formulas :

- A high recall and a low precision would return many results, but most of them would be incorrect according to the comparison between the predicted labels and the training labels.
- A high precision and a low recall is just the opposite, would return few results, but most of them would be correct according to the comparison between the predicted labels and the training labels.
- A high precision and a high recall would be the ideal process. Indeed, it would return many results, and all the results would be labeled correctly.

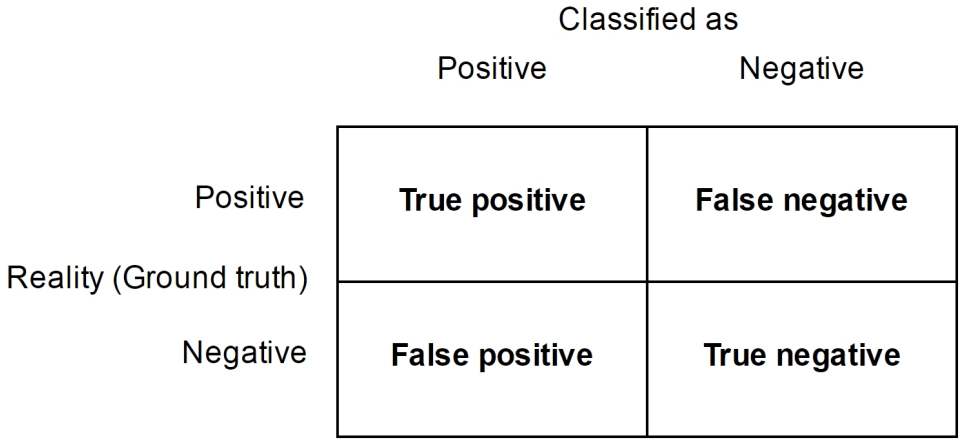


Figure 6.2: Ways of being right or wrong.

As explained in the figure 6.2, there are four ways of being right or wrong when classifying documents under predefined classes. These four ways are useful for the calculation of the precision, the recall and the f1 formula.

- TP is the case was positive and it was predicted positive (NO ERROR);
- TN is the case was negative and it was predicted negative (NO ERROR);
- FN is the case was positive but it was predicted negative (ERROR);
- FP is the case was negative but it was predicted positive (ERROR).

Now all the context of the formulas have been presented, the following formula are : precision (figure 6.3), recall (figure 6.4) and f1 (figure 6.5).

$$Precision = \frac{TP}{TP + FP} \tag{6.1}$$

Figure 6.3: Precision formula.

$$Recall = \frac{TP}{TP + FN} \tag{6.2}$$

Figure 6.4: Recall formula.

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{6.3}$$

Figure 6.5: F1 formula.

6.1.2 Rand index

The rand index formula is particularly used in data clustering in order to evaluate the clusters. Indeed, it is a measure of the similarity between two data clusterings. The formula of the rand index is quite simple and is based on pairs of elements. In fact, the formula checks if two pairs of elements are in the same cluster for the both clusterings. The formula is also based on the four values (TP, FN, FP and TN) as explained in the figure 6.2 but are quite different as it focus on pairs of elements.

For the explanation, two clusterings are needed X and Y.

- a : the number of pairs of elements that are in the same cluster in X and in the same cluster in Y
- b : the number of pairs of elements that are in different clusters in X and in different clusters in Y
- c : the number of pairs of elements that are in the same cluster in X and in different clusters in Y
- d : the number of pairs of elements that are in different clusters in X and in the same cluster in Y

"Intuitively, a + b can be considered as the number of agreements between X and Y and c + d as the number of disagreements between X and Y." (Wikipedia) The explanation is inspired by the wikipedia definition[1].

$$\text{Rand index} = \frac{a+b}{a+b+c+d} \tag{6.4}$$

Figure 6.6: Rand index formula.

6.1.3 Normalized Mutual Information (NMI)

As explained in [35] and in [25], Mutual information is a very popular measure for comparing clusterings.
As the NMI formula is "normalized" (Normalized .. mutual information), it can be used to compare clusterings with different numbers of clusters (and that is very interesting). The result of the NMI formula is always a number between

[1] https://en.wikipedia.org/wiki/Rand_index

0 and 1. The minimum of $I(\Omega;\mathbb{C})$ is 0 if the clustering is random with respect to class membership. This formula is based on the entropy of both clusterings and on the mutual information (MI) which is of a measure of the mutual dependence between the two random variables. The MI is particularly used in probability theory and information theory. However, it does not penalize large cardinalities so fewer clusters are better. For this problem, the normalization by the denominator $[H(\Omega) + H(\mathbb{C})]/2$ fixes this problem since entropy tends to increase with the number of clusters.
The following formulas come from [13].

$$NMI(\Omega;\mathbb{C}) = \frac{I(\Omega;\mathbb{C})}{(H(\Omega) + H(\mathbb{C}))/2} \tag{6.5}$$

Figure 6.7: Normalized Mutual Information formula.

Where $\Omega = \{\omega_1, \omega_2, \ldots, \omega_K\}$ is the set of clusters and $\mathbb{C} = \{c_1, c_2, \ldots, c_J\}$ is the set of classes.

$$I(\Omega;\mathbb{C}) = \sum_k \sum_j P(\omega_k \cap c_j) * log\left(\frac{P(\omega_k \cap c_j)}{P(\omega_k)P(c_j)}\right) \tag{6.6}$$

Figure 6.8: Mutual information formula.

Where $P(\omega_k)$, $P(c_j)$, and $P(\omega_k \cap c_j)$ are the probabilities of a document being in cluster ω_k, class c_j, and in the intersection of ω_k and c_j, respectively.

$$H(\Omega) = -\sum_k P(\omega_k) * log(P(\omega_k)) \tag{6.7}$$

Figure 6.9: Entropy formula.

6.2 Models generation

The main goal of the evaluation step was to find the best vector representation of a message from social network. As explained before in the section 4.4, the focus was made on three vector representations (Word2vec (subsection 4.4.2), Doc2vec (subsection 4.4.3) and TF-IDF (subsection 4.4.4)). The TF-IDF representation depends only of the content of the message, but the Word2vec and Doc2vec depend on the Word2vec or Doc2vec models. So, in order to find the best representation of a tweet, 729 Doc2vec and 729 Word2vec models were generated. They were generated on the entire corpus in order to improve the precision. For the generation, some parameters varied of the Word2vec and Doc2vec models as the vector length, the window size, the minimum word

frequency, the net iteration, the layer size and the learning rate. The focus was made on these parameters according to the explanation from [36]. The other parameters remained unchanged for all the other models. The unchanged parameters were :

- Min learning rate : 0.0001;
- Batch size : 1000;
- Sub sampling : 0.001.

Table 6.1: Doc2vec parameters which varied.

Parameter	Values
Learning rate	0.00025, 0.025, 2.5
Vector length	70, 100, 200
Window size	5, 8, 10
Minimum word frequency	2, 5, 8
Net iterations	5, 10, 100
Layer size	200, 300, 400

6.3 Validation methods

Several ways were used to validate the process introduced so far. For most of the experiments carried out, the handmade clustering was used in order to have consistent and verifiable models. As explained before, the entire corpus was handmade clustered. The two main ways were based on Support Vector Machine "SVM" for the classification and the clustering with Kmeans in order to get exactly the same number of cluster wanted. So, there are a classification and a clustering way to validate the entire process. Both methods are completely different and do not act in the same way.

6.3.1 Classification - Support Vector Machine

Support vector Machines are supervised learning models that analyze data for classification and/or regression analysis. The classification process can be divided into two steps. A phase of learning (number 1 on the picture 6.10) and one of classification (number 2 on the picture 6.10). The SVM training algorithm needs to be train with a set of training examples (which each elements marked for belonging to one of the two categories (classes). The SVM training algorithm builds a model that assigns the test data into one of the two categories.

The SVM model is a representation of points in space as on the figure 6.11

mapped in a way that the separate categories are divided by a clear gap (that is wide as possible). The gap is represented on the figure 6.11 by the red line (which is the optimal hyperplane) and the two grey lines (which are the maximum margins). However, for a set of linearly separable points, there exists an infinity of hyperplanes separators. It's why, the support vector machine try to find the optimal hyperplane since there is a single optimal hyperplane defined as a hyperplane that maximizes the margin between the samples and the separating hyperplane.

Once the model have been trained, the new samples (from the test data) are mapped into the same space (previously made during the training step). This will allows the process to predict the category in which the element belongs based on which side of the gap the element is.

A real advantage to using support vector machines (SVM) is that no prior knowledge of the content of data is required.

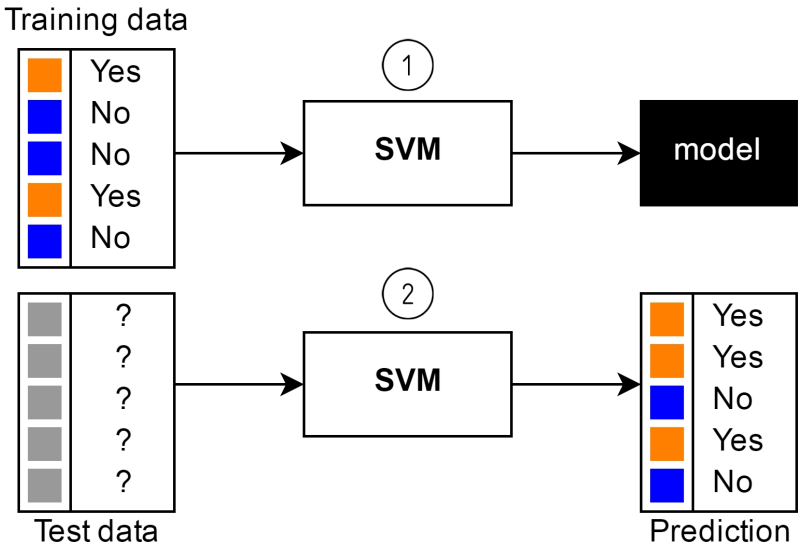


Figure 6.10: SVM process

For the experiments, the focus was made on the Doc2vec models which were generated (as explained in the section 6.2). For each SVM model, it was fitted with 80% of the entire corpus. The model was then tested on the rest of the corpus (20% of the entire dataset), in order to do a binary classification ("Fête des lumières" and "NOT Fête des lumières").

This first step has allowed to classify data for two specific categories which were clearly defined. However, as explained before, the main goal of the entire process is to find some new tokens (words) to enrich the beforehand bag of words, that is to say to enrich the request. The enrichment of the request

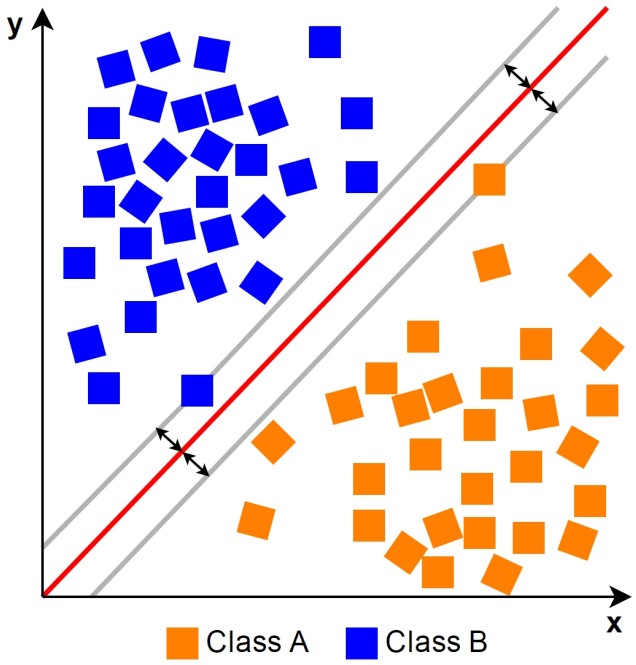


Figure 6.11: Example of a problem of discrimination in two classes. The problem is linearly separable.

therefore improve the collection of data in real time. Once the tweets have been grouped into two distinct categories, that is to say after the classification phase, only the positive tweets (the tweets with the "Fête des lumières" label) were kept.

Finally, as explained in the section 4.6, the goal is now to extract the top X hashtags or tokens of this set containing tweet talking about the "Fête des lumières". You will find the results in the chapter 7.

6.3.2 Clustering

As explained before, the focus was made on 4 clustering algorithms. However, for the clustering evaluations, 1 algorithm has been tested (Kmeans). First of all, Kmeans has been chosen in order to get exactly the number of clusters wanted as the experiments are based on two clusters ("Fête des lumières" and "NOT Fête des lumières"). DBScan is kept in mind because it allows a scalable clustering. Indeed, it could be interesting to not recompute all the data each time the clustering is made. According to [27], DBScan seems to be a good clustering algorithm in order to evolve over time. Indeed, it is a method based on the density, so more optimized for clustering flows than Kmeans or hierarchical clustering. In fact, with Kmeans or hierarchical clustering it is mandatory to redo all the calculations for each time window.

7 Results

As explained in the section 6.2, 729 doc2vec and 729 word2vec models have been generated on the big corpus (31000 tweets) for the evaluation step. These models have been tested for each of the development made, so in this section, the results for the SVM models (subsection 6.3.1) and for the clusterings (subsection 6.3.2) will be presented. However, as there are a lot of models, only the best and the worst result will be kept for the following evaluations.

As explained in the subsection 3.3.1, the entire corpus have been handmade clustered. However, for the evaluation step, three samplings of the corpus will be evaluated. One corpus, representing 700 tweets (one hour collection) from December 08, 2015 at 07pm until December 08, 2015 at 08pm, one corpus representing 7000 tweets in order to keep the same proportions as the small one and the entire corpus.

The sub-sampling of 700 tweets have been chosen for many reasons : 17 % (122/700) of the corpus are tweets talking about the "Fête des lumières" event, and this 700 tweets have been taken during the highest moment of the event, as shown in the Table 3.4. The corpus with 7000 tweets has been chosen in order to keep the proportions of tweets talking about the event. Indeed, there are 1048 tweets on 7000 : 15% (1048/7000) which talk about the "Fête des lumières". Proportionally, in the largest corpus there are fewer messages referring to the event followed ("Fête des lumières 2015"). The are 3 % (1048/31008) of the corpus which is talking about the event.

As explained in subsection 4.4.4, the TF-IDF representation is a good measure, however it is difficult to use it in practice. Indeed, for example, for the experiments on 31000 tweets (for the classification and the clustering), the TF-IDF representation used 47GB of RAM (Random Access Memory). The java heap space has been enlarged in order to compute this representation.

For each of the experiments, the results are ranked according to the rand index measure. Indeed, as the rand index value check if the groups of elements are good without worrying about labels (classes), this is a good valuable result.

Moreover, the rand index value is used because with the clustering method, the process cannot know which one of the both cluster is the one talking about the event followed. For these results, once the clustering is made, the predicted classes are changed (if it was 1, it becomes 0 and vice versa in order to have both clustering possibilities.

7.1 Classification - Support Vector Machine

Most of the time, for learning methods data slicing is substantially identical. The largest part is given to the learning step (training step) and the rest is given for testing the model previously generated thanks to the training step. In the following experimentations 80% of the corpus was given for the training and 20% for the testing. As explained in the chapter 7, three corpus have been tested in order to compare the results with more or less data. For the small corpus (700 tweets), 560 posts have been used for the training (which represents 80%) and 140 posts for the tests (which represents 20% of the corpus). For the second one, (7000 tweets), 5600 posts have been used for the training (which represents 80%) and 1400 posts for the tests (which represents 20% of the corpus). For the entire corpus (31000 tweets), 24800 posts have been used for the training (which represents 80%) and 6200 posts for the tests (which represents 20% of the corpus)

7.1.1 Classification on 700 tweets

Evaluation

For this classification on 700 tweets, there are 16 tweets talking about the event ("Fête des lumières") and 124 tweets which are noise for the process in the test corpus.

Table 7.1: Results of the classification on 700 tweets.

	Precision	Recall	F1	Rand index	nmi
TF-IDF linear	83,34%	93,75%	88,24%	94,41%	69,29%
All others	84,21%	100%	91,43%	95,78%	78,72%

Extraction

As presented in the section 3.1, the initial request for collecting the corpus was with these following keywords #Lyon, #FeteDesLumieres2015, #FDL2015 and #candle. With this classification, it would have been possible to improve, (to enrich) the request with the following #hashtags: #fourviere, #8decembre, #lumignons, #fetedeslumieres, #light, #illuminations, #lumieres2015,

Table 7.2: Confusion matrix on 700 tweets.

TF-IDF linear		False (GT)	True (GT)	
	False (predicted)	121 (TN)	1 (FN)	122
	True (predicted)	3 (FP)	15 (TP)	18
		124	16	140
All others		False (GT)	True (GT)	
	False (predicted)	121 (TN)	0 (FN)	121
	True (predicted)	3 (FP)	16 (TP)	19
		124	16	140

#lumieres. (taken from the Table 7.3). For the classification with the TF-IDF representation and the linear kernel, there is one difference in the top 15 #hashtags extraction which is : The #hashtag 3 which is #8decembre is replaced by #hommageauxvictimes.

Table 7.3: Top 15 hashtags extraction from the classification on 700 tweets.

	Hashtags FDL	Hashtags Noise
1	#republique	#on
2	#fourviere	#non
3	#8decembre	#car
4	#lumignons	#sarkozy
5	#fetedeslumieres	#fn
6	#fdl2015	#time
7	#fete	#lol
8	#light	#tpmp
9	#france	#c
10	#troptroptroptrofiere	#voyage
11	#illuminations	#love
12	#running	#histoire
13	#lumieres2015	#hair
14	#operadelyon	#mdr
15	#lumieres	#boldandbeautiful

Discussion

As presented in the Table 7.1 and in the Table 7.2, there does not seem to be any difference between both vector representations (Word2vec and Doc2vec). Indeed, it is likely that the problem is too easy for the classification task. Moreover, the 729 variations for both models (Word2vec and Doc2vec) don't seem to change anything. For almost all vector representations (expect the TF-IDF representation with the linear kernel) and all the parameters variations, there is a recall of 100% which means that the process has found all tweets about the event follow. Moreover,as show in the Table 7.2, there are only 3 tweets which are incorrectly classified (FP) for all the representation expect the TF-

IDF representation with the linear kernel. (FP) False Positive means that the 3 tweets were False (not talking about the event) and the classifier said that they were Positive (talking about the event). The recall value (Table 7.1) is perfect, because the classifier is not mistaken for the Positive ones (FN). It is able to find all the positive tweets, even to find too much positive tweets (FP). The TF-IDF representation with the linear kernel seems to be a little be less accurate as all the other representations. However, the results remain good. Finally, there are good results and it gives good prospects for the next corpus.

According to the Table 7.3, there are 8 #hashtags which are good in order to improve the initial request. Moreover, there are 3 #hashtags which are related to the city of Lyon, but not related to the event. This result was expected because is not easy to cluster the data, moreover it depends on the person who made the clusters (the handmade clustering). The clustering is subjective. Aaccording to the person who has clustered the data, some tweets have been clustered as the event followed "Fête des lumières" when they were just attached to the city of Lyon. Finally, there is a good point since there is no hashtag that could enrich the query in the top hashtags of noise.

7.1.2 Classification on 7000 tweets

As explained before, the corpus of 7000 tweets have been chosen in order to keep the same proportionality as the small one. Indeed, for the corpus of 700 tweets, there are 122 tweets talking about the event and in this corpus of 7000 tweets there are 1048 tweets talking about the event. The proportions are kept. For this classification on 7000 tweets, there are 191 tweets talking about the event ("Fête des lumières") and 1209 tweets which are noise for the process in the test corpus in the test corpus.

Evaluation

Table 7.4: Results of the classification on 7000 tweets.

Model	Precision	Recall	F1	Rand index	nmi
D2Vs + TF lin	99,48%	100%	99,74%	99,86%	98,72%
W2V linear	100%	100%	100%	100%	100%
W2V rbf	100%	98,95%	99,47%	99,71%	97,67%
TF-IDF rbf	98,96%	100%	99,48%	99,71%	97,68%

Extraction

As presented in the section 3.1, the initial request for collecting the corpus was with these following keywords #Lyon, #FeteDesLumieres2015, #FDL2015

Table 7.5: Confusion matrix on 7000 tweets.

D2Vs + TF lin		False (GT)	True (GT)	
	False (predicted)	1208 (TN)	0 (FN)	1208
	True (predicted)	1 (FP)	191 (TP)	192
		1209	191	1400
W2V linear		False (GT)	True (GT)	
	False (predicted)	1209 (TN)	0 (FN)	1209
	True (predicted)	0 (FP)	191 (TP)	191
		1209	191	1400
W2V rbf		False (GT)	True (GT)	
	False (predicted)	1209 (TN)	2 (FN)	1211
	True (predicted)	0 (FP)	189 (TP)	189
		1209	191	1400
TF-IDF rbf		False (GT)	True (GT)	
	False (predicted)	1207 (TN)	0 (FN)	1207
	True (predicted)	2 (FP)	191 (TP)	193
		1209	191	1400

and #candle. With this classifications, it would have been possible to improve, (to enrich) the request with the following #hashtags: #8decembre, #fetedeslumieres, #lumignons, #lumieres, #hommage, #mercimarie, #lumiere, #victimes, #fourviere, #hommageauxvictimes. (taken from the Table 7.6). For the classification with the Word2vec representation and the rbf kernel, there is one difference in the top 15 #hashtags extraction which is : The #hashtag 2 which is #fetedeslumieres is replaced by #light.

Table 7.6: Top 15 hashtags extraction from the classification on 7000 tweets.

	Hashtags FDL	Hashtags Noise
1	#8decembre	#valls
2	#fetedeslumieres	#non
3	#lumignons	#cauchemarchezlecoiffeur
4	#lumieres	#c
5	#hommage	#fn
6	#fete	#mdr
7	#mercimarie	#chien
8	#lumiere	#lrds
9	#fdl2015	#rt
10	#victimes	#mtvstars
11	#lyonnais	#love
12	#musee	#totasm
13	#fourviere	#profilage
14	#hommageauxvictimes	#car
15	#confluence	#envoyespecial

Discussion

According to the results which are presented in the Table 7.4 and in the Table 7.5, there seem to have more differences between the vector representations than for the small corpus. Indeed, both kernel (linear and rbf) with the Doc2vec representations and the TF-IDF representation with the linear kernel seem to give exactly the same results. With a precision of 99,48% and a recall of 100% the numerical vector representations seem to be almost perfect. Furthermore, the word2vec representation with the linear kernel seems to be perfect. Indeed, there are no error in the classification. The Word2vec representation and the TF-IDF representation, both with the rbf kernel seem to perform the worst. Moreover, the results remain good for the classification task. Furthermore, the 729 variations for both models (Word2vec and Doc2vec) don't seem to change anything. Indeed, for each parameter variations the results remain the same. Finally, the results are better than for the small corpus of 700 tweets. Moreover, there are the same classes distribution. As expected, when the classifier train its model on more ground truth examples, it is able to be more accurate in classifying the test sub-sampling. In this classification, there are 10 times more elements to classify and the classifier is better. Indeed, with the Doc2vec representations and the TF-IDF (with linear kernel) there is only one tweet which is incorrectly classified (FP). As explained in the previous classification, (FP) False Positive means that the tweet was False (not talking about the event) and the classifier said that it was Positive (talking about the event). The recall value (Table 7.4) is also perfect, because the classifier is not mistaken for the Positive ones (FN).

It is interesting to note that the Word2vec representation seems to be the best. In addition, the RBF kernel does not seem to be adapted to the data as it provides less good results. Indeed, it seems that the data is linearly separable.

According to the Table 7.6, there are 10 #hashtags which are good in order to improve the initial request. Moreover, there are 2 #hashtags which are related to the city of Lyon, but not related to the event. As explained before this result was expected because is not easy to cluster the data, moreover it depends on the person who made the clusters (the handmade clustering). Finally, there is a good point since there is no hashtag that could enrich the query in the top hashtags of noise.

7.1.3 Classification on 31000 tweets

Evaluation

For this classification on 31000 tweets, there are 198 tweets talking about the event ("Fête des lumières") and 6004 tweets which are noise for the process in the test corpus.

Table 7.7: Results of the classification on 31000 tweets.

Model	Precision	Recall	F1	Rand index	nmi
D2Vs + TF lin	99,5%	100%	99,75%	99,97%	99,09%
W2V linear + TF rbf	100%	100%	100%	100%	100%
W2V rbf	100%	98,99%	99,49%	99,94%	98,33%

Table 7.8: Confusion matrix on 31000 tweets.

D2Vs + TF lin		False (GT)	True (GT)	
	False (predicted)	6003 (TN)	0 (FN)	6003
	True (predicted)	1 (FP)	198 (TP)	199
		6004	198	6202
W2V linear + TF rbf		False (GT)	True (GT)	
	False (predicted)	6004 (TN)	0 (FN)	6004
	True (predicted)	0 (FP)	198 (TP)	198
		6004	198	6202
W2V rbf		False (GT)	True (GT)	
	False (predicted)	6004 (TN)	2 (FN)	6006
	True (predicted)	0 (FP)	196 (TP)	196
		6004	198	6202

Extraction

As presented in the section 3.1, the initial request for collecting the corpus was with these following keywords #Lyon, #FeteDesLumieres2015, #FDL2015 and #candle. With this classification, it would have been possible to improve, (to enrich) the request with the following #hashtags: #fourviere, #attentatsparis, #hommage, #hommageauxvictimes, #prayforparis, #fetedeslumieres. (taken from the Table 7.9).

Discussion

According to the results which are presented in the Table 7.7 and in the Table 7.8, it does not appear to be large differences with the corpus of 7000 tweets. Indeed, when comparing the two tables (Table 7.4 and Table 7.7), that there is a small difference. This could be explained by the fact that the large corpus does not allow the classification to improve its model a lot. The

Table 7.9: Top 15 hashtags extraction from the classification on 31000 tweets.

	Hashtags FDL	Hashtags Noise
1	#fdl2015	#c
2	#fourviere	#fn
3	#saone	#mdr
4	#attentatsparis	#love
5	#hommage	#lrds
6	#hommageauxvictimes	#valls
7	#prayforparis	#ps
8	#fetesdeslumieres	#cop21
9	#incity	#smile
10	#g7x	#lol
11	#granderouelyon	#sarkozy
12	#fakedeslumieres	#car
13	#ilovelyon	#cauchemarchezlecoiffeur
14	#croixrousse	#estrosi
15	#wheel	#rt

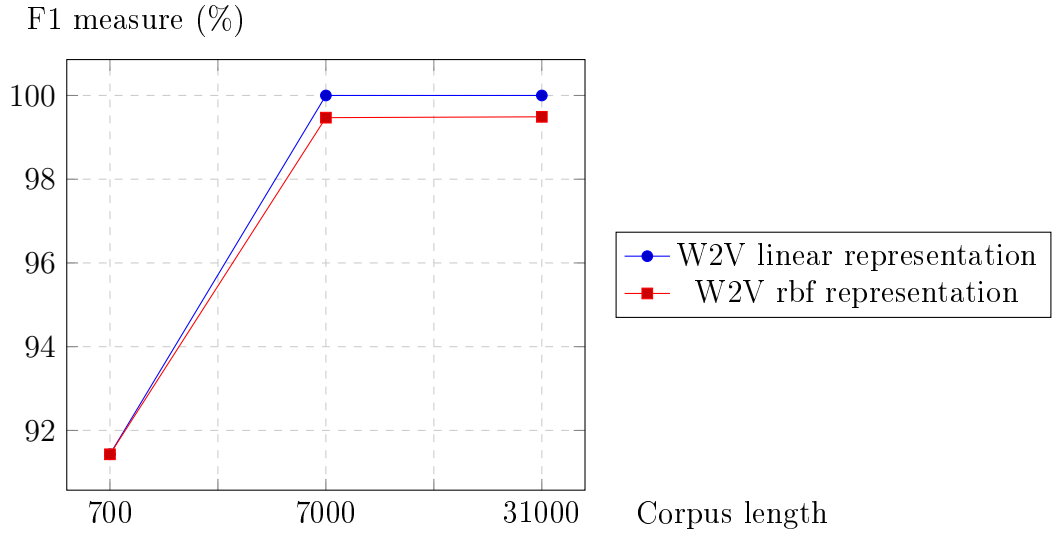
results are thus substantially identical although a little better. It is possible to distinguish that there are 3 groups of results. Actually, the Doc2vec representations (both kernel) and the TF-IDF representation with the linear kernel give the same results which are good. Moreover, the Word2vec representation with the linear kernel and the TF-IDF representation with the rbf kernel give also the same results. Both of the previous cited representations seem to be the best. Indeed with a precision, a recall and a F1 of 100%. Furthermore, the Word2vec representation with the rbf kernel remain the worst one as presented for the previous corpus with 7000 tweets. Everything is relative, being the worst with 99% accuracy, 100% recall and 99,75% recall is still a good result.

According to the Table 7.9, there are 6 #hashtags which are good in order to improve the initial request. Moreover, there are 5 #hashtags which are related to the city of Lyon, but not related to the event. It is possible to see there, that the extraction is not so good as before. This could be explained by the fact that the distribution of the classes is not balanced. Indeed, there are almost 29 times more noise tweets as tweets talking about the event. Finally, there is a good point since there is no hashtag that could enrich the query in the top hashtags of noise. It is also interesting to see that in the extraction of the #hashtags for the noise classification, there are a lot of politics. Indeed, #valls, #sarkozy, #estrosi are name of politicians and #fn and #ps are both French political parties.

7.1.4 Discussion about the classification of the whole corpus

Corpus	W2V linear	W2V rbf
700	91.43	91.43
7,000	100	99.47
31,000	100	99.49

Table 7.10: Table of the best and the worst representations with the classification according to the F1 measure.



The chart shows therefore the best (Word2vec linear) and the worst (Word2vec rbf) representation according to the F1 measure.

Figure 7.1: Curves of the best and the worst representations with the classification according to the F1 measure.

To conclude on the overall results concerning the classification, it must be mentioned that the results are good. Actually, the worst result for the classification concern the small corpus which contains 700 tweets with 83,84% precision, 93,75% recall, 88,24% f1, 94,41% rand index and 69,29% nmi. Being the worst result with this results remain good result.

The results which are lower for the small corpus (containing 700 tweets) can be explained by the fact that there are not enough ground truth examples to train the SVM model correctly. Indeed, as explained before, the corpus has been sliced in order to give 80% to the training step and 20% to the testing step. Actually, 560 tweets (or more precisely 560 numerical vectors representing the tweets) are not enough to train the SVM model correctly. Everything is relative, it is not enough in order to be perfect.

As both corpus (700 and 7000 tweets) are more comparable, the comparison will be made between them. The Doc2vec representations (for both kernel) and the TF-IDF representation with the linear kernel seem to give exactly the same results. Actually, according to Table 7.4 and to Table 7.7, the results for the big corpus seem to be better with 99,75% against 99,74% for the corpus with 7000 tweets. However, taking a look in the confusion matrices Table 7.5 and Table 7.8, the results are exactly the same in the number of errors. Indeed, there is only one (FP - False Positive) for both results. False Positive (FP) means that the tweet was False (not talking about the event) and the classifier said that it was Positive (talking about the event) which is an error. However, as the large corpus contains more tweets, the percentage for the precision, the recall etc. increase. It would be interesting to see if this is not an error in the handmade clustering.

According to Table 7.4 and to Table 7.7, the Word2vec representation with the rbf kernel seems to make errors in classifying with the FN label (False Negative). That means that the tweet was True (talking about the event) and was predicted False (not talking about the event). Otherwise, all the other representations seem to make errors in classifying with the FP label (False Positive). As explained before, False Positive (FP) means that the tweet was False (not talking about the event) and the classifier said that it was Positive (talking about the event).

For the extraction step, according to the Table 7.3, Table 7.6, and Table 7.9, the best extraction is done with the corpus containing 7000 tweets. Indeed, there are 10 #hashtags which are good in order to improve the initial request. Moreover, there are 2 #hashtags which are related to the city of Lyon, but not related to the event. Moreover, for all the representation there is no hashtag that could enrich the query in the top hashtags of noise. It is also interesting to see that in the extraction of the #hashtags for the noise classification, there are a lot of politics with politicians and French political parties.

Finally, as explained before, the Doc2vec and Word2vec parameter variations don't seem to change anything, so the process is not able to give the best parameters representation for a tweet. However, the Word2vec representation with the linear kernel seems to be the best one (and the perfect one) because it doesn't make any errors in classifying. Furthermore, it is also possible to see that the Word2vec representation with the rbf kernel seems to be the worst one (but the result remain good). According to all the results from the clas-

sification part, it is possible to assume that the data seems to be linearly separable.

7.2 Clustering - Kmeans

For the clustering evaluations, the process try to match the best it can the handmade clusters with the Kmeans algorithm to get exactly the same number of cluster. Indeed, as the main goal of the process is to only keep the tweets talking about the event (that is to say about the "Fête des lumières"), the process clusters the dataset into 2 parts ("FDL" and not "FDL"). In contrast to the classification, for the clustering part, all the tweets of each corpus are given to the process in order to cluster them. That is to say that there is no slicing (80% - 20%).

Furthermore, as explained in the section 5.2, if the linear kernel with the support vector machine gives good results, that means that the data is linearly separable and that an algorithm like Kmeans should work well. According to the results from the section 7.1, K Means algorithm should give good results.

7.2.1 Clustering on 700 tweets

Evaluation

For this clustering on 700 tweets, there are 122 tweets talking about the event ("Fête des lumières") and 578 tweets which are noise for the process.

Table 7.11: Results of the clustering on 700 tweets.

Model	Precision	Recall	F1	Rand index	nmi
D2V worst one	20,36%	64,75%	30,98%	49,93%	0,66%
D2V best one	16,36%	86,07%	27,49%	66,94%	1,1%
W2V worst one	19,46%	95,08%	32,31%	57,49%	1,57%
W2V best one	88,24%	61,48%	72,46%	85,02%	30,19%
TF-IDF	17,45%	100%	29,72%	70,99%	0,12%

Extraction

As presented in the section 3.1, the initial request for collecting the corpus was with these following keywords #Lyon, #FeteDesLumieres2015, #FDL2015 and #candle. With this clustering, it would have been possible to improve, (to enrich) the request with the following #hashtags: #fete, #fdl2015, #fourviere and #hommageauxvictimes. (taken from the Table 7.16). For the clustering

Table 7.12: Confusion matrix on 700 tweets.

D2V worst one		False (GT)	True (GT)	
	False (predicted)	269 (TN)	43 (FN)	312
	True (predicted)	309 (FP)	79 (TP)	388
		578	122	700
D2V best one		False (GT)	True (GT)	
	False (predicted)	41 (TN)	17 (FN)	58
	True (predicted)	537 (FP)	105 (TP)	642
		578	122	700
W2V worst one		False (GT)	True (GT)	
	False (predicted)	98 (TN)	6 (FN)	104
	True (predicted)	480 (FP)	116 (TP)	596
		578	122	700
W2V best one		False (GT)	True (GT)	
	False (predicted)	568 (TN)	47 (FN)	615
	True (predicted)	10 (FP)	75 (TP)	85
		578	122	700
TF-IDF		False (GT)	True (GT)	
	False (predicted)	1 (TN)	0 (FN)	1
	True (predicted)	577 (FP)	122 (TP)	699
		578	122	700

with the Word2vec representation, there are differences in the top 10 #hashtags extraction which are : #illuminations, #monlyon, #prayforabetterworld and #lumiere.

Table 7.13: Top 10 hashtags extraction from the clustering on 700 tweets

	Hashtags FDL (D2V)	Hashtags FDL (W2V)
1	#france	#hommageauxvictimes
2	#mdr	#foot
3	#yr	#illuminations
4	#fete	#fcnantes
5	#lol	#blind
6	#twitter	#handisport
7	#fdl2015	#di
8	#love	#monlyon
9	#fourviere	#prayforabetterworld
10	#hommageauxvictimes	#lumiere

Discussion

There is no slicing of the data here for the clustering, therefore there are 700 tweets to cluster for this corpus. It is therefore probably the simplest corpus to cluster. Indeed, when having more tweets to cluster, there is more chance of being wrong.

According to the rand index value, the Word2vec representation seems to be better than the Doc2vec representation. Actually, regarding the Table 7.12, the Word2vec representation allows to find almost all the tweets (568/578) which are not talking about the event followed. Furthermore, the Word2vec representation also allows to find a big part of the tweets talking about the event followed. It seems to be more difficult for this representation to find the tweets talking about the "Fête des lumières". Indeed, there are 47 (FN), which are False Negative, which means that 47 tweets were True (talking about the event) and were predicted False (not talking about the event).

For the TF-IDF representation, according to the Table 7.11, it seems that the clustering is not so bad because it allows to find the tweets talking about the event followed (with a high recall). However, having a look to Table 7.12, the TF-IDF representation is very bad because it made two clusters : one with only 1 tweet and another with 699 tweets. Actually, the TF-IDF representation clusters all the tweets together.

According to the Table 7.12, we can see for all the representations that there are a lot of errors with the False Positive (FP) labels, which means that it was predicted true (talking about the event), while it was false (not talking about the event).

Finally, as opposed to the classification, the clustering clusters not so good the tweets. Indeed, according to the Table 7.13, there are only 3 #hashtags who are related to the event followed for the Doc2vec representation and only 4 #hashtags who are related to the event followed for the Word2vec representation.

7.2.2 Clustering on 7000 tweets

Evaluation

For this clustering on 7000 tweets, there are 1048 tweets talking about the event ("Fête des lumières") and 5952 tweets which are noise for the process.

Extraction

As presented in the section 3.1, the initial request for collecting the corpus was with these following keywords #Lyon, #FeteDesLumieres2015, #FDL2015 and #candle. With this clustering, it would have been possible to improve, (to enrich) the request with the following #hashtags: #lumignons and #hom-

Table 7.14: Results of the clustering on 7000 tweets.

Model	Precision	Recall	F1	Rand index	nmi
D2V worst one	17,13%	55,43%	26,17%	50,19%	0,31%
D2V best one	15,27%	96,76%	26,37%	69,07%	0,23%
W2V worst one	14,98%	82,82%	25,37%	60,55%	0%
W2V best one	15%	99,71%	26,08%	73,96%	0,04%
TF-IDF	14,97%	100%	26,05%	74,52%	0,01%

Table 7.15: Confusion matrix on 7000 tweets.

D2V worst one		False (GT)	True (GT)	
	False (predicted)	3141 (TN)	467 (FN)	3608
	True (predicted)	2811 (FP)	581 (TP)	3392
		5952	1048	7000
D2V best one		False (GT)	True (GT)	
	False (predicted)	324 (TN)	34 (FN)	358
	True (predicted)	5628 (FP)	1014 (TP)	6642
		5952	1048	7000
W2V worst one		False (GT)	True (GT)	
	False (predicted)	1024 (TN)	180 (FN)	1204
	True (predicted)	4928 (FP)	868 (TP)	5796
		5952	1048	7000
W2V best one		False (GT)	True (GT)	
	False (predicted)	32 (TN)	3 (FN)	35
	True (predicted)	5920 (FP)	1045 (TP)	6965
		5952	1048	7000
TF-IDF		False (GT)	True (GT)	
	False (predicted)	1 (TN)	0 (FN)	1
	True (predicted)	5951 (FP)	1048 (TP)	6999
		5592	1048	7000

mage. (taken from the Table 7.16). For the clustering with the Word2vec representation, there are differences in the top 10 #hashtags extraction which are : #fetedeslumieres and #8decembre.

Table 7.16: Top 10 hashtags extraction from the clustering on 7000 tweets.

	Hashtags FDL (D2V)	Hashtags FDL (W2V)
1	#valls	#fetedeslumieres
2	#cauchemarchezlecoiffeur	#8decembre
3	#fn	#france
4	#it	#valls
5	#lumignons	#cauchemarchezlecoiffeur
6	#hommage	#non
7	#lrds	#c
8	#chien	#merci
9	#car	#mdr
10	#rt	#fn

Discussion

As explained before, there is no slicing of the data here for the clustering, therefore there are 7000 tweets to cluster for this corpus. It is therefore more difficult to cluster 7000 tweets than 700. Indeed, there is more chance of being wrong. Also, since the corpus will be larger it is expected that the quality of the results decreases. According to the Table 7.14 and Table 7.15, it seems that the results are less good than the results for the clustering of 700 tweets (Table 7.11 and Table 7.12).

According to the rand index value, the Word2vec representation seems to be better than the Doc2vec representation. Actually, regarding the Table 7.15, the Word2vec representation allows to find almost all the tweets (1045/1048) talking about the event followed. However, the Word2vec representation seems bad to cluster the noise. Indeed, there are 5920 (FP) which are False Positive, that means that 5920 tweets were false and were predicted true. Moreover, regarding the cluster distribution there are on cluster with 35 tweets and one with 6965 tweets.

The same thing for the TF-IDF representation, according to the Table 7.14, it seems that the clustering is almost good because it allows to find the tweets talking about the event followed. However, having a look to Table 7.15, the TF-IDF representation is very bad because it made two clusters : one with only 1 tweet and another with 6999 tweets.

According to the Table 7.15, we can see for all the representations that there are a lot of errors with the False Positive (FP) labels, which means that it was predicted true, while it was false. Furthermore, we can see that there are not a lot of errors for the False Negative (FN) label. It seems that the clustering is able to find all the tweets talking about the event followed which is the "Fête des lumières" because it clusters almost all the tweets in the same cluster. The clustering is therefore not good.

Finally, for the extraction of the top #hashtags, the accuracy of the clustering on 7000 tweets is worse than the clustering on 700 tweets or than the classification. Indeed, according to the Table 7.16, there are only 2 #hashtags who are related to the event followed for the Doc2vec representation and only 2 #hashtags who are related to the event followed for the Word2vec representation. This results were expected because with a bigger corpus, it is expected to have more errors proportionally.

7.2.3 Clustering on 31000 tweets

Evaluation

For this clustering on 31000 tweets, there are 1048 tweets talking about the event ("Fête des lumières") and 29958 tweets which are noise for the process.

Table 7.17: Results of the clustering on 31000 tweets.

Model	Precision	Recall	F1	Rand index	nmi
D2V worst one	3,81%	56,20%	7,13%	50%	0,04%
D2V best one	3,47%	94,94%	6,69%	81,15%	0,05%
W2V worst one	3,56%	91,79%	6,85%	73,72%	0,07%
W2V best one	3,39%	100%	6,55%	93,03%	0,08%
TF-IDF	3,38%	100%	6,55%	93,23%	0,05%

Table 7.18: Confusion matrix on 31000 tweets.

D2V worst one		False (GT)	True (GT)	
	False (predicted)	15083 (TN)	459 (FN)	15542
	True (predicted)	14875 (FP)	589 (TP)	15464
		29958	1048	31006
D2V best one		False (GT)	True (GT)	
	False (predicted)	2271 (TN)	53 (FN)	2324
	True (predicted)	27687 (FP)	995 (TP)	28682
		29958	1048	31006
W2V worst one		False (GT)	True (GT)	
	False (predicted)	3862 (TN)	86 (FN)	3948
	True (predicted)	26096 (FP)	962 (TP)	27058
		29958	1048	31006
W2V best one		False (GT)	True (GT)	
	False (predicted)	73 (TN)	0 (FN)	73
	True (predicted)	29885 (FP)	1048 (TP)	30933
		29958	1048	31006
TF-IDF		False (GT)	True (GT)	
	False (predicted)	40 (TN)	0 (FN)	40
	True (predicted)	29918 (FP)	1048 (TP)	30966
		29958	1048	31006

Extraction

As presented in the section 3.1, the initial request for collecting the corpus was with these following keywords #Lyon, #FeteDesLumieres2015, #FDL2015 and #candle. With this clustering, it would not have been possible to improve, (to enrich) the request. Indeed according to the Table 7.19), there is no #hashtag related to the event followed which is the "Fête des lumières".

Table 7.19: Top 10 hashtags extraction from the clustering on 31000 tweets.

	Hashtags FDL (D2V)	Hashtags FDL (W2V)
1	#gautierconquet	#on
2	#archinfo	#lyon
3	#profdephilo	#c
4	#moijevote	#fn
5	#pitie	#non
6	#tameimpala	#france
7	#sorry	#love
8	#junglechristmas	#mdr
9	#ggmu	#merci
10	#doranaconda	#ldrs

Discussion

For this clustering, there are 31000 tweets to clusters, it is therefore more difficult to cluster 31000 than 7000 tweets or 700. Indeed, there is more chance of being wrong. Also, since the corpus will be larger it is expected that the quality of the results decreases. According to the Table 7.17 and Table 7.18, it seems that the results are less good than the previous results.

According to the rand index value, the Word2vec representation seems to be better than the Doc2vec representation. Actually, regarding the Table 7.18, the Word2vec representation allows to find all the tweets (1048/1048) talking about the event followed. However, the Word2vec representation seems bad to cluster the noise. Indeed, there are 29885 (FP) which are False Positive, that means that 29885 tweets were false and were predicted true. Moreover, regarding the cluster distribution there are on cluster with 73 tweets and one with 30933 tweets (the rest).

The same thing for the TF-IDF representation is observed, according to the Table 7.17, it seems that the clustering is almost good because it allows to find the tweets talking about the event followed (and all of them). However, having a look to Table 7.18, the TF-IDF representation is very bad because it made two clusters : one with only 40 tweet and another with 29958 tweets. Moreover, the big cluster is the one which are considered as the one talking about the event followed.

According to the Table 7.18, we can see for all the representations that there are a lot of errors with the False Positive (FP) labels, which means that it was predicted true (talking about the event followed), while it was false (not talking about the event followed). Furthermore, we can see that there are not

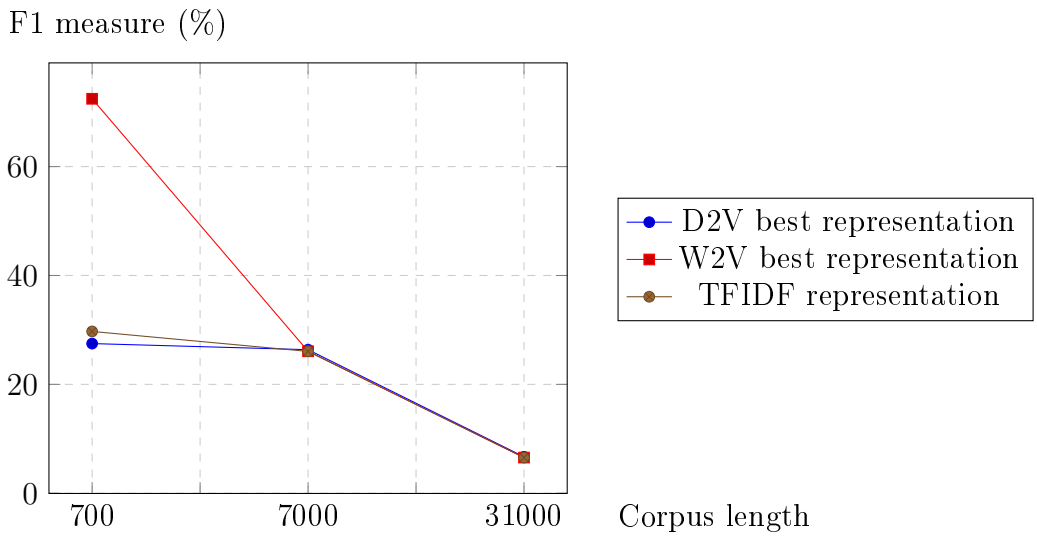
a lot of errors for the False Negative (FN) label. It seems that the clustering is able to find all the tweets talking about the event followed which is the "Fête des lumières" because it clusters almost all the tweets in the same cluster. The clustering is therefore not good.

Finally, for the extraction of the top #hashtags, the accuracy of the clustering on 31000 tweets is the worst one. Indeed, according to the Table 7.19, there are 0 #hashtags who are related to the event followed for the all the representation This results were expected because with a bigger corpus, it is expected to have more errors proportionally.

7.2.4 Discussion about the clustering of the whole corpus

Corpus	D2V	W2V	TFIDF
700	27.49	72.46	29.72
7,000	26.37	26.08	26.05
31,000	6.69	6.55	6.55

Table 7.20: Table of the best representations with the clustering according to the F1 measure.



The chart shows therefore the best Doc2vec, Word2vec and TF-IDF representations according to the F1 measure.

Figure 7.2: Curves of the best representations with the clustering according to the F1 measure.

According to the Table 7.1, Table 7.4, Table 7.7 and the Figure 7.2, we can see that the quality of the clustering decreases as the cluster size increases. This was expected since it is harder to cluster more tweets. From the corpus

containing 700 tweets to the one containing 7000 tweets the results slowly decrease. Furthermore the difference between the corpus containing 7000 tweets and the largest one (that is to say 31000 tweets) is huge. Indeed, it goes from an average of 26% for the f1 measure to an average of 6%.

For all the representation, a high recall and a low precision are observed. Indeed, the high recall comes from the fact that there is a small number of False Negative (FN) that is to say that the process find almost always all the tweets talking about the event followed. Moreover, the low precision can be explained by the fact that there is many False Positive (FP) that is to say that the process confuses the negatives (not talking about the event) with the positives (talking about the event). As observed in the Table 7.2, Table 7.5 and Table 7.8, the process often finds two clusters that have totally disproportionate sizes (not balanced sizes). Indeed, it is logical to have a high recall with one big cluster because all the positive tweets (talking about the event) are inevitably there.

The rand index is a good measure in order to evaluate a clustering when the label are not important. Indeed, the rand index measure allows to see if the process find good clusters without worrying about labels (classes). However, with the type of clustering that we observe, the rand index is not discriminant because almost all the tweets are in the same cluster. However, here the nmi measure allows to see that the clustering is bad.

For all the corpus, it has been observed that the worst Doc2vec representation make two balanced clusters with almost the same number of elements for each cluster. Moreover, the TF-IDF representation seems to be very bad as it clusters the data into two cluster. In reality it just make one cluster and another with almost any tweet. According to Table 7.2 and Table 7.5, there is only one tweet in one cluster and all the other tweets in the other cluster. The recall is good but the precision is bad. For the extraction step, according to the Table 7.13, Table 7.16, and Table 7.19, the best extraction is done with the smallest corpus containing 700 tweets. Indeed, there are 3 #hashtags which are good in order to improve the initial request.

Finally, it seems to have a best representation for the Doc2vec and Word2vec parameter variations. Indeed, it has been observed for the Doc2vec representation with a vector length of 70, a min word frequency between 5 and 8, a net iteration of 5, a layer size of 200 and a window size of 5 that gives good result. Concerning the Word2vec representation, it is not so easy to find a correlation

between the paramters, but a vector length of 70, a min word frequency of 5 and a net iteration of 50 gives good result.

Accoding to all the results from the classification part and the rand index, it is possible to assume that the best representation if the Word2vec one. However, taking a look to the confusion matrices (Table 7.2, Table 7.5 and Table 7.8) it seems that Doc2vec representation is much better because it allows to find more False Positive tweets (tweets which are not talking about the event) than the Word2vec representation. The Word2vec representation is certainly less performing to find True Positive but it allows to find a lot of False Positive. That is a good balanced representation.

8 Perspectives

There are a lot of perspectives and possibilities concerning this wide subject. First of all, it could be interesting to try to improve the classification through the metadata concerning the tweets as :

- The timestamp;
- The geo-location;
- The popularity of the tweet (with the number of time the tweet have been re-tweeted);
- The number of followers of the user which has post the tweet (his influence);
- The total number of message posted by the user;
- The language of the user and/or of the tweet.

Furthermore, as the media as the images, the videos are also saved, the data from the picture could also be integrated for helping the clustering or the classification. -> "Content-based image retrieval". For example, it could be interesting to get images which are close to the tweet's image and to extract the #hashtags they have. Then it would be possible to enrich the bag of words with this new #hashtags discovered.

Moreover, it could be also interesting to have a look on the geo-location of each message talking about the event followed. Indeed, as explained in the subsection 4.1.3, the data collectors can have some location restrictions with a point and a radius or four points delimiting a zone. If most of the messages talking about the subject are near of a border, it's maybe that there are tweets posted outside of the geographical area and that the collector is missing a lot of posts.

It could be also interesting to use external content. For example, as shown in the chapter 3, there are a lot of messages containing at least a link. It could be envisaged to retrieve the tags "title" and "meta keywords" from the site by doing a simple query on it and to add the token into the tweets.

As presented in the chapter 7, the clustering with kmeans doesn't seem to

perform well. Furthermore, as the classification with the linear kernel works well, a clustering with kmeans should work well or even better than it is now. It would be interesting to dig two tracks. On the one hand it could be good to generate more Word2vec and Doc2vec models by varying all the parameters. It would be great to vary the parameters with really significant values. Furthermore, it would also be interesting to modify the test corpus. Indeed, the number of elements of the corpus seems to play a role, since as has been seen in Figure 7.1, the results decrease proportionally to the increase of element in the corpus.

Furthermore, as the main goal is to execute the entire process in real time in order to keep a relevant bag of words which evolves over the times, a clustering as DBScan seems more appropriate. According to the [27], the DBScan algorithm seems to be a good clustering algorithm in order to evolve over time. That is to say, to not recompute all the data, when some new data come.

Finally, most of the time, cities have their own events website and publish explanations and images about past or future events. For example with (the "Oktoberfest" [16] in Munich, or the "Passauer Dult" [14] in Passau). It might be smart to get the text and try to get out keywords in order to use then as a base keywords for the beforehand bag of words. Increasingly, the well-known events have their own keywords so it is possible to follow them on social networks (it also can be a precious base for the bag of words).

9 Conclusion

In this research, the main goal was to enrich a bag of words used for querying the API of social networks in order to retrieve as effectively as possible the most possible messages from social networks talking about a specific event. For this goal, it is hooked several research questions. Indeed, as presented before there is a heterogeneity in the data and almost all messages from social networks are hollow and don't contain context. Moreover, the data is large, noisy and constantly evolving. It was therefore necessary to maintain relevant bag of words that evolves over time depending on the activities of social networks. It is sometimes difficult for humans to classify a tweet belonging to a subject or another (because it is not easy, it is subjective and depends on the person who do the classification), so for the process it was necessary to try to improve the content of messages from social networks.

Furthermore, the data clustering step is not possible on textual data, so a numerical representation was necessary. For that, the choice was made to try three numerical vector representations (Word2vec, Doc2vec and TF-IDF). The goal was therefore to try to find the best numerical vector representation for a tweet. For that, several parameters variations have been made for the Word2vec and Doc2vec representation.

Then, several clustering algorithms have been pointed in order to see the advantages and disadvantages of each. The clustering with K-means has been tested for the experiments. It was also interesting to try to compare a classification way to "cluster", to classify the data and a clustering algorithm. According to the results presented in the chapter 7, the classification performed better than the clustering with the K-means algorithm. The results were expected because both approaches do not perform in the same ways.

For the classification task, the Word2vec representation seems to be the best one with 100% of precison, 100% of recall, 100% of F1, 100% of rand index and 100% of nmi for both biggest corpus (Table 7.5 and Table 7.8). Indeed, there are no error in the classification. In addition, the RBF kernel does not seem to be adapted to the data as it provides less good results. Indeed, it

seems that the data is linearly separable. Furthermore, it is interesting to see that the results increase when the number of ground truth elements increase (Figure 7.1). That is to say, that for the classification task, more there are ground truth elements for the training step, more the model will be accurate.

Concerning the clustering, it is totally not the same results. Indeed, there is no training step, the process try to cluster the elements according to the Kmeans algorithm. According to the results (Table 7.12, Table 7.15 and Table 7.18), it seems that the kmeans algorithm does not perform as well as the classification. The results were expected, because nowadays the machine learning techniques tend to give better results for almost all the applications. Furthermore, it is possible that the parameters variations made for the project are not discriminant for Kmeans. Almost all the parameters for the Word2vec and the Doc2vec representations have been tested expect the min learning rate, the batch size and the sub sampling. For this three parameters the default values have been kept. As presented in the chapter 8, it would be interesting to generate more models.

Finally, as discussed throughout this document, clustering messages from social networks is not an easy task. Indeed, there is too much heterogeneity in the data to make the task easy. Although it does not work great for clustering, it does not seem to be a problem for the Support vector machine.

The results for the classification are very satisfactory and thus proves that all the layers developed for the process seems to be correct. However, it would be good to deepen the experiments regarding the Kmeans clustering to find a good representation for the tweets. Indeed, the tweet's representation is not unique, it depends on the corpus, on the algorithm, on the number of tweets. Since the Word2vec, Doc2vec and TF-IDF models are context dependent, it must distinguish the best representation for each algorithm, each corpus, etc.

10 Acknowledgements

As explained before, this project (master thesis) is an end of study project ("Masterarbeit", “PFE") in the context of double master degree with the University of Passau in Germany and INSA Lyon (National Institute of Applied Sciences) in France. This double master degree has two master degree programs : a degree in computer science engineering at the INSA (National Institute of Applied Sciences) in Lyon and a Master in Informatik (Schwerpunkt : Information und Kommunikationssysteme) at the University of Passau.

First of all, I would like to thanks the initiators of the double Master IFIK[1] for their valuable efforts and commitment to the german-french collaboration : Prof. Dr. Harald KOSCH for the german side and Prof. Dr. Lionel BRUNIE for the french one.

I would especially like to thank my two supervisors. Prof. Dr. Michael GRANITZER for the german side and Dr. Habil. Elöd EGYED-ZSIGMOND for the french side.

I would like to thank Dr. Habil. Elöd EGYED-ZSIGMOND for its follow during my specific project in France, my bibliographic synthesis in France, as well as his weekly monitoring throughout my double degree in Germany. Moreover, I would like to thank him for the time he spent guiding me, for his precious help and advices during almost 2 years.
I would like to thank Prof. Dr. Michael GRANITZER for its follow during my double degree in Germany. Moreover, I would like to thank him for the time he spent helping me, for the meetings we made, for his precious advice during the entire project.

I would also like to thank Mrs Morwenna JOUBIN for all Franco-German courses she taught us and her good humor and kindness.

Finally, I thank all the people of the chair of Prof. Dr. Harald KOSCH for their good humor, their kindness and their advice during my initial presentation.

[1] https://www.dimis.fim.uni-passau.de/MDPS/de/ifik-doppelmaster.html

11 Bibliography

[1] About twitter - website. https://about.twitter.com/company. Accessed: 2016-08-13.

[2] Dbscan - website. https://en.wikipedia.org/wiki/DBSCAN. Accessed: 2016-06-12.

[3] Deeplearning4j (word2vec) - website. http://deeplearning4j.org/word2vec.html. Accessed: 2016-04-05.

[4] Doc2vec - website. https://radimrehurek.com/gensim/models/doc2vec.html. Accessed: 2016-05-20.

[5] Facebook - website. https://www.facebook.com/. Accessed: 2016-03-18.

[6] Flickr - website. https://www.flickr.com/. Accessed: 2016-03-18.

[7] Hierarchical clustering - website. https://en.wikipedia.org/wiki/Hierarchical_clustering. Accessed: 2016-06-13.

[8] IDENUM project - website. http://imu.universite-lyon.fr/qui-sommes-nous/projet/. Accessed: 2016-08-24.

[9] Instagram - website. https://www.instagram.com/. Accessed: 2016-03-15.

[10] K-means clustering - website. https://en.wikipedia.org/wiki/K-means_clustering. Accessed: 2016-06-13.

[11] Mongodb - website. https://www.mongodb.com/. Accessed: 2016-03-18.

[12] My github page - website. https://github.com/afaraut. Accessed: 2016-06-26.

[13] Normalized mutual information - website. http://nlp.stanford.edu/IR-book/html/htmledition/evaluation-of-clustering-1.html. Accessed: 2016-07-20.

[14] Oktoberfest - website. http://www.oktoberfest.de/en/. Accessed: 2016-09-18.

[15] Optics clustering - website. https://en.wikipedia.org/wiki/OPTICS_algorithm. Accessed: 2016-06-13.

[16] Passauer dult - website. http://www.passauer-dult.de/. Accessed: 2016-09-18.

[17] Pinterest - website. https://www.pinterest.com/. Accessed: 2016-03-18.

[18] Rand index - website. https://en.wikipedia.org/wiki/Rand_index. Accessed: 2016-07-20.

[19] Twitter - website. https://twitter.com. Accessed: 2016-03-15.

[20] Twitter api - website. https://dev.twitter.com/rest/public/search. Accessed: 2016-04-20.

[21] Twitter api (entities in objects) - website. https://dev.twitter.com/overview/api/entities-in-twitter-objects. Accessed: 2016-04-25.

[22] Twitter4j api - website. http:/twitter4j.org/en/index.html. Accessed: 2016-04-20.

[23] Word2vec - website. https://code.google.com/p/word2vec/. Accessed: 2016-04-05.

[24] Word2vec capitals - website. http://deeplearning4j.org/img/countries_capitals.png. Accessed: 2016-09-03.

[25] a.F. McDaid, D. Greene, and N. Hurley. Normalized Mutual Information to evaluate overlapping community finding algorithms. *Arxiv preprint arXiv:1110.2515*, pages 1–3, 2011.

[26] U. Brandes, D. Delling, M. Gaertler, R. Görke, M. Hoefer, Z. Nikoloski, and D. Wagner. On Finding Graph Clusterings with Maximum Modularity. In *Graph-Theoretic Concepts in Computer Science*, volume 4769, pages 121–132. 2007.

[27] F. Cao, M. Ester, W. Qian, and A. Zhou. Density-based clustering over an evolving data stream with noise. *Proceedings of the Sixth SIAM International Conference on Data Mining*, 2006:328–339, 2006.

[28] Y. Han, H. Lee, and Y. Kim. A real-time knowledge extracting system from social big data using distributed architecture. In *Proceedings of the 2015 Conference on research in adaptive and convergent systems - RACS*, pages 74–79, New York, New York, USA, 2015. ACM Press.

[29] G. Ifrim, B. Shi, and I. Brigadir. Event Detection in Twitter using Aggressive Filtering and Hierarchical Tweet Clustering. In *SNOW-DC@WWW*, pages 1–8, 2014.

[30] Q. Le and T. Mikolov. Distributed Representations of Sentences and Documents. *International Conference on Machine Learning - ICML 2014*, 32:1188–1196, may 2014.

[31] C. Li, J. Weng, Q. He, Y. Yao, A. Datta, A. Sun, and B.-S. Lee. TwiNER. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval - SIGIR '12*, page 721, New York, New York, USA, aug 2012. ACM Press.

[32] H. Mousselly Sergieh, E. Egyed-Zsigmond, G. Gianini, M. Döller, J.-M. Pinon, and H. Kosch. Tag Relatedness in Image Folksonomies. *Document numérique*, 17(2):33–54, oct 2014.

[33] O. Ozdikis, P. Senkul, and H. Oguztuzun. Semantic Expansion of Tweet Contents for Enhanced Event Detection in Twitter. In *2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 20–24. IEEE, aug 2012.

[34] H. S. Packer, S. Samangooei, J. S. Hare, N. Gibbins, and P. H. Lewis. Event Detection Using Twitter and Structured Semantic Query Expansion. In *Proceedings of the 1st International Workshop on Multimodal Crowd Sensing*, pages 7–14, New York, New York, USA, nov 2012. ACM Press.

[35] S. Romano, J. Bailey, N. X. Vinh, and K. Verspoor. Standardized Mutual Information for Clustering Comparisons: One Step Further in Adjustment for Chance. *Proceedings of The 31st International Conference on Machine Learning*, 32:3, 2014.

[36] X. Rong. word2vec Parameter Learning Explained. *arXiv:1411.2738*, pages 1–19, nov 2014.

[37] S. Tuarob, W. Chu, D. Chen, and C. S. Tucker. TwittDict: Extracting Social Oriented Keyphrase Semantics from Twitter. In *Proceedings of the International Workshop on Novel Computational Approaches to Keyphrase Extraction.*, pages 25–31, 2015.

Erklärung zur Masterarbeit - Declaration of Authorship

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Die Arbeit wurde bisher keiner anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

_____

I hereby declare that the thesis submitted is my own unaided work. All direct or indirect sources used are acknowledged as references.

This paper was not previously presented to another examination board and has not been published.

Passau, October 06, 2016

Anthony FARAUT

Appendices

Exemple of JSON returned by the Twitter API


```
{
    "created_at" : "Tue Dec 08 20:27:13 +0000 2015",
    "geo" : {
        "coordinates" : [45.7333000000000000, 4.9166699999999999
            ], "type" : "Point"
    }, ...
    "id_str" : "674324371742302213",
    "text" : "A lilttle light in the night. ??? \#LightsParty \#
        Lyon \#PrayforParis https://t.co/V45Wn5Ngkf",
    "lang" : "en",
    "timestamp_ms" : "1449606433221",
    "entities" : {
        "urls" : [
            {"indices" : [68, 91], "url" : "https://t.co/
                V45Wn5Ngkf"}
        ],
        "hashtags" : [
            {"indices" : [34, 46], "text" : "LightsParty"},
        ], ...
    },
    "user" : {
        "friends_count" : 1159,
        "favourites_count" : 2256,
        "description" : "One Direction aholic...",
        "created_at" : "Fri Jan 20 23:23:40 +0000 2012",
        "screen_name" : "hazz_aholic",
        "id_str" : "469756268",
        "lang" : "fr",
        "statuses_count" : 6214,
        "followers_count" : 416, ...
    }
}
```