Imperial College London

Department of Computing

Deep Learning for Emotion Analysis

Author:
Valentin Chelle

Supervisor:
Dr. Anandha Gopalan

A thesis submitted for the degree of

MSc Advanced Computing

October 7, 2019

Abstract

Social networks are becoming a huge part of current human interactions. People can publish publicly their thoughts, feelings and emotions. Such emotions, easily understandable by humans, represent a valuable data. Extracting emotions from users' messages can lead to the analysis of the evolution of emotions over time for a given user. It could then be used to predict the emotion of a future message, detect ups and downs of a user and detect depressive periods. The study is based on Twitter since the system of hashtags can be used to generate a dataset quite efficiently. The project presents simple models and algorithms in order to show how an analysis of emotions can be done on small servers. For instance, our work has been used to create a web platform that gives a way for everyone to analyze the evolution of the emotions on their Twitter account. We believe that such work can be used by social networks to prevent or counter depression by simply analyzing the evolution of the emotion of their users and applying some emergency procedures.

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Dr. Anandha Gopalan, for his excellent guidance, support and flexibility throughout this project. I thank my family for their unwavering support and encouragement throughout my studies. I thank Gabriella Morales for her support and for proofreading this paper.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Social networks are a consistent part of the everyday life. An average of two hours are spent daily on social networks in the world[10]. Through those platforms, people are interacting with each other publicly by posting pictures, messages and other types of medias. Most users are expressing their thoughts and their feelings, and those websites represent a great way to get mental support, by the others users, but it can also bring the moral down. Understanding the emotions expressed by the users of those social networks is becoming important in order to avoid a downward spiral of mental health occurring in the case of a depressed user.

Extracting emotions from messages on social networks is a real challenge. The problem in itself is not easy since it is even hard for humans to analyze someone else's emotions. Fully understanding the emotion of messages requires being able to grasp obvious emotions like hate, happiness or sadness, but also requires detecting subtleties like irony or humor. This study stays within the limits of the emotions happiness, sadness, neutral, hate, surprise and fun. The construction of the dataset will be done while trying to avoid humor and irony. The motivation of this project is to first analyze the emotions in the messages of a given user. Then, the study of the evolution of those emotions, over time, for a given user is used to detect depression.

Twitter is a famous social networks (262 millions of users in 2018 [11]), on which people are expressing a lot of their thoughts. Their emotions are often highlighted by emoticons and hashtags that make this source of data very interesting. The dataset to train our machine learning algorithms will be built based on that. Twitter also allows us to retrieve easily tweets of a given user (via their API). Our final platform will help us to predict the emotion of the messages of a given user and display their evolution over time. Such a platform will allow people to analyze their emotions, give us feedback to improve our system, and be concerned about depression. This can also be used to analyze some famous accounts, such as political accounts, and analyze the way, emotionally speaking, they are delivering their messages on Twitter.

1.2 Modeling Emotions

1.3 Challenges

The blurry line between emotions

Putting a name on feelings is not always easy. Understanding others' emotions is an everyday challenge for humans and in many cases there is not even a proper answer. It is a problem where understanding the context definitely plays an important role. While in reality the context required to understand the emotion can be of an unlimited size, the memory limit on computers make us unable to retrieve such an unlimited context. We have to deal with a limited context which is the first thing that makes some emotions indistinguishable for our system.
Irony and humor are also two factors that can heavily alter the performances. This will also create noise in our dataset, especially since they are formed via Twitter.

Aim to be deployed online

This project aims to give an access to the classifier online. It requires to limit the capacity of the model. A system too complex wouldn't be easily deployed on a web server and wouldn't be serving a result in a relatively small amount of time. Users expect a quick response from the server, so we decided to make our system simple enough to give results in less than 30 seconds, making compromises on the complexity of the models.

Chapter 2

Background

2.1 GloVe Embedding

GloVe [3] is a project that offers embeddings for more than one million words in several languages. GloVe Twitter is a branch of the project where the embeddings have been trained based on tweets. For this project we have chosen to use the embedding with a hundred features. This choice allows us to have a complex embedding matrix while staying light which is important in our front-end delivery system.

2.1.1 A Naive Approach: the One-Hot Encoding

Word Embedding is a common technique in natural language processing. It places the words in a vector space with a dimension quite smaller than the dimension of the vocabulary. Indeed, a naive vector space for words is the one hot encoding where, for a word in a vocabulary of 10,000 words, v_{word} the one hot encoded vector, representing the word chosen, follows the rule : $v_{word[i]} = 1$ if vocabulary[i] $=$ word, $v_{word[i]} = 0$ else. The advantage of the one hot encoding is its being extremely simple to compute. The inconvenient of this approach is that the dimension of the embedding depends on the vocabulary size, which can be enormous.

2.1.2 Embeddings and Emotions

The GloVe Embedding allows us to represent the words in a space of only a hundred dimensions. And the similar words (words than appeared often together in the corpuses used to train the embedding) will be close distance-wise in this vector space. It is very interesting for our project to have words associated to a similar emotion close from each other in that space.
The plot below, where x is the first feature of the embedding and y the fourth, shows that words associated to different emotions can be separated quite easily even on only two dimensions. We chose words that refer to two really different emotions: hate and love.

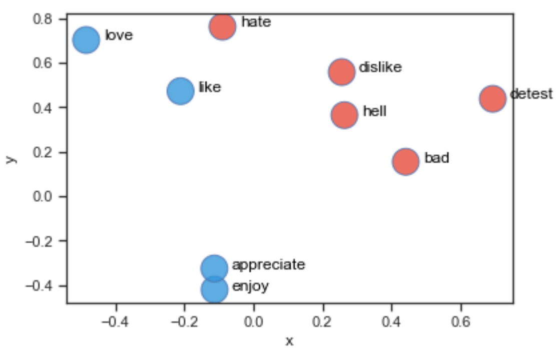

Figure 2.1: Embeddings represented on dimensions 1 and 4

2.2 Understanding the Emotion Representation

There are many approaches to understanding emotions, and we chose one that is widely spread in emotion analysis. In their study, Pete Trimmer *et al.*[12] presents a way to represent emotions on two dimensions, the arousal and the valence. Here is the representation of those emotions.

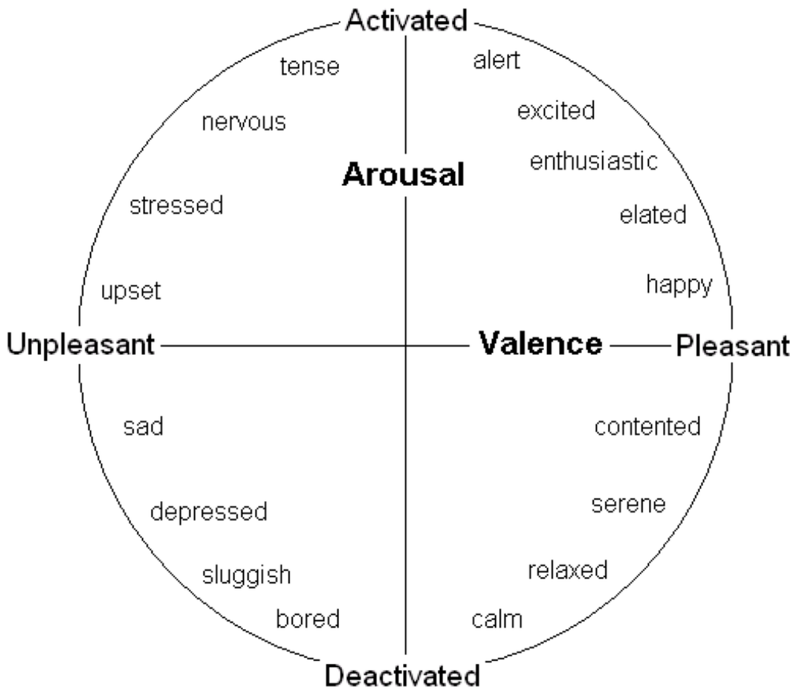

Figure 2.2: Arousal Valence Model for Emotions

Here is an idea of how the emotions can be visualized together. Their work is interesting when it comes to pick the emotions we want to classify. Classifying emotions means being able to distinguish them. Taking emotions too similar (i.e close on the graph) would not give interesting results and would be a problem too hard for our deep learning models. For instance, if we choose to classify both the emotions sad and sluggish, it would be very hard to tell the emotions of a message like "I don't feel good".

It is critical to not choose emotions that are too close. However, it is also very important to cover the whole spectrum of emotions. For instance, it would be impossible to understand the emotions of the users with only the two emotions *bored* and *tense*. It is then recommended to pick at least one emotion in each quarter of the Arousal Valence model.

The strength of this representation is the arousal axis. Considering the fact that we need to create a dataset for each of those emotions, the emotions must be active enough to be identifiable in the messages of the users. For example, creating a dataset for the emotion serene is harder than creating a dataset for the emotion happy, since a user would usually express their happiness much more than they would express their serenity. Also, messages can contain more than one emotion. Choosing mainly active emotions might avoid some conflicts between an active emotion that would attenuate the visibility of a less active emotion. The happiness would for example attenuate the serenity.

The difficulty of picking emotions to classify is getting the closest as possible to the lower quarters (close to the deactivated emotions) while staying in the upper quarter. We also consider two emotions that are not on the model: love and neutral. Neutral would be the point at the intersection of the arousal and valence axis. Love, which is important in order to identify the emotions of people on social networks, would be somewhere close to the center, in the upper right quarter.

Finally, we use on the Arousal Valence model the following emotions that are easy to identify: sad, worried (similar to stressed and nervous), happy (standing for a mix between contented, serene, and happy according to the previous model) and fun (similar to excited and enthusiastic).

2.3 Neural Networks

In this section, we will present briefly the different structures we use in our models. These structures can have different usages such as text analysis, prediction from sequences and classification from sequences.

2.3.1 Long-Short Term Memory Neural Networks

Most of our networks are using Long-Short Term Memory (LSTM) cells. They provide a way to remember a certain quantity of past values. They are usually composed of three units : the input gate, the output gate and the forget gate. The input gate controls the flow of the new values by authorizing or blocking the update. The output gate controls whether the state is transmitted as the output or not. Finally, the forget gate can reset the state of the cell.

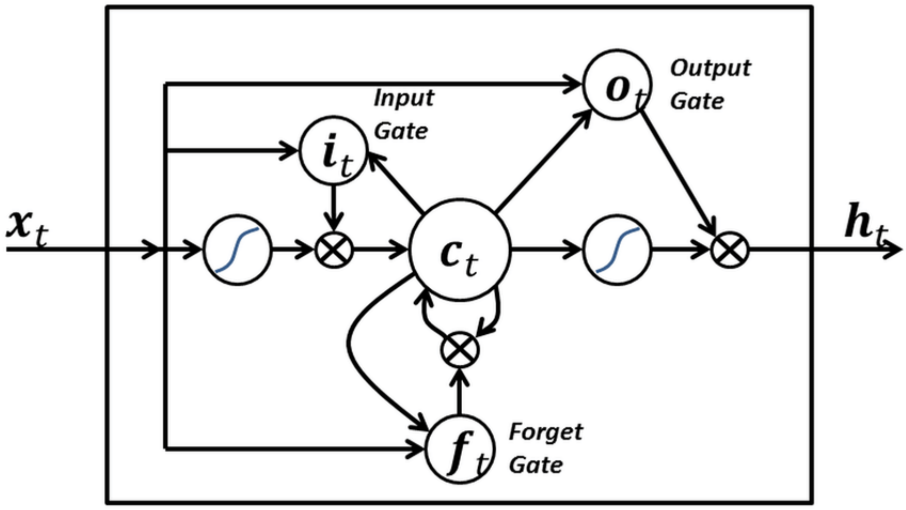

Figure 2.3: A simple LSTM gate with only input, output, and forget gates. [8]

The strength of the LSTM is exploited several times in our models. Using LSTM when doing some text analysis, as in our first part, will make the network able to take into account the words seen previously and thus, have a deeper understanding of the complex structure. For instance, the sentence "I do not hate you" would probably be classified as a hate message by a classic neural network. Since the LSTM cells can "remember" past values, it will remember the "do not" part, and then will probably be able to understand the negation.

2.3.2 Convolutional Neural Networks

The Convolutional Neural Networks (CNN) is a feed forward net where the connection between the neurons is inspired by the neural visual cortex of animals. It has been used mainly for image analysis and speech analysis. They have the ability to consider smaller and simpler patterns of an input instead of looking at the whole.
It is usually constituted of a succession of convolution layers and sub sampling layers (usually a pooling layer). The convolutional layer's main parameter is a set of filters (usually of a small size), that will be convolved across the input volume. The convolutional layer is constituted of the results of every convolution for each filter. The downsizing layer is used to reduce the dimension of the network after the convolutional layer.

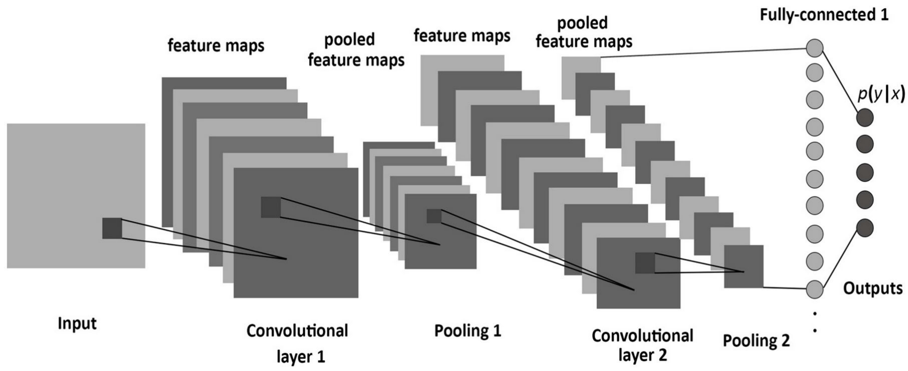

Figure 2.4: The structure of a CNN, consisting of convolutional, pooling, and fully-connected layers.[1]

The CNN is used in our case to detect underlying patterns and grasp the difference of context by looking at the surroundings of a word. The position of a word in a sentence doesn't really matter anymore, only the words around will be helpful for the network to grasp the meaning.

Chapter 3

Related work

3.1 Classification of Sentiments

A lot of work has been done in this field since it represents a real economic aspect for companies. For instance, a sentiment is what a customer thinks of a product. Many studies are elaborating ways to predict if a text is either *"positive"* or *"negative"* (binary sentiment classification) or *"positive"*, *"negative"* or *"neutral"* (tertiary sentiment classification). Pang *et al.* [5] classify a dataset of movie reviews into positive and negative sentiments.

In [9], M. Sosa describes a deep learning method, combining Convolutional Neural Network and Long-Short Term Memory Recurrent Neural Networks to classify a tweet as negative or positive.

3.2 Classification of Emotions

A study by Maryam Hasan *et al.* [2] presents really interesting aspects of the emotion classification. First, they give an interesting list of features they used: uni-gram features, emoticons features, punctuation features and negation features. Then, they use classifiers such as Naive Bayes, SVM and KNN and they reach an accuracy of 90%.

Purver *et al.* [7] use two interesting methods to generate their dataset. A first one is based on matching the hashtags with emotions and the other one uses simple emoticons *:(,:)*. We are going to use those methods and improve them to create a relevant dataset. They present interesting results about a survey, taken by more than 400 people, that aimed to understand which emotion is linked to a certain emoticon. While some emoticons are making consensus, some have a different meaning for different people. When generating our dataset, we prioritize the use of the emoticons which people have the same opinion about.

Timothy Liu [4] describes a deep learning multi-class emotion classification method that combines two different networks constituted of CNN and LSTM cells. It applies M. Sosa's work [9] to emotion multi-classification and reach an accuracy of more than 62%. That system is performing well, but it involves more than 6 million trainable parameters, mainly due to a trainable embedding cell. Hence, it can not be deployed efficiently on a web server.

Chapter 4

Dataset

Before even start to try to model emotions on Twitter, we gather data via the Twitter API. Two approaches for the data collection are studied. A first one collects and labels messages regarding their hashtags. A second one collects and labels messages regarding the emoticons they contain.

4.1 The hidden benefits of Twitter

Twitter provides a very useful tool for data collection: the advanced search API[13]. While most of the users use the basic search bar, where you can simply type words or hashtag, Twitter also offers search options allowing logical operators and filters on the requests. For instance, typing "*#funny OR #fun -filter:retweets -filter:links*" in the search will look for every tweet that contains the hashtag *#funny* or *#fun*, but will exclude the retweets and the tweets sharing links. This is our main method for the data collection with different combinations of filters and logic operators.

4.2 The Negation Problem

When it comes to extracting emotions, a very basic idea would be to search for specific words in tweets that would indicate a feeling, such as looking for the word *hate* in a tweet and labelling it as a sentiment of hate. But this solution is extremely limited by the negative form that most languages offer. "I do not hate you" is actually an elegant sentence meant to be a sentiment of love. That is why we can not really use this approach, especially on Twitter where negation is far from being the only type of language elaboration.

4.3 Tackling the Problem with the Hashtags

The choice of Twitter as the main source has been made considering the data collection. Its system of hashtags allows us to create an important dataset with a naive method. For instance, a tweet can be classified as a funny tweet if it contains the hashtags *#fun* or *#funny*. Collecting several tweets for a hashtag associated to a given emotion and then removing those hashtags from the tweets gives us a good dataset for almost any emotion we want.
This system is more robust to the negation problem since the only case where a hashtag would be used in a negation form would in a sentence itself such as : "*I do not #hate you*". However, it is less frequent since the common use of the hashtags is to place them at the end of the message. Users have interests to put accurate hashtags given the content of their tweet since it will be displayed in the corresponding category. The following table provides some examples of hashtags and the matching emotions.

The data collected contains a lot of noise and the hashtags have to be chosen wisely to avoid irony and other misunderstanding of the emotions. A simple counter example would be the hashtag *#hate* in "*big no to your hate, @user! #enough #hate*". Many tweets containing this hashtag are denouncing hate, instead of expressing a feeling of hate. One can think about ironic messages, where the hashtags would be the opposite of the real feeling of the message.

Relevant Hashtags	**Emotion**
#enjoy, #enjoylife, #relax, #laugh, #lovelife, #sohappy, #excited,#feelgood,#happier,#goodmood, #joy	Happiness
#sadness,#sad,#depressed,#alone,#lonely,#sadquotes, #depression,#anxiety,#crying,#upset,#lifesucks	Sadness

Table 4.1: Example of classification based on hashtags

4.4 Another Approach with the Emoticons

The hashtag system is still sensitive to irony and humor. Another approach studied is to label a message depending on the emoticons inside. In most of the cases, users will tend to use emoticons and those will be even more relevant than the hashtag when it comes to expressing an emotion. However, it could lead to some confusion between similar emotions. For instance, the emoticon heart ("<3") can be present in a happy tweet such as "*Finally done with my exams <3*" and in a love tweet such as "*I love you <3*". But this limit is not really problematic in our case. The nature of the problem itself makes it hard to differentiate some sentiments, and that kind of error is not limiting our application, where that kind of confusion would not lead to a big difference in the evolution of the emotion of the user through time. On the following table, some examples of the emoticons we used to label our data are given:

Relevant Emoticons	**Emotion**
:), :D, :p, :'), ;), ;'), :'D, :]	Happiness
:(, :/, :'(, ;(, ;'(, :[	Sadness

Table 4.2: Example of classification based on emoticons

After some data exploration over our newly formed dataset from emoticons, we discovered that a lot of tweets are actually not relevant. Indeed, some users have the tendency to use an excessive number of smileys in their messages. To avoid those messages, we limit the number of smileys in a same message to three.

4.5 The problem of the Actuality

When creating the dataset, the actuality can make the results biased. For instance, running the crawler during a football game would make the dataset mainly constituted of football related messages. To counter this problem, messages have to be extracted from different periods of the year. And to get genuine emotions, it might be better to avoid constituting the dataset when the date corresponds to a political event or any other world event.

4.6 Dealing with an Unbalanced Dataset

Our dataset is initially composed of 13 classes, which are: *anger, boredom, empty, enthusiasm, fun, happiness, hate, love,neutral, relief, sadness, surprise* and *worry*. However, it is unbalanced as shown on the following distribution of the dataset over the thirteen classes :

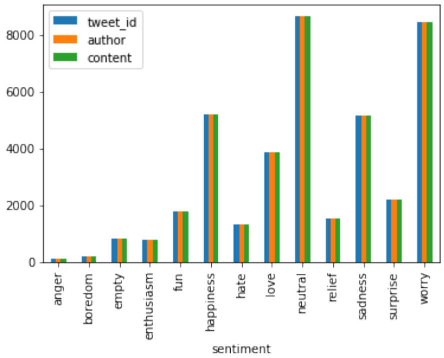

Figure 4.1: Initial distribution of the dataset

To address the unbalancing and to increase the relevance of the classes and their differences, some of those classes are being merged. For instance, *anger* and *hate* are merged into a unique class. Doing that, the problem has been reduced to six classes which are *fun, happiness, love, neutral, sadness, worry.*
Here is the distribution of the new dataset :

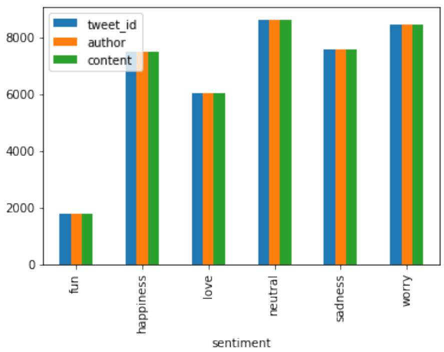

Figure 4.2: New distribution of the dataset

Additionally, during the training, weights will be applied to classes to deal with the unbalanced dataset.

Chapter 5

Emotion Multi Classification

5.1 General Idea

5.1.1 The Constraint of Deployment

The goal of the whole project is to load the train a model and serve a simple web page to users on which they can analyze twitter accounts using this trained model. This means that the complexity of our model is limited by that memory constraint of the web server.

5.1.2 The Pipeline

The following diagram explains how our architecture is organized and what the links are between our local machine on which the model is trained, and the server on which the model is used.

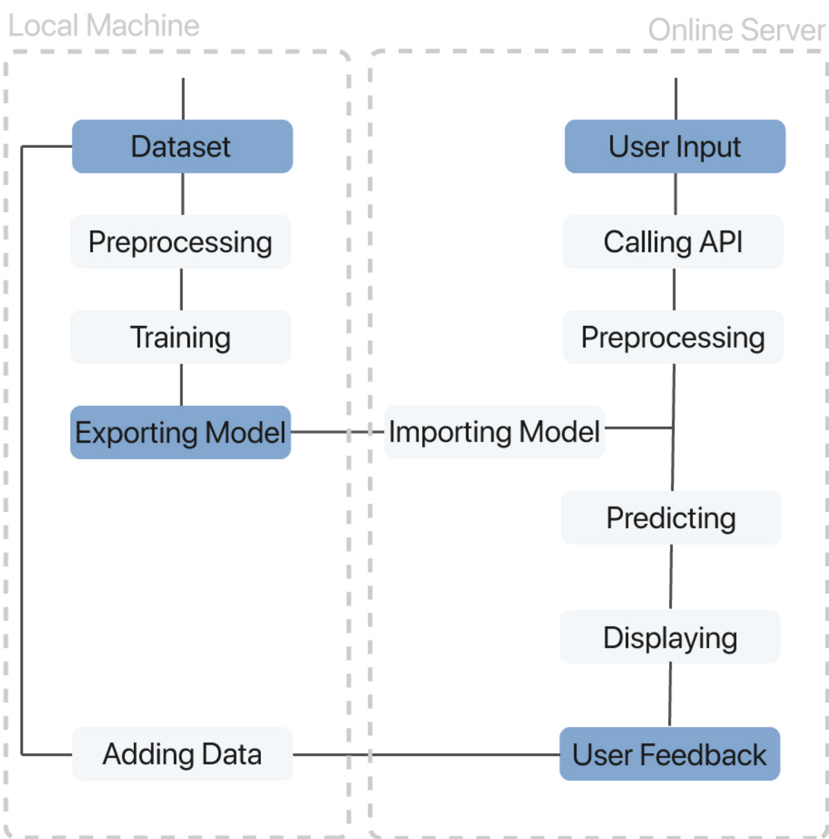

Figure 5.1: Pipeline of the project

5.2 Pre-processing

After collecting our data from the Twitter API, messages are pre-processed before getting passed to the neural network.

Truncating the tweets

Neural networks often require a fixed-sized input in terms of words. Here is the distribution of the number of words over the whole dataset :

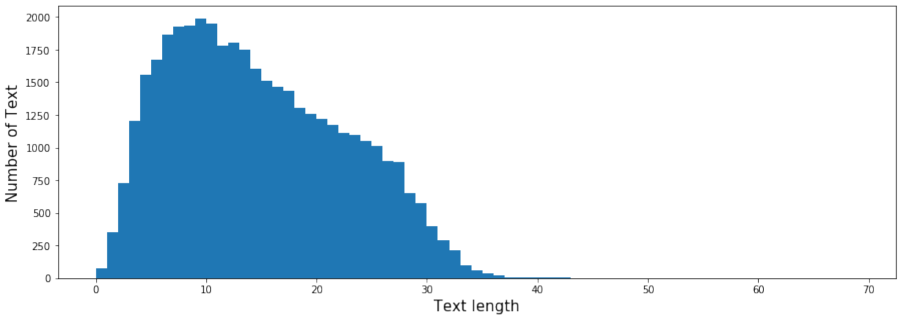

Figure 5.2: Distribution of the number of words

Given that result, we can state than limiting the size of the inputs to 55 words would not affect the performances. Moreover, a message longer than 55 words probably delivers more than only one sentiment, which would in any case make our system not work well.

Removing URLs and Usernames

Detecting the emotions doesn't require taking links into account, so they are removed. Leaving them would not help and, worse, could lead to noise. For instance, the ":/" in "http://" could be understood as an emoticon.
The usernames are not useful in terms of content, but it is useful to know their position. Indeed, "*I hate my life*" is a sad tweet while "*I hate @Bob*" is a hateful tweet. If "@Bob" was completely removed, the second tweet would become "*I hate*" and might be considered as simply sad. But knowing it is @Bob or @Luke isn't useful, so we replace every "*@User*" by the tag "*<USER>*".

Original	**Pre-processed**
Hey @Tom, check that out http://google.com/fun	Hey <USER>, check that out

Tokenizing the emoticons

The objective of this step is to gather similar emoticons under the same tag. For instance, **:)** and **:]** are both replaced by *<SMILE>*. The work of Romain Paulus on pre-processing tweets for the GloVe project introduces a dynamic method to identify the emoticons [6]. To deal with the different combinations of eyes, noses and mouths, a dynamic approach is used, by simply matching the regular expressions formed from that list of nose, eyes and mouths, where mouths often indicates the sentiment:

Eyes	8 : = ;
Nose	[’ ‘ \-] ?

and the corresponding combinations with the tags:

<table>
<tr><th>Expression</th><th>Example</th><th>Tag</th></tr>
<tr><td>"[(d]" + nose + eyes</td><td>(-:</td><td><SMILE></td></tr>
<tr><td>eyes + nose + "[/|l*]"</td><td>=|</td><td><NEUTRALFACE></td></tr>
</table>

Table 5.1: Emoticons smart detection

Many configurations still have to be taken in account, but this method helps to drastically reduce the lines of code and increase the robustness of the system.
When using the dataset based on the emoticons, it is important to suppress the emoticons during the training. If not, there will probably be an over fitting problem, where the weights of the neural networks will be disproportionately large for the emoticons' embeddings. The model would then classify the messages only by classifying the emoticons within the message, and would not take into account the entire content of the message.

Handling the hashtags

If the dataset has been entirely built on the hashtags labelling technique, then the hashtags have to be removed in order to avoid over fitting. The alternate solution we use is to consider hashtags as normal words. To do that, we simply remove the # sign.

Handling apostrophe

English is a language that uses many apostrophes. In order to minimize the vocabulary size, we need to extract the full form from the contracted form that contains the apostrophes. This is crucial in the case of the negation. "You aren't" is transformed into "you are not" and helps us to emphasize the negation. Here is a list of examples of this pre-processing :

Original	**Pre-proccessed**
I'm happy	I am happy
You aren't very clear on that	You are not very clear on that
I couldn't do it	I could not do it

Cleaning

This step aims to make the data uniform. Everything is set to lowercase and only ASCII characters are authorized. Punctuation can't really be ignored, since an exclamation mark gives information about the sentiment of a sentence, for example. To avoid having noise due to punctuation ("*Wow!*" would be considered as a unique word instead of "*wow*" and "!"), we force a space before and after every punctuation symbol. The following table shows a few examples of this pre-processing step:

Original	**Pre-proccessed**
Wow! That is crazy	wow ! that is crazy
I love having money in my Bank Account $$	i love having money in my bank account

Tokenizing and padding

This step transforms a message, presented as a string, into a list of words. This will allow us to pre-process words independently from each other. Each word will later be transformed into a list of coefficients from the embedding matrix. We still need to have the inputs of an equal size for the neural network. When the number of words in the tweets is less than the maximum words determined earlier, we add empty words to reach that number. Here are examples of the tokenizing and padding processes for a maximum number of words equal to 5.

Original	**Pre-proccessed**
You look really good today	["you", "look", "really", "good", "today"]
I love you	["","","i", "love", "you"]

Removing stop words

In English, stop words are the most common words that are not useful in understanding the meaning of a sentence, such as "the", "a" or "an". Hence, those words are not really useful and would increase the vocabulary size. They are simply removed.

Lemmatizing

Lemmatizing consists of removing the conjugated forms and the plural forms of the words to reduce it to the root of the word. This reduces drastically the size of the vocabulary. For instance, "I am eating tomatoes" is transformed into "i be eat tomato".

5.3 The Emotion Multi Classifier

5.3.1 Model

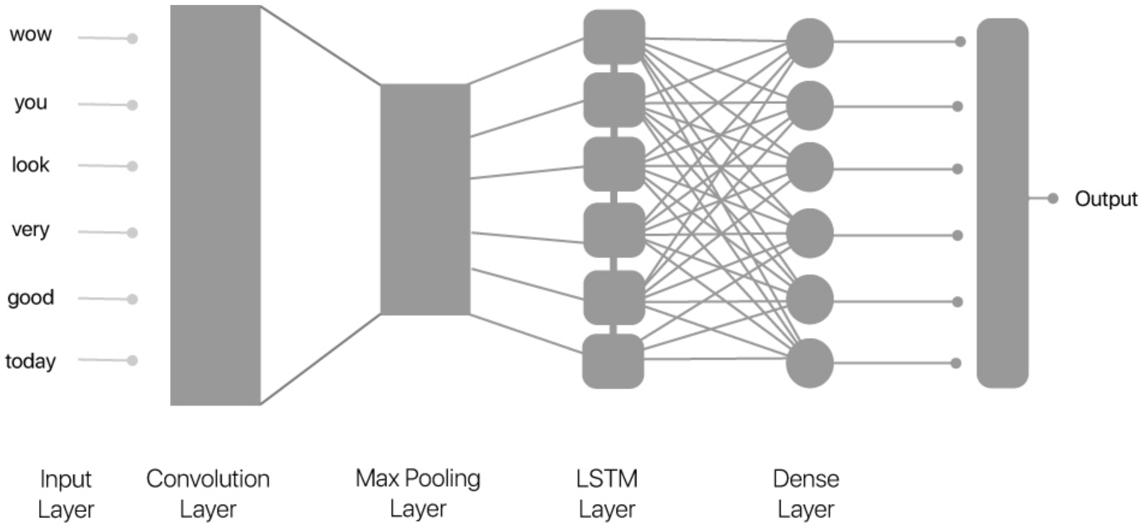

Figure 5.3: Emotion Multi Classifier

The model is largely inspired by the work of M. Sosa [9] but contains some differences. While their best model was composed of a LSTM layer followed by a CNN layer, the performances on our dataset seemed better with a CNN layer followed by a LSTM layer and a dense layer. Even if we increased the complexity of the model by adding a dense layer, it is still a simple model, since a dense layer isn't a very complex structure and can be perfectly used on a web server.

5.3.2 Parameters

The best parameters for our models were tuned through grid search and then manual testing.

Distinct classes	6
Dense Layer Size	6
LSTM Units	6
Embedding Dimension	100
Filters	64
Kernel Size	4
Pool Size	4
Dropout	0.5
Dropout LSTM	0.1
Epochs	30
Batch Size	32
Lr	0.00004

Table 5.2: Parameters selected for our prediction model

5.4 Results

Our best performing model, given the condition of relative simplicity (simple enough to be exported on the web server) is a CNN layer followed by a LSTM layer. The average accuracy may not seem very important, but we can notice valuable aspects of this system by looking more closely at the following matrix of confusion.

	funny	happy	love	neutral	worried	sad
funny	**0.82**	0.06	0.04	0.05	0.02	0.02
happy	0.17	**0.41**	0.14	0.14	0.09	0.05
love	0.10	0.17	**0.43**	0.13	0.10	0.08
neutral	0.12	0.12	0.07	**0.47**	0.14	0.09
worried	0.08	0.09	0.06	0.17	**0.40**	0.21
sad	0.06	0.07	0.05	0.16	0.23	**0.43**

Table 5.3: Confusion Matrix Prediction Emotion LSTM Dense

First of all, every class is represented in the prediction, which was not granted with such an unbalanced dataset. And the smallest class is the best predicted class.
Most of the wrong predictions are in fact predicting a similar emotion, and this is not necessarily bad. This could be explained by a bad generation of the dataset that mislabeled some messages. This also illustrates the difficulty to classify similar emotions. In our case, where we want to limit the complexity of the problem, those performances are acceptable.

Here is the evolution of the F1-score, a better metric, through the epochs.

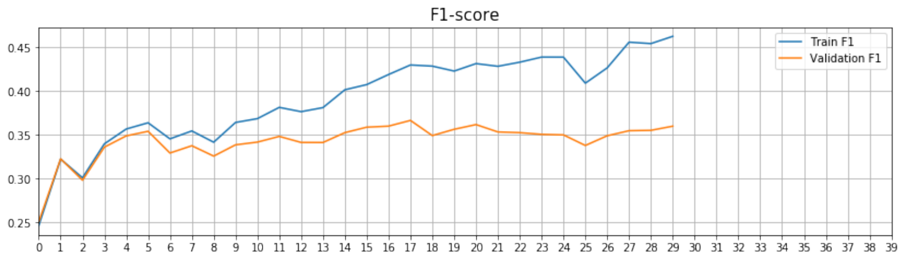

Figure 5.4: Evolution CNN LSTM

5.5 Feedback System

5.5.1 The Necessity of a Feedback System

When the model is pushed to production, we need a way to evaluate its performances when it is used by real users. Having such performance metrics is useful but it is not always possible. The metrics that we presented before depend on the model. But it doesn't give any idea about how good our dataset is. Indeed, the model is trained on the dataset. Hence, if the dataset is built with a lot of wrong labels, the model might perform very well on the dataset and might perform very poorly in reality.
Getting feedback from real users allows us to see if the dataset has been built well enough and if the theoretical performances of the model are not changing much in reality. The results of the feedback system are interesting mainly when they are put in perspective with our theoretical metric. Perfectly equal performances between the model predictions on the validation set and the feedback would imply that the dataset has been built perfectly.

5.5.2 The Feedback System

When it comes to emotion multi classification, a human can easily classify the messages. The idea behind the feedback system is to be helped by the users to validate or correct our prediction. Our feedback system is designed to be optional. It is possible that only a few of the users will take the time to correct the classification, which explains the necessity to extract as much valuable information as we can from this correction.

The users can modify the emotions they think are not accurate and then send the modifications. When the modifications are sent, both the messages that have been corrected and the one that have not are taken into account. When the users send their modification, the messages in which the emotion class is unchanged are implicitly correct.

5.5.3 Results of the Feedback

Here are the results for the feedback collected for more than 150 different users. It corresponds to 4500 emotions corrected by a user.

	funny	happy	love	neutral	worried	sad
funny	**0.71**	0.15	0.04	0.07	0.03	0.01
happy	0.21	**0.38**	0.15	0.13	0.08	0.05
love	0.11	0.18	**0.42**	0.13	0.8	0.09
neutral	0.12	0.10	0.09	**0.47**	0.12	0.11
worried	0.08	0.09	0.08	0.15	**0.35**	0.26
sad	0.07	0.07	0.04	0.18	0.20	**0.44**

Table 5.4: Confusion Matrix Prediction Emotion LSTM Dense

We can notice that the performances are bit lower compared to the performances on the validation set. This could partially be explained by the difficulty of analysing emotions even for users, who might make mistakes when correcting the classification.

After some hand testing, it also appears that the business and newspaper accounts have worse performances than personal accounts. This is not very surprising since the dataset has been built based on emoticons, which are mainly used in personal accounts.

Chapter 6

Emotion Prediction

6.1 General Idea

Predicting the emotion of the next post of a user is a step forward in the understanding of emotions. Instead of looking at the emotion of a user at a given instant, predicting emotions implies looking at the evolution of their emotions. The result of such a prediction represents valuable data about the user and can be used later to detect ups and downs before they actually occur. From a given number of recent messages, the system tries to predict what will be the emotion of the next message of a user.

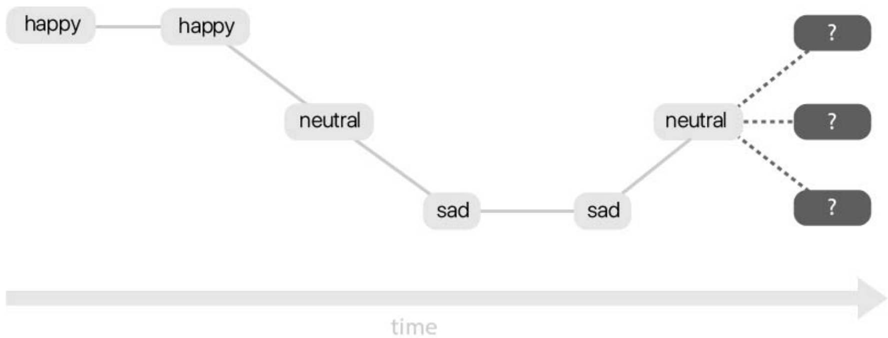

Figure 6.1: Concept of the emotion prediction

It is important to notice that we are predicting only the next emotion. The objective is to grasp a limited emotion trend in time. It is not supposed to understand the emotions of the users from a global point of view. For instance, it would be useful to understand how a user might feel about a thing considering the emotions they expressed about similar things. However, it cannot really induce anything about the mental state of the user. That would require a global understanding and analysis of their emotions.

6.1.1 Creating the Dataset

Pipeline for the dataset

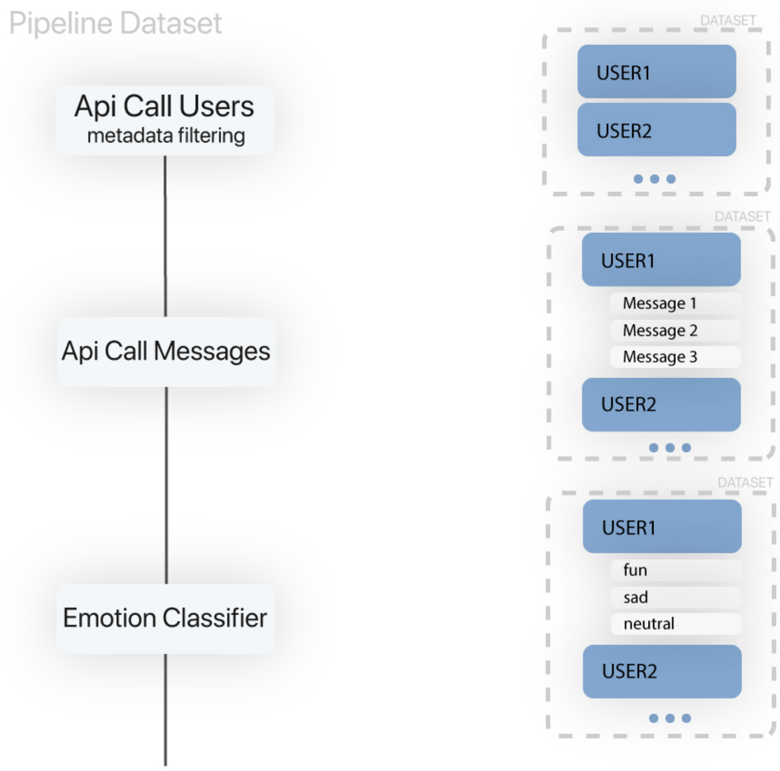

Figure 6.2: Pipeline for the Dataset Creation

API Call Users

The first part of this pipeline consists of calling the Twitter API in order to retrieve a list of users. The metadata filtering includes filters based on the number of recent tweets in order to select only active users, for whom the evolution of the emotion might be more meaningful. The API of Twitter only allows the retrieval of a certain amount of the most recent messages of a user. Hence, filtering on the absolute number of posts of a users is not efficient, since it could select inactive users that have not posted in a while and for whom the API would not retrieve any tweets. To avoid retrieving impersonal accounts, also called businesses, we filter this metadata. Finally, in order for the messages to be passed to the emotion classifier, we only select the English speakers. Here is the metadata parameters we used.

Parameter	**Value**
Number of Users	2,000
Language	English
Business	False
Number of Recent Messages	>31

Table 6.1: Parameters used to build the dataset for the Twitter API

API Call Messages

For each user, we call the API to retrieve their previous 31 messages. From those messages links and retweets are filtered out. They are then pre-processed in order to be fed to the classifier. Since the quantity of calls to the API is quite important, some pauses of several minutes are required in order to stay in the authorized quota. As a consequence, it makes the process long and it is the reason why we chose to only retrieve the messages for 2,000 users.

API Call Emotion Classifier

After retrieving the messages, we apply the emotion classifier to all of the messages of every user and store the results in one-hot vectors. From the 6 emotions given by the classifier, we only take 4 emotions for our prediction system in order to reduce the noise coming from the classifier and to balance the dataset. Here is the table showing the conversion of the emotions classified into the emotions to be predicted:

emotion	new label
sad	0
neutral	1
love	1
worry	2
happy	3
fun	3

From those 31 messages per user, 30 are used as an input and the most recent one is used as the validation output.

6.1.2 Data Augmentation with the Window Method

The dataset retrieved by the API was not large enough to train a performing model. Hence, we applied the window method. Instead of considering the last 30 previous messages to predict the most recent message's emotion, we can work with a lower number of past messages. By using a window of x messages ($x < 31$) means that for a sequel of 31 messages which initially constituted only one input, we can now extract $31 - x$ new features with their validation output. A trade off has to be found between the size of the dataset (increased by increasing x), and the precision of the model (increased by decreasing x).
Here is an example of the window method for a window size of 4, and 4 different classes, labelled from 0 to 3.

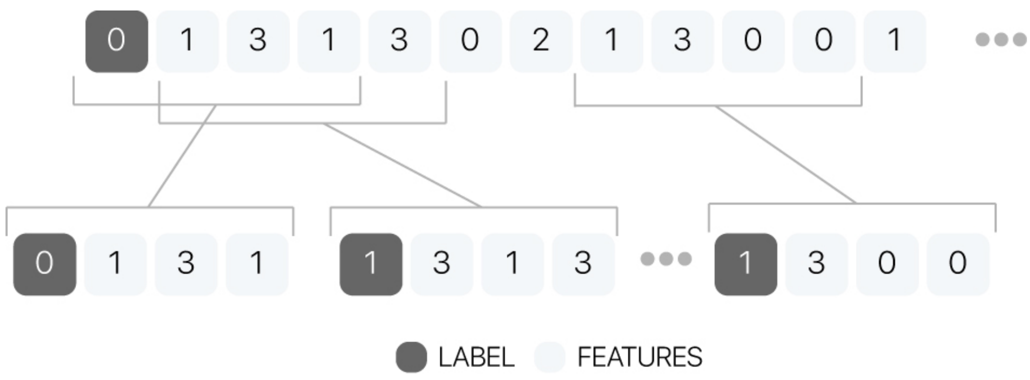

Figure 6.3: Window Method Illustration

We chose to work with 7 messages per data line. One message, the most recent, is used as the label, and the 6 other messages are the input. From a dataset of 2,000 entries (one for each user) of 30 messages each, we created a new dataset of 46,000 entries of 7 messages each.

6.2 Pipeline

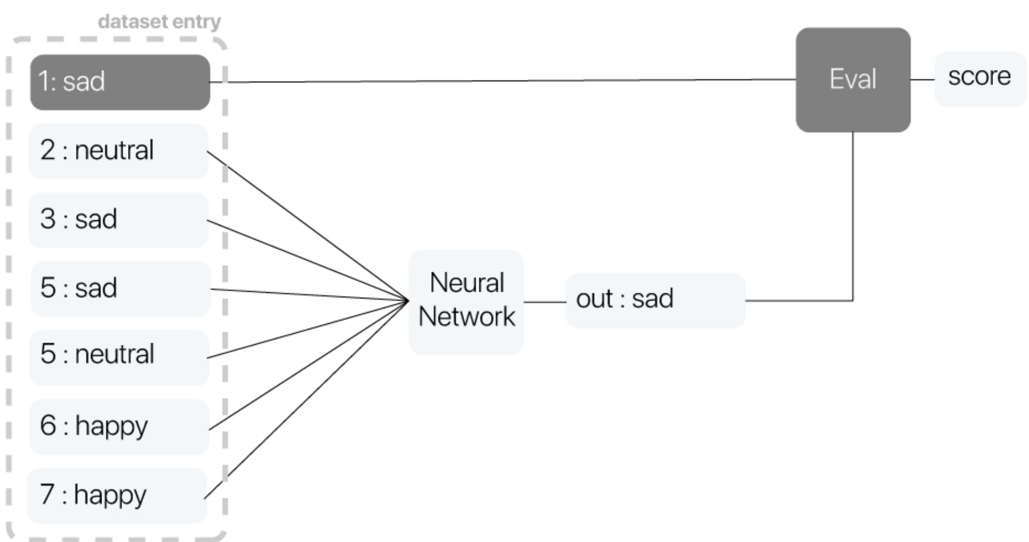

Figure 6.4: Prediction Pipeline

Once the dataset is built, we divide each entry into an input and a label. The most recent emotion will be considered as the expected output. The next emotions will be fed into the neural network. The comparison between the expected output and the output of the neural network will help us to train the model and evaluate the model afterwards.

6.3 Model for Emotion Prediction

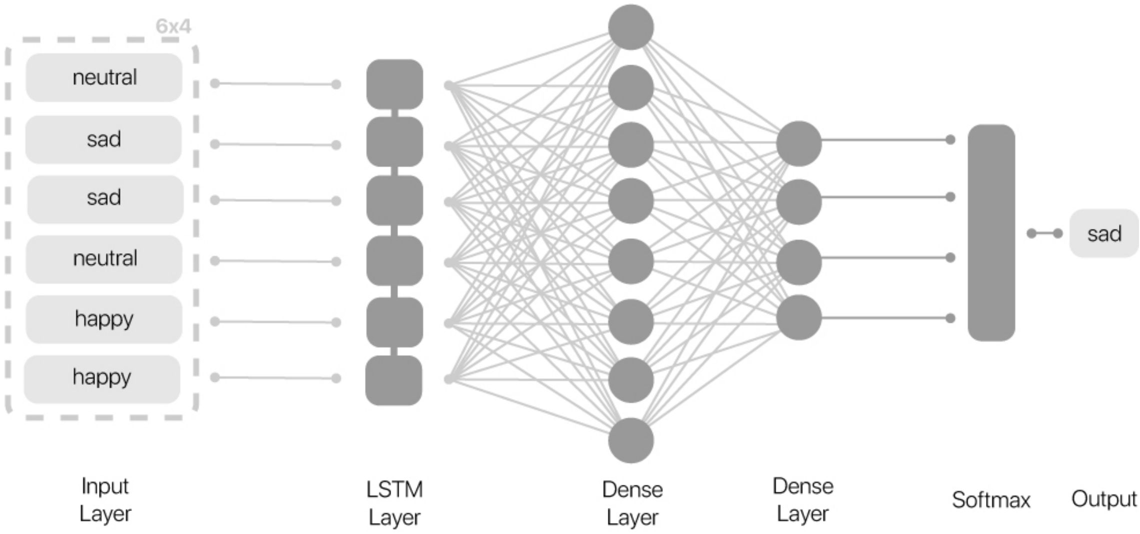

Figure 6.5: Neural Network for Emotion Prediction

6.3.1 Parameters

The best parameters for our models were tuned through grid search and then manual testing.

Distinct classes	4
Dense Layer Size	30
LSTM Units	6
Batch Size	64
Epoch	100
Optimizer	Adam
Lr	0.0004
Lr decay	0.0001

Table 6.2: Parameters selected for our prediction model

6.4 Results

6.4.1 Experiment Setup

The dataset we worked on, resulting of the window method, counts 46,000 entries of 7 past emotions each. This gives us inputs of a size of 6 past emotions, the most recent emotion being used as a label. Our training set represents 80 percent of this dataset while the testing set represented only 20 percent of it. The accuracy and other metrics presented below are measured when labelling the testing set.

6.4.2 Model Evaluation

Here is the confusion matrix when running the model only on the testing set.

	sad	neutral	worried	happy
sad	**0.40**	0.17	0.21	0.21
neutral	0.17	**0.38**	0.20	0.25
worried	0.22	0.23	**0.31**	0.23
happy	0.24	0.20	0.23	**0.33**

Table 6.3: Confusion Matrix Prediction Emotion LSTM Dense

Studying the confusion matrix for emotions is important since it can help to understand what the model is understanding or misunderstanding. The results of such a prediction are promising. Here, for each emotion, we are quite above the naive score of 0.25 (for a random classifier with 4 classes). Among other trained models, this is the one we picked to be sent on production. We can see that its best performances are when predicting sadness, which is the most important emotion to grasp in the scope of our project.

6.5 Feedback System for the Emotion Prediction

6.5.1 The Automatised Feedback System

On the website, the idea for the feedback system is to use the window method on the requested user. From the results, we can now measure the performances of the prediction the same way we did to evaluate the model. Hence, for one given user, with 30 retrieved messages, we can extract and get a feedback of our model on 23 sequences of emotion, by using a window size of 7.

6.5.2 Combined Feedback System

The downside of this feedback system presented above is that it is fully automatised and the users don't correct the prediction themselves. It means that such a feedback system is biased by our emotion multi classifier.
However, since the emotion multi classifier gets feedback directly from the user, we can then extract a user feedback for the prediction model from the emotion multi classifier feedback. Based only on the sequences where the user corrected the emotion, we can apply a similar feedback system with the window method as presented above. When evaluating the model on those corrected sequences of emotions, we can be certain that the performance evaluation only evaluates the model for the prediction and is not influenced by the emotion classifier.

6.5.3 Architecture of the two Feedback Systems

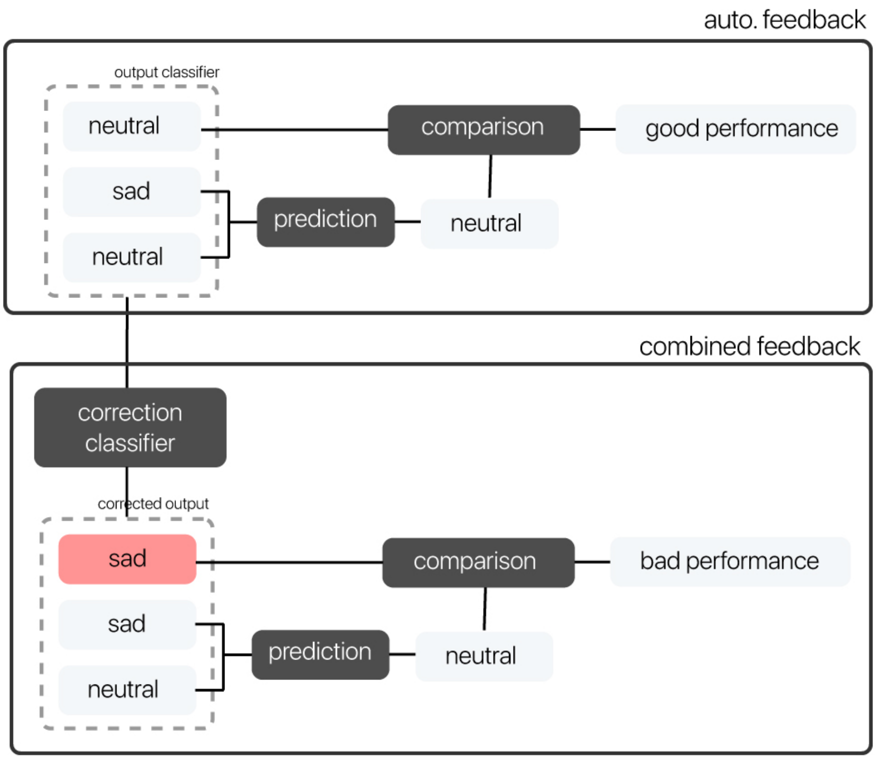

Figure 6.6: Feedback Systems for the Emotion Prediction

The upper part of the figure presents the automatised feedback that does not need any action from the user. It takes the input (here of size 3), takes the most recent message as the expected output, and predicts the emotion with the model with the input of size 2. Finally, it compares the expected emotion with the predicted one.

The combined feedback, on the lower part, requires the user to correct the emotion classification. When the user sends a correction, the modified emotions are sent to the prediction model feedback system and the result is compared to the most recent corrected emotion.

6.5.4 Results

The automatised feedback has been collected on more than 500 users, which means 11,500 entries to evaluate our model after the window method processing. And among those 500 queries, 150 were corrected by a user, which represents 3,500 emotion sequences on which we can evaluate the performance of the model. For those corrected inputs, the rate of corrected emotions is around 0.3. It means that roughly one third of the emotions are changed before feeding the combined feedback.

	auto. feedback	comb. feedback
accuracy	**0.35**	**0.33**

Table 6.4: Accuracy Score for the feedback of the emotion prediction

The auto feedback evaluates an accuracy that is almost equal to the accuracy we had during the model validation. The combined feedback detects a lower accuracy since this feedback takes in account the emotion classification correction. The difference is not significant and this can be explained by the model grasping patterns of emotions that do not depend on every single emotion in the input. The model seems quite resistant to significant noise - the noise being the mistakes of the emotion classifier - since it keeps around the same performance once the user correction is applied.

6.5.5 The Influence of the Dataset

This feedback system is influenced by how the dataset has been built and if the dataset covers a portion of the population representative enough of most of the users of our app. For instance, if during the creation of the dataset, we only select people between 16 and 18 years old, and the users of our website are more than 25 years old, the performances measured by this feedback system might be sensibly worse. A way to counter such an unrepresentative dataset for our user base would be to analyze their metadata.

6.6 Limits and Potential Improvement

6.6.1 The Reduced Quality of the Dataset

The window method is great when it comes to increasing the amount of available data. It allows us to improve the performances when the data originally collected is in limited quantity. But the window method has a price. It reduces the quality of the model on two aspects. First, it reduces the length of the input: from 30 messages to 7 messages in our case. Secondly, it may create some wrong patterns. Using the window method might duplicate the same sequence many times. Consider the following example with a window of 2: sad - happy - sad - happy - sad -> happy. The mood of the user is alternating, and in the initial dataset it would have been only one line that would not receive a lot of importance during training. The window method would create the following training data: sad->happy, happy->sad, sad->happy, happy->sad, sad->happy. After applying the window method to this changing mood, it creates two new patterns repeated several times that will be of bigger importance during training and would not be representative of a global behaviour.

6.6.2 The Double Error

The dataset used to train and test the model has been built by our emotion multi classifier described above. It means that the errors of the classifier will make the emotion prediction worse, since the error of the prediction will be composed of the error of the model for the prediction and the model for the classification. Increasing the performances for the model can be done by either improving the classification model or improving the prediction model. This can also be done by improving the quality of the dataset.

6.6.3 Multi Prediction Model

A great idea would be to implement a prediction that would predict the emotion for the next few messages instead of the emotion of the next one only. This would considerably increase the interpretability of the result. Indeed, with only one emotion predicted, it is hard to get any general understanding of the future emotion of the user. With a prediction of the emotions of the next few messages, we could extract some deeper patterns, and then have a global understanding of the user's future emotion. This would require having a larger window in order to have more data as the input. A larger window would require having a bigger dataset and avoiding the window method.

Chapter 7

Depression Classification

7.1 General Idea

Chapter 5 gave us a powerful tool to analyze emotions by multi classifying them. While Chapter 6 used the past few emotions to predict the next emotion, in this chapter we study the sequence of emotions of a given user from a more general point of view. The objective is to determine if a user is depressed or not with their history of messages.

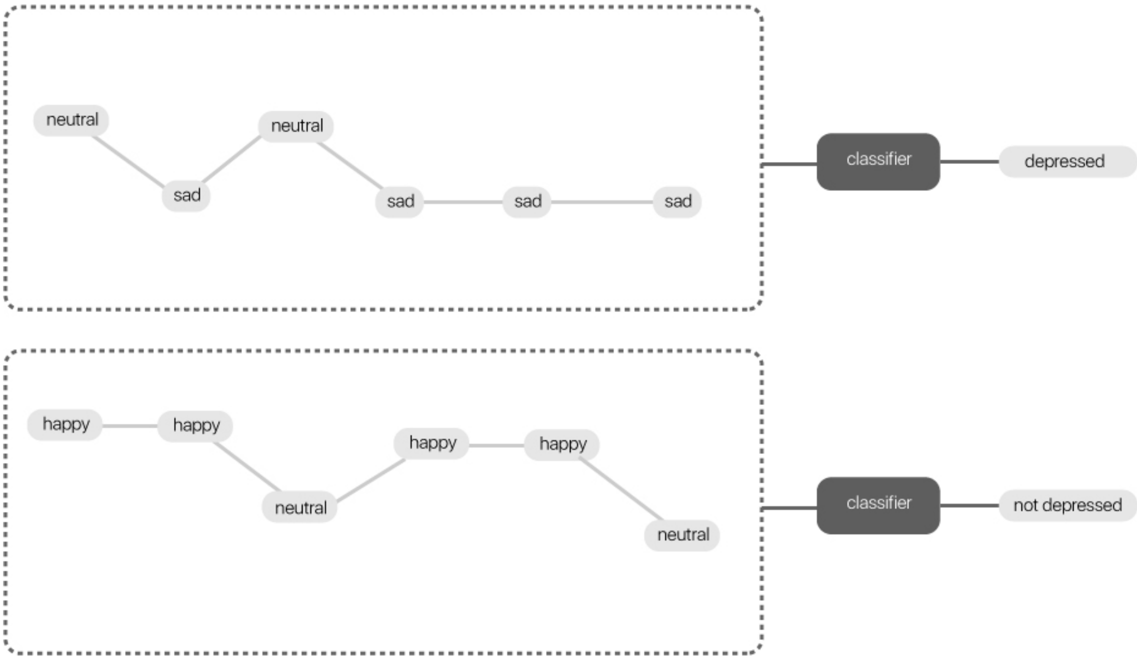

Figure 7.1: Concept of the depression classification

7.2 Creation of the Dataset

7.2.1 The Rule for Depression

Twitter is a powerful tool when it comes to extracting messages for a given emotion or feeling as long as a rule is properly defined to identify this particular emotion. A rule is well defined if a message verifying the rule will be considered by a human as containing the emotion we are trying to detect and a message not verifying the rule will not be considered as this same emotion.

Several mistakes can be done, by creating rules whether too restrictive or too general. For instance, many depressive messages will contain the word "sad", but considering this rule would lead to a lot of wrong classifications between sad messages and depressive messages. While a sad message can be written by almost anyone, a depressive message is supposed to be written by depressive users.
The rule of selecting messages that contains the hashtag *#depressed* is the best solution that we found. Most of the messages that will contain that hashtags will be from depressive users. Yet, it is not the perfect rule since those users, who are using the hashtag *#depressed*, are the depression aware users: people that are aware of their depression. And the point of detecting depression is actually to tell unaware people that they might be depressed. The depression of a user aware of their depression will be certainly more visible than on an account of someone who does not know about their depression. But a rule that wouldn't consider only the depressive aware users is not easy to create and it would certainly involve a metadata analysis. For instance by clustering by age, gender or nationality.
Looking for a simple model, our rule will simply be "A user is depressive if one of its last messages contains the hashtag *#depressed*" but it is obviously not a perfect solution. The label "not depressed" will be attributed to users that are not using the hashtag *#depressed* in their recent messages.

7.2.2 Constituting a Dataset of Sequences

For each user in our dataset, we retrieve their last 30 messages and the messages that are preprocessed are then sent to the emotion multi classifier. For each user, the last 30 emotions of their messages, called the emotion sequence, correspond to the input of our system. In order to have a simple model, we are not taking the date of the messages into account. In other words, only the position of appearance matters. This time, the window method is not used for two reasons. First, we only have two classes, depressed and not depressed, and the dataset is made from an equal number of both classes. Secondly, classifying depression is dealing with a serious issue, and it is very important to use every piece of information at our disposition to give the most accurate result as possible. Using for instance the last seven messages instead of the last 30 would severely decrease the depth and the accuracy of the prediction. We need to use as much information as we have to detect depression since the depression is detected from a general point of view, a trend in the emotions of a person. It is not a local emotion that only depends on the last few emotions.

7.3 Pipeline of the Depression Classification

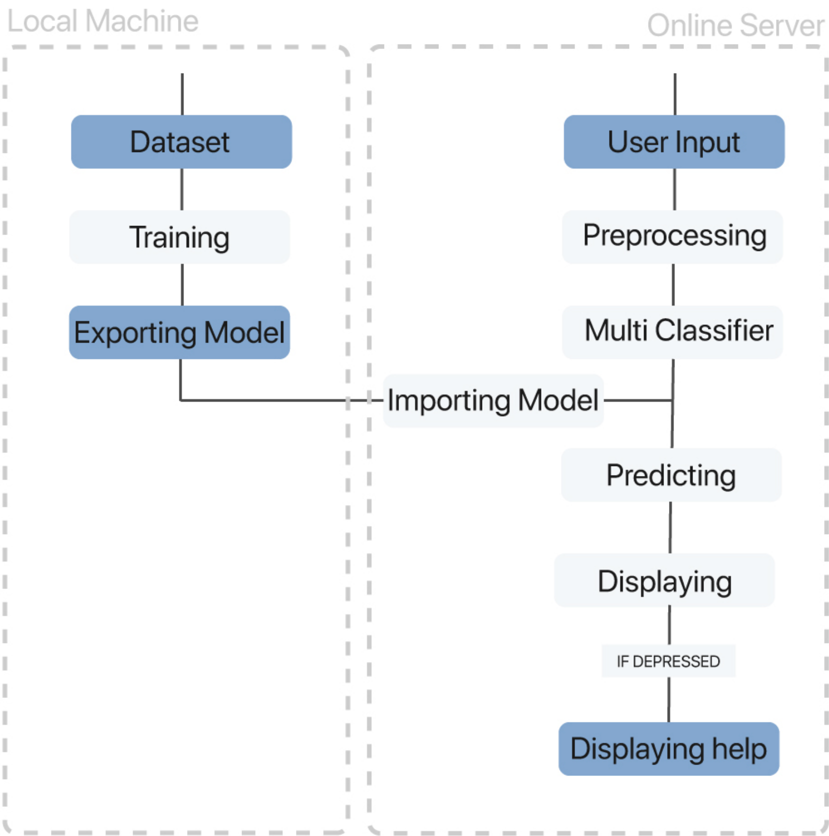

Figure 7.2: Pipeline of the depression classification

Interesting point: the display. Displaying the result of the depression is not easy. Displaying the result as a simple yes or no is too limiting. The interface has to be able to differentiate visually a user for whom the model is 90% sure of the depression and one for whom the model is really indecisive, for instance around 50% confident. Instead of displaying a binary interface, we display a progressive result for the depression classification, on the form of a score from 1 to 100. This number is simply the output of our model for the class not depressed, multiplied by a hundred. When the score is more than 50/100, the user is considered as not depressed, and nothing much happens, but when it is less than 50/100, a contextual window shares some links to counter depression.

7.4 Model for the Depression Classification

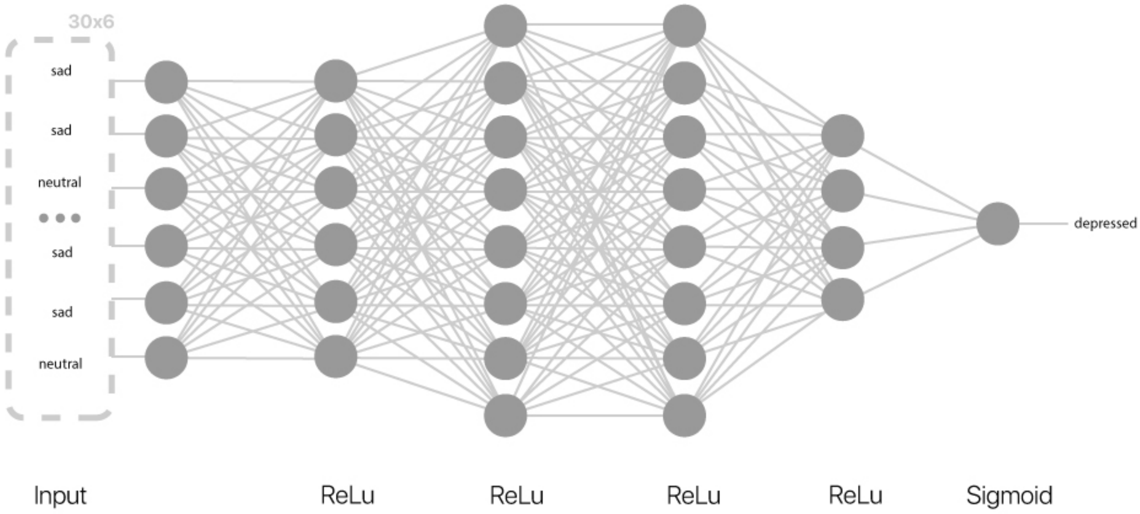

Figure 7.3: Neural Network for Depression Classification

It appeared that using LSTM didn't improve the performances so we instead use a simple model made of a succession of dense layers.

7.4.1 Parameters

The best parameters for our models were tuned through grid search and then manual testing.

Distinct classes	2
Dense Layer Size	30
Batch Size	64
Epoch	120
Optimizer	Adam
Lr	0.00004
Lr decay	0

Table 7.1: Parameters selected for our prediction model

7.5 Results

7.5.1 Model Evaluation

Here is the confusion matrix when running the model only for the testing set. Among many trained

	depressed	not depressed
depressed	**0.55**	0.45
not depressed	0.35	**0.65**

models, the model with these performances is the one used in production. Its average accuracy of 60% is not the best we obtained. But we decided to prioritize the true negative rate of the class depressed. It means that we want to limit the error rate when the prediction is "not depressed". Saying "not depressed" to people suffering from depression is quite dangerous since they may not take their mental health seriously enough.

7.6 Limits

7.6.1 A Blurry Definition

The main difficulty is about the construction of the dataset. The rules defined to detect depression are naive and are probably selecting only a few cases of depression, mainly the cases where the depressed people express their feelings on Twitter. But many depressed people isolate themselves from social networks, and thus, they would not be represented in the dataset.

7.6.2 Multiple errors

The dataset used to train and test the depression classifier has been built by our emotion multi classifier described above. And the rule used to label the dataset is not perfect either. This means that the errors of the classifier and the errors of the labelling will make the depression classification worse, since the error of the depression classification will be composed of the error of the model for the depression classification, the error of the dataset labelling and the error of the model for the emotion multi classification. Increasing the performances for the model can be done by improving the emotion classification model, the depression classification model or the dataset labelling.

7.6.3 No Feedback System

While for the emotion classification, it was pretty easy to implement a user feedback, in the case of the depression classification it is much harder. Only humans can provide an accurate feedback system when it comes to the emotions. However, it is hard and requires some expertise to be able to say if someone is depressed or not. And it is for sure impossible to do so by considering only the last few messages on social networks. Even asking potential depressed users if they are feeling depressed would not be very useful since people cannot psychoanalyze themselves. An accurate feedback system for this problem would require the help of specialists that could give their opinion about the users' mental state.

Chapter 8

Web Platform

This part presents the main points of the web platform that has been developed for this project. Since it is not really in the scope of the subject, it will not be explained in much depth. Screenshots are available in the appendix.

8.1 General Architecture

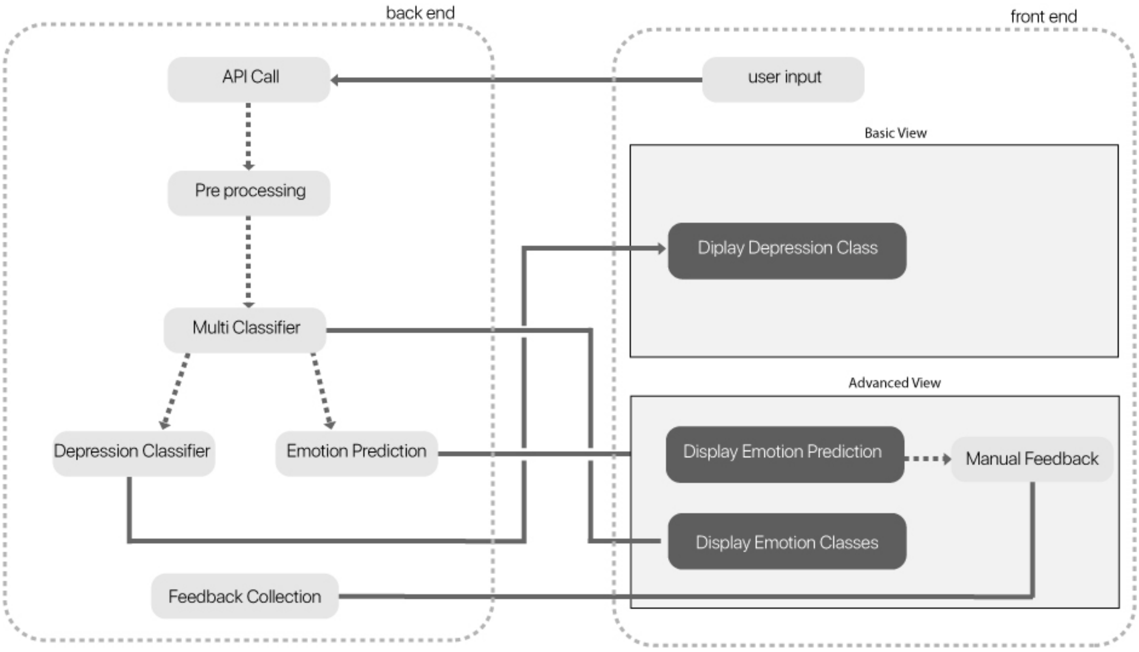

Figure 8.1: Interactions between the back-end and the front-end of the web app

8.2 Back-end Functionalities

For the NodeJs back-end server, the different functionalities are mainly used to recreate the environment in which the models have been trained.

API Call Calls the Twitter API for a given username, given as an input, and retrieves the last few tweets of the user.

Pre-processing Implements the pre-processing pipeline defined earlier.

Predict Loads the different trained models and predicts outputs. Sends the output to the front-end interface.

Feedback Collection Retrieves the different correction for the emotion multi classifier and computes the error rate for both the emotion prediction and the emotion multi classifier.

8.3 Front-end Functionalities

Input username Displays a field, retrieves the input of the user for the username to analyze and sends it to the back-end.

Loading page Displays loading information to the user during the back-end computations.

Conditional Depression Class Display Displays the results of the depression classification and displays a helping modal window in the case where the user is depressed. This window gives the user some useful links to get information about depression.

Advanced View for other results Displays an advanced view on which are displayed the details of the multi-classification for the emotions. The results of the prediction are also displayed on that view.

Manual Feedback Allows the user to review and modify the results of the emotion multi classification in the advanced view and sends the results of the feedback to the back-end.

8.4 Bridge for Models

The whole architecture relies mainly on the modules Tensorflow and Tensorflow-js. The first one, a Python module, allows us to train and select easily our models on Python. Then we chose a back-end in NodeJs to implement the library Tensorflow-js, which is a library that implements in Javascript most of the common neural network cells and can import pre-trained models. One of the aspects of the project is to use simple models so that the import through such a bridge like Tensorflow-js could be done easily.

Chapter 9

Discussion

9.1 Detecting Depression

We presented a way to detect depression when it is occurring. Even if this has not been tested and we did not evaluate any performances for that task, we believe that our model could totally predict depression before it occurs. This would require access to many more messages for a given user both to construct a more complete dataset and have a sample of the user mood before the depression, and make the system more accurate by increasing the dimension of the inputs.

9.2 Reducing Exposure

Not only can our work be used to detect depression, but it can additionally be applied with ease to actively mitigate the spread of depression.
A common pattern for depressed users is to follow and interact with other depressed users. They would also be more affected by depressing content like bad news or sad articles.
Hence, a first active step toward helping depressed people would be to reduce their interactions with other depressed people. This would imply classifying every user of the social network into depressed users or not and then make the depressing posts less visible to depressed users and more visible to not depressed users.
A second step would be to classify news medias on those platforms by classifying their articles/news (which would require an emotion classifier for articles), and then prioritize the ones with positive content to be displayed on the page of the depressed users.

Chapter 10

Conclusion

Looking for application of deep learning on emotion analysis, we introduced three different deep learning models. The first one is the multi classification of short messages, into six different classes of emotions. The second one is the prediction of the emotion of the next publication of a user considering their last few messages. The last one aims to detect whether a user is depressed or not by looking at their last messages. By using mainly Long Short Term Memory cells and Convolutional Neural Networks, we managed to create models performing well but simple enough to be exported on a web server.

We presented several methods to generate a dataset from the Twitter API. The first method uses the hashtags to label the messages and the second uses emoticons. We also showed how a dataset can be built with the emotion classifier and how this can be used to generate datasets of sequences of emotions.

The emotion multi classifier is at the center of our work since the other models take classes of emotions in input. The two other analyses are studying emotions from different points of view. The emotion prediction is a local analysis over only the last few messages that tries to find a pattern in those last few emotions for a given user. The depression classification takes a more global point of view by analyzing the whole available history of the emotions of a user and tries to grasp a trend in the mood of the user.

We produced a web platform, to make sure of the simplicity of the use of our models and to give an idea about how the results can be used on a web platform. This website was used to collect the feedback for the different models and was also used to sensitize users to mental health.

We showed how simple it can be to analyze emotions and detect depression. The depression detection is a step towards mitigating the spread of depression over social networks and can be used to fight against isolation of depressed individuals. Such work has to be acknowledged and can be easily reproduced and improved in order to provide support and help to the users in need.

Appendix A

Web Platform Screenshots

A.0.1 The Homepage

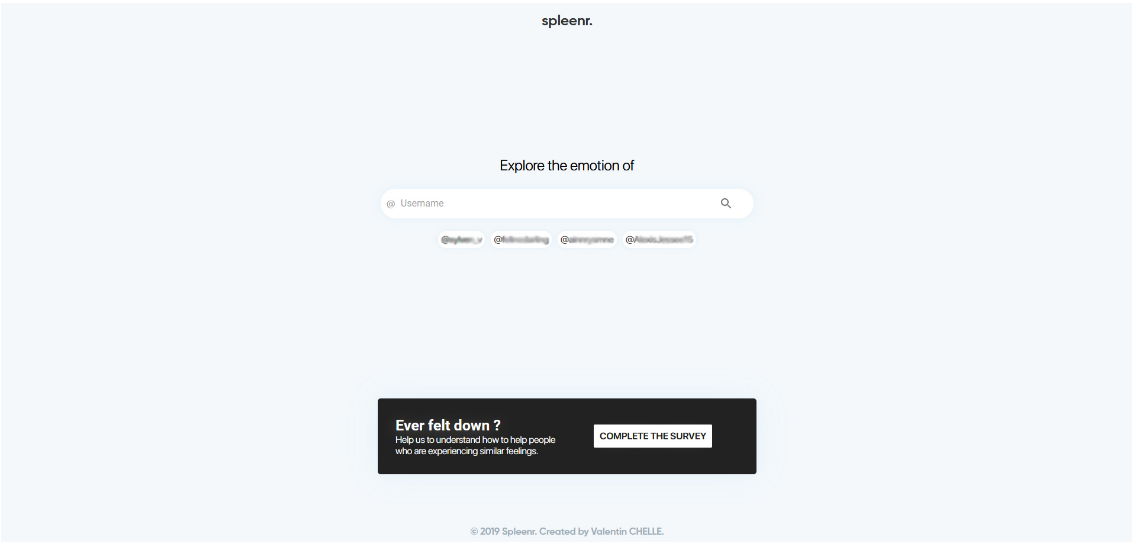

Figure A.1: Screenshot of the homepage with the username input

The main component of the homepage is the field where the user has to fill the username of the account they want to analyze. Some recommended accounts are given under the search bar. Finally, a link to a survey about mental health is provided on the bottom of the page.

A.0.2 Conditional View for the Depression Class Display

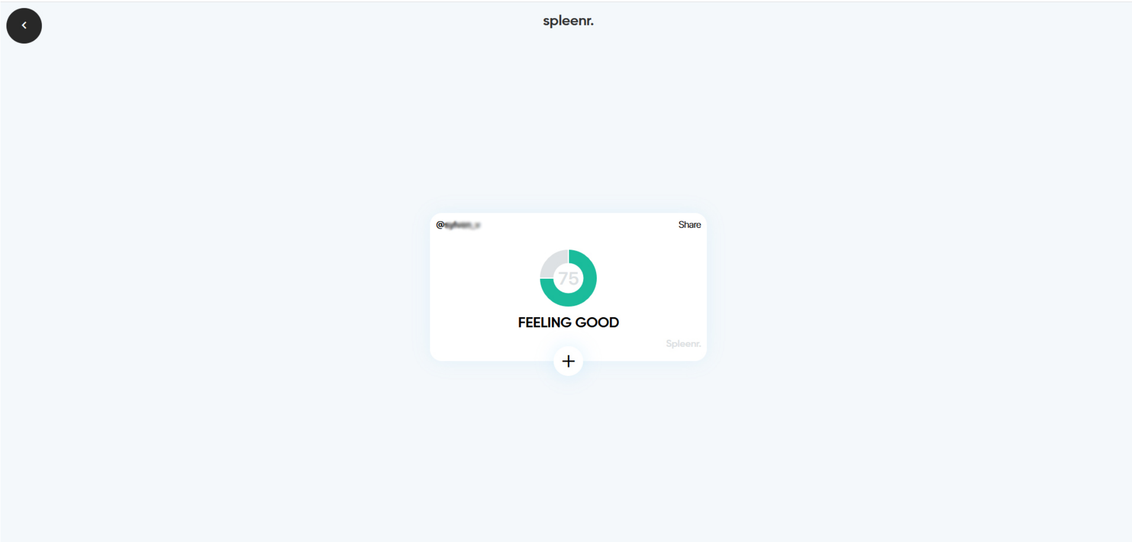

Figure A.2: Screenshot of the depression class display for a not depressed user

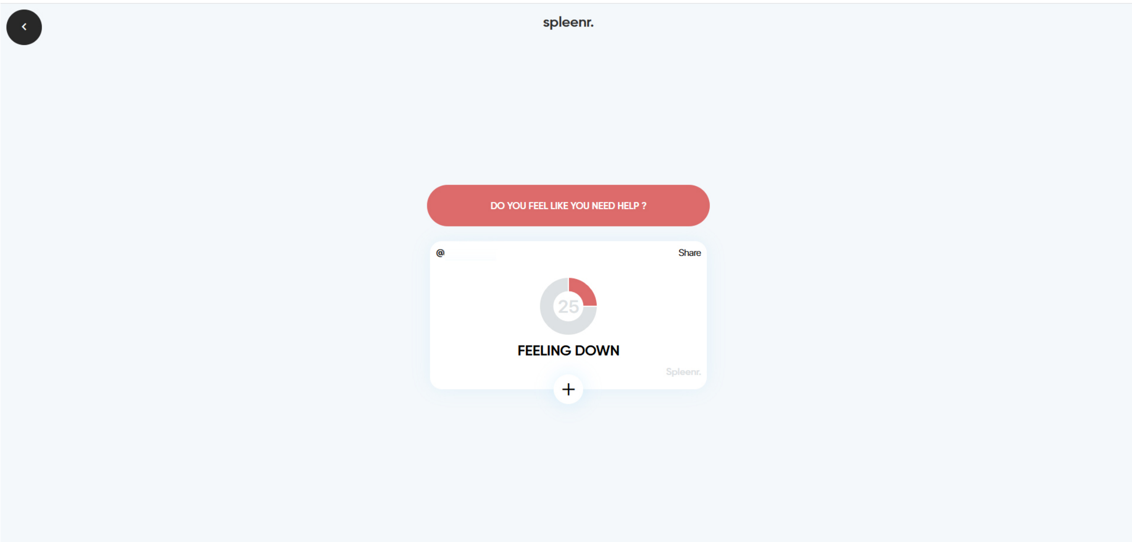

Figure A.3: Screenshot of the depression class display for a depressed user

A link to the mental health section of the NHS website is provided in the case of a user detected as depressed. A button, in the shape of a plus, invites the user to be redirected to the advanced view.

A.0.3 The Advanced View

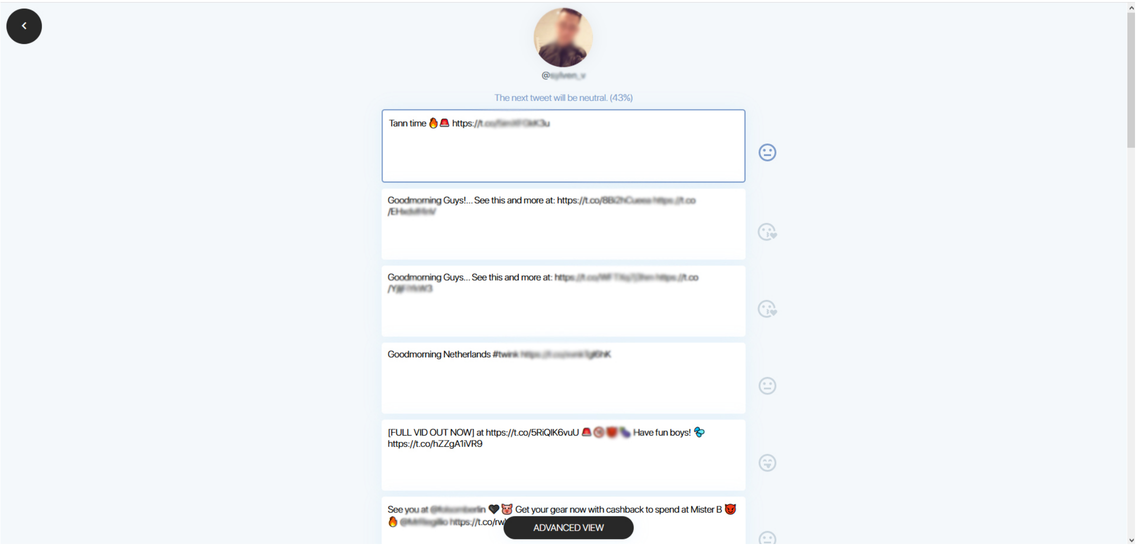

Figure A.4: Screenshot of the advanced view

The advanced view displays the messages one by one and on the left an emoticon symbolizes the prediction of the emotion for this tweet.

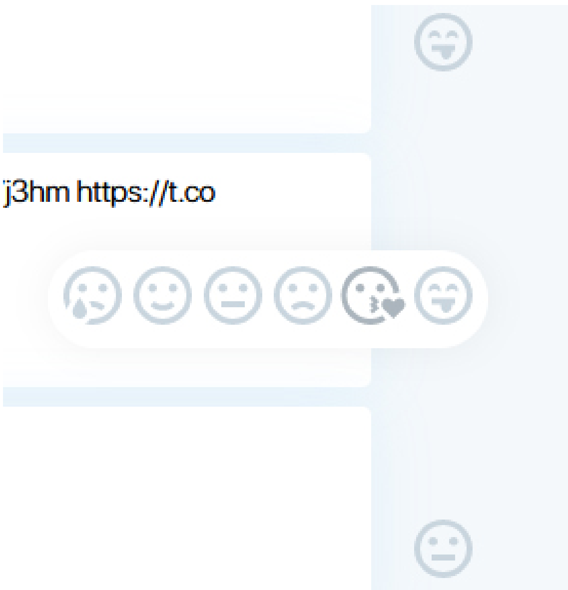

Figure A.5: Screenshot of the feedback button

When the user clicks on the emotion next to the message's emotion, they can modify the emotion, and the feedback is sent back to the server.

A.0.4 The detailed Advanced View

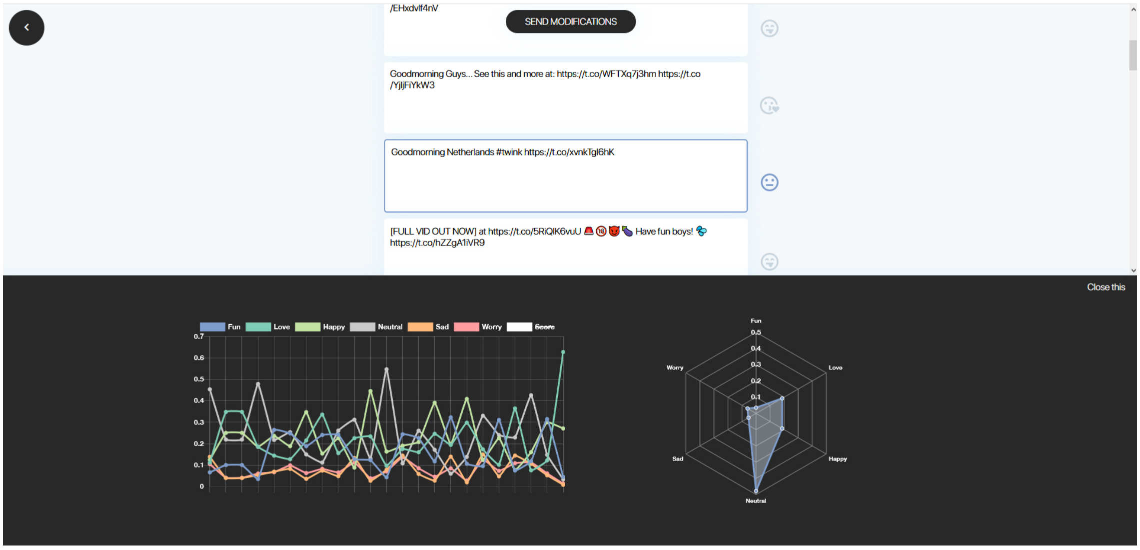

Figure A.6: Screenshot of the detailed advanced view

The left part of the detailed view is a graph representing the evolution of every emotion for every tweet retrieved through time.

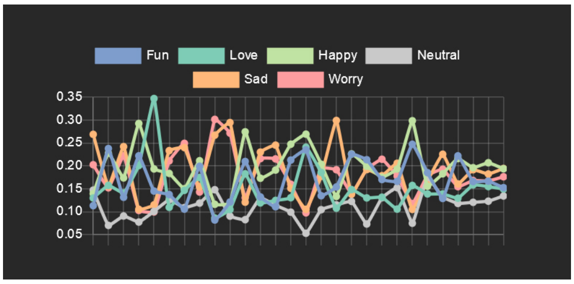

Figure A.7: Screenshot of the detailed Emotion Analysis of a unique tweet

The bottom right of the detailed view corresponds to the analysis of a unique message and provides details of the prediction by the neural network for every class of emotion.

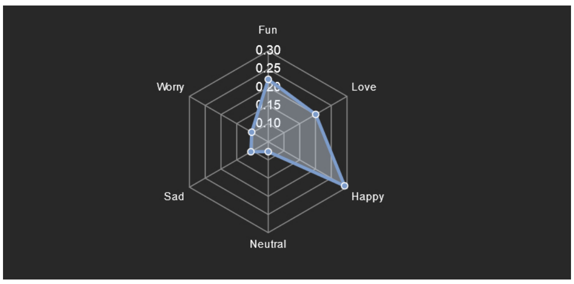

Figure A.8: Screenshot of the detailed Emotion Analysis of a unique tweet

Appendix B

Legal and Ethical Considerations

The work presented in this project is of theoretical nature. The project has not involved human embryos, humans, human cells or tissues, animals or developing countries. The author is not aware of any possible environmental impact by the project. The personal data used, i.e. the messages of the user, is read through the Twitter API but is never collected and never saved on the server. The datasets, built from personal data, do not contain any personal information. No copyrighted code or data has been used or produced. Great care has been made to credit original authors, when their work is referenced or built upon. The author is not aware of any direct possible misuse of this work. For more details, see the Ethics Checklist next page.

	Yes	No
Section 1: HUMAN EMBRYOS/FOETUSES		X
Does your project involve Human Embryonic Stem Cells?		X
Does your project involve the use of human embryos?		X
Does your project involve the use of human foetal tissues / cells?		X
Section 2: HUMANS		
Does your project involve human participants?		X
Section 3: HUMAN CELLS / TISSUES		
Does your project involve human cells or tissues? (Other than from "Human Embryos/Foetuses" i.e. Section 1)?		X
Section 4: PROTECTION OF PERSONAL DATA		
Does your project involve personal data collection and/or processing?	X	
Does it involve the collection and/or processing of sensitive personal data (e.g. health, sexual lifestyle, ethnicity, political opinion, religious or philosophical conviction)?		X
Does it involve processing of genetic information?		X
Does it involve tracking or observation of participants? It should be noted that this issue is not limited to surveillance or localization data. It also applies to Wan data such as IP address, MACs, cookies etc.		X
Does your project involve further processing of previously collected personal data (secondary use)? For example Does your project involve merging existing data sets?		X
Section 5: ANIMALS		
Does your project involve animals?		X
Section 6: DEVELOPING COUNTRIES		
Does your project involve developing countries?		X
If your project involves low and/or lower-middle income countries, are any benefit-sharing actions planned?		X
Could the situation in the country put the individuals taking part in the project at risk?		X
Section 7: ENVIRONMENTAL PROTECTION AND SAFETY		
Does your project involve the use of elements that may cause harm to the environment, animals or plants?		X
Does your project deal with endangered fauna and/or flora /protected areas?		X
Does your project involve the use of elements that may cause harm to humans, including project staff?		X
Does your project involve other harmful materials or equipment, e.g. high-powered laser systems?		X
Section 8: DUAL USE		
Does your project have the potential for military applications?		X
Does your project have an exclusive civilian application focus?		X
Will your project use or produce goods or information that will require export licenses in accordance with legislation on dual use items?		X
Does your project affect current standards in military ethics – e.g., global ban on weapons of mass destruction, issues of proportionality, discrimination of combatants and accountability in drone and autonomous robotics developments, incendiary or laser weapons?		X
Section 9: MISUSE		
Does your project have the potential for malevolent/criminal/terrorist abuse?		X
Does your project involve information on/or the use of biological-, chemical-, nuclear/radiological-security sensitive materials and explosives, and means of their delivery?		X
Does your project involve the development of technologies or the creation of information that could have severe negative impacts on human rights standards (e.g. privacy, stigmatization, discrimination), if misapplied?		X
Does your project have the potential for terrorist or criminal abuse e.g. infrastructural vulnerability studies, cybersecurity related project?		X
SECTION 10: LEGAL ISSUES		
Will your project use or produce software for which there are copyright licensing implications?		X
Will your project use or produce goods or information for which there are data protection, or other legal implications?		X
SECTION 11: OTHER ETHICS ISSUES		
Are there any other ethics issues that should be taken into consideration?		X

Bibliography

[1] A. Albelwi, S.; Mahmood. A framework for designing the architectures of deep convolutional neural networks. 2017.

[2] Maryam Hasan, Elke A. Rundensteiner, and Emmanuel Agu. Emotex: Detecting emotions in twitter messages. 2014.

[3] Richard Socher Jeffrey Pennington and Christopher D. Manning. Glove: Global vectors for word representation. 2014. URL: https://nlp.stanford.edu/pubs/glove.pdf.

[4] Timothy Liu. Multi-class emotion classification for short texts. URL: https://github.com/tlkh/text-emotion-classification/.

[5] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Sentiment classification using machine learning techniques. pages 79–86, 2002. URL: https://doi.org/10.3115/1118693.1118704, doi:10.3115/1118693.1118704.

[6] Romain Paulus. Preprocessing tweets. URL: https://nlp.stanford.edu/projects/glove/preprocess-twitter.rb.

[7] Matthew Purver and Stuart Battersby. Experimenting with distant supervision for emotion classification.

[8] researchgate. URL: https://www.researchgate.net/figure/A-simple-LSTM-gate-with-only-input-output-and-forget-gates_fig7_304066008.

[9] Pedro M. Sosa. Twitter sentiment analysis using combined lstm-cnn models.

[10] Statista. URL: https://www.statista.com/statistics/433871/daily-social-media-usage-worldwide/.

[11] Statista. Number of twitter users worldwide. URL: https://www.statista.com/statistics/303681/twitter-users-worldwide.

[12] Pete Trimmer, Elizabeth S Paul, Mike T Mendl, John Mcnamara, and Alasdair I Houston. On the evolution and optimality of mood states. *Behavioral sciences (Basel, Switzerland)*, 3:501–21, 09 2013. doi:10.3390/bs3030501.

[13] Twitter. Rules and filtering in twitter search. URL: https://developer.twitter.com/en/docs/tweets/rules-and-filtering/overview/standard-operators.html.