ML project II - Text Classifier

Lucas Massemin
Alexandre Dumur
Raphaël Dunant
Faculty of Information & Communication, EPFL Lausanne, Switzerland

***Abstract*—[to do last] The rise of social media has, as a consequence, massive creation of social data. This social data, which, by definition, is correlated to people in some way can be used to learn more about people actions, opinions, and feelings. Such a gain in knowledge could allow a State to analyse public opinion, a merchant to analyse people's reactions to the items it is selling, and many more applications. Our work focuses on tweet processing in order to predict feeling of the tweets, we achieve a new approach by combining neural network and n-grams, as well as the definition of a new metric. We chow that this approach improves the accuracy of the classifier:**

A critical part of scientific discovery is the communication of research findings to peers or the general public. Mastery of the process of scientific communication improves the visibility and impact of research. While this guide is a necessary tool for learning how to write in a manner suitable for publication at a scientific venue, it is by no means sufficient, on its own, to make its reader an accomplished writer. This guide should be a starting point for further development of writing skills.

I. Introduction

Text sentiment analysis is the field consisting of labelling text with a sentiment. Both binary [I] and more complex [II] classifications are implemented in the state of the art. This growing interest is motivated by the rise of social media which make it even more useful to be able to label text. For instance, one could predict public opinion on a topic by classifying a set of tweets correlated to the topic, more examples are given in [III]. This paper presents a tweet binary classification approach (positive or negative), associated to a convolutional neural network. The classifier has been created and trained especially for tweets. The whole classifying process consists of three main phases, $cleaning$ presented in section [X], $featuring$ explained in section [Y], and then $labelling$ tackled in section [Z],
We claim that the use of n-grams, combined to a broad stemming process, enhances the accuracy of the classifier. Moreover, we introduce a new metric, the $word\, pertinence$, intended to express the relative impact that the presence of the word in the text will have on classification outcome.

II. Models and methods

The classifier goes through three main phases in order to achieve final classification. Those phases are $cleaning$, $featuring$ and $labelling$. We will go through each phase, presenting the model associated and the methods that were used.

A. Cleaning

The cleaning phase is dedicated to making the text data more uniform.

1) Cleaning Model: We assume that the emotion is contained in the words' semantic, and not in the words' syntax. This seems to be a reasonably weak assumption, especially in a typo-prone environment like twitter. Moreover, we consider that words sharing a common root are likely to have the same semantic as their root.This applyes to plurals and conjugated words For instance, the words $"constructed"$ and $"construction"$ or $"consrtruuuction"$ can be map on $"construct"$ without a major loss of semantic information regarding to sentiments.

2) Cleaning Methods: The cleaning process is done in two steps.
First, the Standardization. This phase is responsible of getting rid of the duplicates, the useless syntaxes. The word that contains repetitions are matched with the one without repetitions. The second phase is about the Stemming and the vocab extraction. We construct here, a map between each word w of length l (the root) to a list of words that begin by w.In every tweets, we substitute each words in that list by the root.
We also expect some words to be written with various syntaxes ("haha", "Haaha", "hahaha", etc.), such words are stemmed by considering their first letters. Finally we use the natural language processing toolkit library $nltk$ in order to stem the words. Once we have processed all the words that contains the tweets, and have nice cleaned tweets, we can generate the vocabulary which is simply words mapped on their occurrences. The vocabulary also include the bi-gram and tri-gram representation of the words. It's important to us since it makes sense that considering two or three words together instead of one gives us more flavor of the sentiment of a tweet. We will talk more about the n-gram in the featuring part. The vocabulary will help us in the next process: the featuring. For more understanding about the cleaning process, please refer to Fig. 1

B. Featuring

The aim of the featuring part, is to create a feature representation of a Tweet. So that a tweet can be handle with this representation by our Classifier.

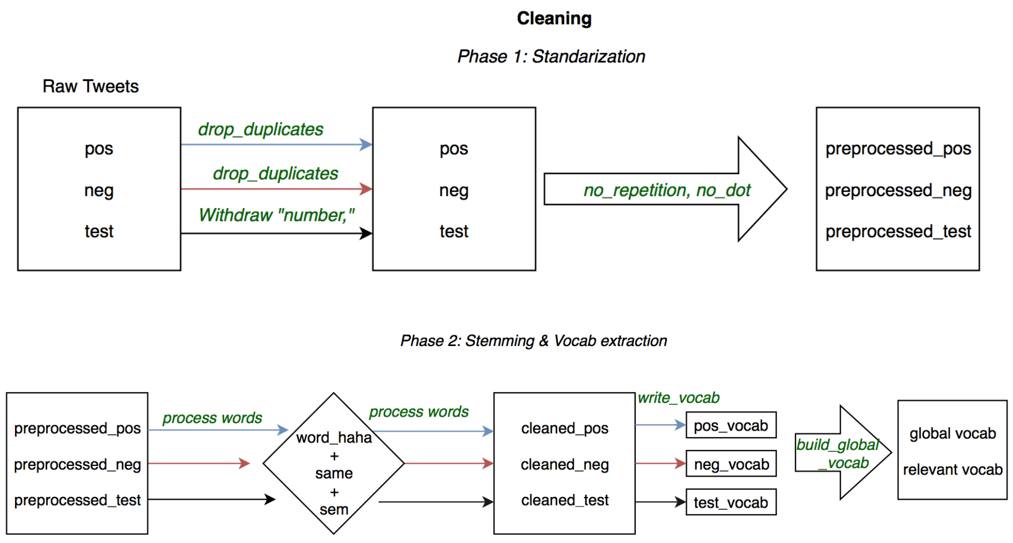

Figure 1. Explicit cleaning Pipeline. The standardization phase and the stemming and Vocab generation part. In green are the function that are used to transit from one entity to another.

1) Featuring Model: A tweet is said to be composed of several negative and positive aspects, and that its sentiment is more impacted by the aspects with higher magnitudes. Also, we consider that those aspects are expressed by the words that the tweet consists in. It thus makes sense to construct features for words, and to express the features of a tweet in function of the features of its words. The features of a word will be computed by using the GloVe model which involves a coocurence matrix and produce a vector representation for each word. [IV]. Furthermore, we forecast that some words need a context to fully express their semantic (like the negation, for instance). Finally, the model takes into account that some words are more expressive than others when it comes to sentiment.

2) Featuring Methods: To measure how expressive a word is in term of sentiment, we introduce a new metric, the $pertinence$. Given word w occurs $pos(w)$ times in positive tweets, and $neg(w)$ times in negative tweets, we define the pertinence as follow :

$$pertinence(w) = 2 * |0.5 - \frac{pos(w)}{pos(w) + neg(w)}|$$

This implies a vocabulary which is refer to the relevant vocabulary already depicted in Fig 1. Note that the pertinence is in $[0, 1]$ and that The more neutral a word is, the less pertinent it will be to determine the sentiment of our text, given we only label positive or negative tweets. Also note that this definitions is an estimate, accurate if all words appear sufficiently enough. Hence, in order to avoid major bias and statistic inaccuracy on words' pertinence, we consider only words that appear more than cut times in all tweets :

$$C = \{\, w \mid w \in W \ \text{ and } \ occ(w) > cut \,\}$$

Where W is the set of all words present in the tweets and

$$occ(w) = pos(w) + neg(w)$$

Now that our pertinence measure is meant to be accurate, we discard neutral words because they are not useful in determining if a tweet sentiment is positive or negative. We therefor define the set R as the set of $relevant\ words$, such that $R \subset C$:

$$R = \{\, w \mid w \in C \ \text{ and } \ pertinence(w) > P_{thres} \,\}$$

Finally, we claim that some words are characteristic to positive or negative feeling, so that we can label a tweet as soon as we see that it contains one of these $characteristic\ words$:

$$S = \{\, w \mid w \in C \ \text{ and } \ pertinence(w) = 1 \ \text{ and } \ occ(w) > S_{thres} \,\}$$

A characteristic word is associated to a feeling, either positive or negative. As an implication, it should be impossible to find two characteristic words with opposite sentiment in the same tweet. We define S_{thres} to be the minimal nmber of occurences such that two opposite characteristic words cannot be found in the same tweet, training set included.

Capturing the semantic of only individual words would mean that a tweet whose words are shuffled in any order would still be labeled the same. This is not acceptable regarding to negation, because it is quite apparent that negation sentiment semantic is strongly dependent on the attached word and vice versa. For instance, in sentence ”I don’t like it.” the positiveness of ”like” should be compensated by ”don’t”, but ”don’t” is not intrinsically negative. Indeed, ”I don’t worry” should be labelled positive. We therefor should be able to indicate that ”don’t” is attached to ”like.”
This is achieved by using n-grams, whose use as already be made by [V] [VI]. From the sets constructed above, and adapted to n-grams, we train the GloVe model to create a semantic vector of nb_{dim} dimensions for each n-gram.We consider bigrams and trigrams, as sequences of two and three successive words in a tweet respectively.
The GloVe model is thus fed with unigrams, bigrams and trigrams.

Once n-grams vectors of features are computed, we still have to create the feature of one single tweet.

we developed two ways of doing it. The first one, is the simplest and is done by the function `contruct feature`. This function simply takes the weighted average, accordingly to the pertinence of all the word, n-gram, (bringing by the relevant vocab) of the words’ features representation present in a particular tweet.
The other way of creating the feature of a tweet based on the featured words that it contains is explained in the following. A weighted-by-preference average on relevant n-grams feature vectors has been tried, as well as a concatenation of the feature vectors of the nb_{concat} relevant n-grams having the most important pertinence. Note that if no relevant n-grams are found, we use non-relevant n-grams. Results of these two methods will be discussed in section [X].

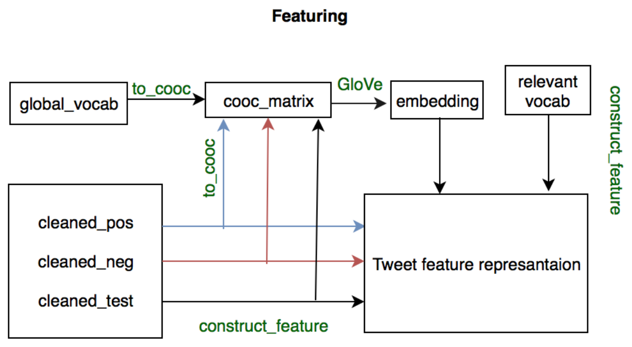

Figure 2. Explicit featuring Pipeline. In green are the function that are used to transit from one entity to another.

To select the best ways of constructing features we runned several times the . To be fair in our judgment, we also considered the time it took to run them and derived a fair criteria which is simply given by the ratio of both. i.e. $\frac{accuracy_{gain}}{efficeny_{gain}}$.
A table bellow is showned for the two of them

C. Labelling

The labelling is the process of attributing positive or negative sentiment to tweets, given their tweet features.

1) Labelling Model: A tweet can be labelled differently in function of its feature vector, which is supposed to capture its semantic. . Finally it is taken for granted that two tweets exactly similar can be labelled the same.

2) Labelling Methods: The function mapping tweet features to sentiment is approximated by running a neural network with a ”Adam”[VII] stochastic gradient descent agorithm adapted to large datasets. The fist layer has a size equal to the size of a feature vector, and the last layer has size 2 as we want to achieve binary classification.
Once labelling is done by neural network, labelling of tweets containing characteristic words is enforced to be the same as the matching characteristic word.
As a final step, tweets being duplicates of training data set tweets are given the same label as their duplicates.

III. Result

In this section, we discuss the overall results we had, and where we could made improvement on the accuracy, but also on the computational power. The computational power is an important criteria and should not be put apart.

First of all, stemming and bringing back to the root the tweet was a good improvement in both computational power and accuracy result.

With this model, our gain was about 20% times faster for the labeling part. This is explained by the fact that we dropped the duplicated tweets and that some of the words kept the same semantic after stemming them and was mapped to one representative. We then no longer had to build the feature representation of all of those words since they where mapped to the representative.

An improvement we made on the accuracy but had counter part in the computational power is obviously the use of the n-gram. The flavor that we had in the beginning was confirmed but by expanding the tweet with the n-gram we had to

We could also improve the the labeling part of our prediction by the choosing the adam optmizer instead of a traditional SGD optimizer. The adam optimizer became to be good optimizer for predicting with a lot of training tweets.

In the overall, we could bring small but at the end a cumulated big improvement in each part of the classification process. This prove

Abstract
: Short description of the whole paper, to help the reader decide whether to read it.

Introduction
: Describe your problem and state your contributions.

Models and Methods
: Describe your idea and how it was implemented to solve the problem. Survey the related work, giving credit where credit is due.

Results
: Show evidence to support your claims made in the introduction.

Discussion
: Discuss the strengths and weaknesses of your approach, based on the results. Point out the implications of your novel idea on the application concerned.

Summary
: Summarize your contributions in light of the new results.

IV. Tips for Good Writing

The ideas for good writing have come from [1], [2], [3].

A. Getting Help

One should try to get a draft read by as many friendly people as possible. And remember to treat your test readers with respect. If they are unable to understand something in your paper, then it is highly likely that your reviewers will not understand it either. Therefore, do not be defensive about the criticisms you get, but use it as an opportunity to improve the paper. Before your submit your friends to the pain of reading your draft, please *use a spell checker*.

B. Abstract

The abstract should really be written last, along with the title of the paper. The four points that should be covered [2]:

V. Summary

We saw throughout this project that there are several important aspect while building a prediction. It's not just about building a classifier and learn directly on the raw data. A big part is done by processing those raw data in order to give them a shape such that our classifier will extract the best of them. The other important part to create the feature of the data in order that they fit the classifier. There are several way and it can be tricky to pick the best one.

References

[1] Editorial, "Scientific writing 101," *Nature Structural & Molecular Biology*, vol. 17, p. 139, 2010.

[2] S. P. Jones, "How to write a great research paper," 2008, microsoft Research Cambridge.

[3] G. Anderson, "How to write a paper in scientific journal style and format," 2004, http://abacus.bates.edu/ ganderso/biology/resources/writing/HTWtoc.html.