CS 735/835 Information Retrieval Project - Evaluation

Shayan Amani
(993550898, sa1149)

Soheil Gharatappeh
(933639024, sg1147)

1 Introduction

Twitter is one of the most famous social apps that gives this opportunity to its users to communicate their ideas via text-based messages. These feature has made twitter a unique platform, for doing all sorts of language/text-based analysis. In order to understand the importance of this research, just think about the fact that the president of the most powerful country in the world, uses twitter on a daily basis to convey his messages and ideas to the world. It is a crucial piece of information for a political strategist, to have an understanding of the true feeling of a user about a particular issue, to come up with ideas in their negotiations to solve, for example, international disputes, or make deals between countries. The outcome of our analysis is not limited to only the world of politics. Knowing the users' emotion (at the moment of writing the tweet, or about a specific topic in general), can be useful for businesses and marketing strategists to better sell their products. By knowing about emotions of people towards something in a specific area, we can come up with a better business plan that works best for that society. We can adjust advertisements produced for them to their tastes, and use elements that can better convince them to buy that product.

1.1 Task Definition

The problem to be solved is called *Affect in Tweets*, and it is defined as how we can determine the user's emotions according to their tweets to know what sort of feelings they have, and to what degree they have those feelings. Four emotions are studied here are semantically distinct emotions; *anger, fear, joy, sadness*. Our task is; given a tweet, and an emotion X, we have to determine the intensity or degree of emotion X felt by the speaker. The data we have in our output is a value between 0 and 1, in which the maximum value of 1 stands for feeling the maximum amount of emotion X, and the minimum score of 0 stands for feeling the least amount of emotion X. This can be interpreted as, when the user is having a maximal/minimal mental state toward/away from emotion X.

In this project, for sake of simplicity, we only consider one emotion; joy, mainly because we all are looking for it in this day and age. Hopefully, this

research will help us to know the underlying meaning of the language we use every day.

1.2 Methods

To solve the problem of finding users emotion given a tweet using conventional IR methods, there are three possible approaches; using a *Vector Space Models*, and *Language Models*, or using *Regression* methods. Regardless of the method we are going to implement, we need a document that somehow plays a reference role for happiness. If we look at this as the similarity perspective, the document shows the direction of *pure* happiness in our space. So, our job is to find the similarity of tweets to the reference document, to decide whether the document points to the same direction as the reference or not. In section 3.1, we discuss how we obtained our happiness reference document.

For the vector space model approaches, similarities between the happiness document, and a given tweet are calculated. Among all $tf-idf$ methods lnc.ltn, bnn.bnn, anc.apc are picked to be examined. These metrics, intuitively show how similar tweets are to the happiness document. In addition, some language model methods such as; *Laplace, Jellinek, and Dirichlet* Smoothing are applied to the problem to compare the results with $tf-idf$ based approaches.

2 Related Work

In this report, emotion identification using vector space model and language model, in addition to a SVR based model is studied. The effect of choosing different $tf-idf$s has not been studied in the literature. However, in [6], Kun-Lin et al. proposed a smoothed language model method for Twitter sentiment analysis. In [3], Amati et al. proposed a probabilistic model for Information Retrieval. Alec Go et al. [5] presented a twitter sentiment sentiment classification in 2009 using distant supervised learning. Andryushechkin et al. [4] suggested an SVR based solution to detect emotion intensity in an emotion recognition problem.

3 Approach

3.1 Happiness Reference

To understand how much a tweet is happy, we need to have a good reference document. If we, somehow, can generate this document, we can make comparison between the words in the given tweet, and our reference and say how much they look similar. We can make this document by adding "happiness" synonyms to our document. So, we searched the keyword "happy" on the website http://www.thesaurus.com, and add all of relative synonyms to our database. Next, we opened each of the synonyms, and added them to our database (duplicate word have not been removed on purpose).

To boost up our database, we then used another helper database. This document is taken from the competition Multilingual Emoji Prediction [1]. In the provided database, they associate each tweet to an emoji. So, we took tweets that are associated with happy faces emojis, namingly; smile, tears of joy, wink, and etc. and add them to our database. The idea of improving our database using other tweets is mainly based on the fact that the language used in tweets are different than what we can find in dictionaries. So, if we only use dictionaries to obtain words related to happiness, we cannot guarantee that we have a reasonable database that contains vocabularies used by people in their social life that indicates their happiness.

To enrich the database even further, Twitter API was employed. This API lets users to search for a specific keyword in the last available tweets. The powerful interface of Twitter API also gives users this option to choose the language of retrieved tweets. So, the retrieved tweets can be easily added to the database, with no further filtering.

Here, it is assumed that if a tweet contains some happy words such as; "haha", "Lol", ":)", the user was happy when generating the tweet. So, all the words in the tweet can be conceived as happy words and added to the database (which is a naive assumption).

In 1, which is explained further in section 4, 5 different dataset obtained with different keywords are inspected. For instance, bnn.bnn is applied to the main joy dataset, which is placed under the "Thesaurus" row. And, in the next row, the result of bnn.bnn algorithm for adding some other tweets to the joy dataset that are obtained by searching the keyword "haha" in the tweeter API is put. This will be repeated for the next rows with another keywords, that are mentioned in the table. In the next columns, the same analysis is done using Laplace and Jilinek-Mercer smoothing.

3.2 Vector Space Model and Language Model

In the previous section, we discussed how we can obtain a document that shows the direction of happiness in space.

As it is noted on the competition website, the reported scores in the training and evaluation database have no meaning by itself. Therefore, it should be seen as a relative value between [0,1], considering the explanation given in 1. And, as we know, none of the above-mentioned methods give us back a value between [0,1]. So, what we need to do is to normalize them into the interval of [0,1], and do the evaluation on them. Score files [1] and the original dataset[2] can be found here. They are formated as:

```
1 id[tab]tweet[tab]emotion[tab]score
```

Six different VSM and LM methods were selected for making comparison between these different methods. In addition, these information will be used as

[1] under the name structure of `runfile-affects-<method>`

[2] `EI-reg-en-joy-train.txt`

features to train the SVR model. Please find the implementation for this part of the project in here.

Among these 6 methods, 3 worked better; bnn.bnn, Laplace and Jelinek-Mercer. bnn.bnn uses boolean term frequency for both document and query, and it does not consider document frequency (idf). In addition, it has no normalization criteria. It is noteworthy that probably the fact that it has boolean approach to the term frequency is the reason of it's superiority in comparison to lnc.ltc in this project. Because tweets are only 140 characters, and if we consider a logarithmic tf for them, this will definitely affect our result in a negative way.

In Laplace language model smoothing, the basic idea is to add "one of each" different event to our urn, and make new urn and pick from the new urn we have.

$$P(t|d) = \frac{n(t,d)+1}{\Sigma_{t\prime} n(t\prime,d) + |V|} \tag{1}$$

And, for Jelinek-Mercer, λ events are chosen from the main document, and $(1-\lambda)$ events are chosen from the main corpus, and the final evaluation is done on the newly made urn with the following probability distribution:

$$P(t|d) = \lambda \frac{n(t,d)}{\Sigma_{t\prime} n(t\prime,d)} + (1-\lambda)p(t) \tag{2}$$

3.3 Regression and Support Vector Machine

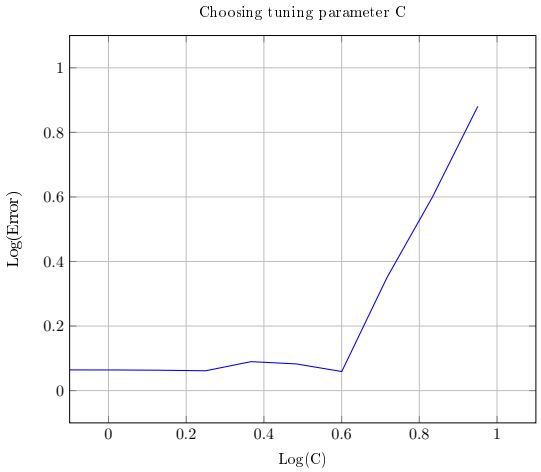
The second possible approach is to look at this problem as a regression problem. Firstly, it is noteworthy to mention that this is not a normal IR problem. IR problems can be categorized as classification problems, in which a user issues a query, and the system tries to show some relevant documents to the user. So, at the end, the retrieved documents can be *relevant* or *non-relevant*. Consequently, the ground truth contains information about the fact that the retrieved documents were relevant or not. However, here, the problem is not to determine the fact that a tweet is happy or not. But, the problem is to find out to what level a tweet is happy.

This problem can be seen as a regression problem; in which a set of training data is given, and a score (an indicator of the intensity of an emotion) for a new given tweets is asked. This is basically an SVR problem. The first task in an SVR (support vector regression) problem, is to come up with a good feature matrix. It is reasonable to use $tf - idf$, and language model data as features. These IR methods reflect a relation between a query and a document. So, they can be a good indication of the fact that whether the query vector was similar to the document or not.

To train a regression model based on SVR, 60% of the data is taken out as the training set. Then, the rest 40% is also divided into two sets; validation set and test set. The validation set is then used for choosing tuning parameter (C), and then the error of fully trained model will be calculated on test set. According to figure 3.3, C is chosen to be $1.5 * 10^5$.

Implementation of the regression algorithm for this project is done on Python. Sklearn package is particularly used to train an SVR model, based on a *Radial Basis Functions*(RBF) kernel. The source codes for this part can be found in here.

Apparently, because there is no meaningful correlation between the features and the scores, the trained model could not perform well, and showed a bad result in Pearson correlation test (-0.002).



4 Evaluation

The idea of this project is taken from the competition published in [2]. To find more details about their research, please refer to [7]. The training and development sets are available online in the given website in 3 languages; English, Arabic and Spanish. The following is a sample line of the data set to measure anger in the given tweet:

```
10004 Don't join @BTCare they put the phone down on you, talk
  over you and are rude. Taking money out of my acc willynilly! #
  fuming anger 0.896
```

The dataset, used as the ground-truth for this project contained 1616 annotated tweets. In general, 4 training datasets was available for 4 different emotions, and the dataset for this project is taken from the joy dataset.

The main evaluation measure used in this study is *Pearson Correlation Coefficient*. Pearson Correlation (equ. 3) is a measure of the linear correlation between two variables X and Y. It has a value between +1 and −1, where 1 is total positive linear correlation, 0 is no linear correlation, and −1 is a total negative linear correlation.

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{3}$$

IR method	bnn.bnn	Laplace	Jelinekmercer	Standard Deviation
Thesaurus	0.190	0.166	0.177	0.012
Thesaurus + “haha”	0.087	0.086	0.089	0.001
Thesaurus +“:)”	0.109	0.100	0.120	0.010
Thesaurus +“lol”	0.043	0.050	0.0493	0.004
Thesaurus + smiley	-0.024	0.120	-0.024	0.004
average	0.0812	0.104	0.082	0.007

Table 1: Correlations

To do evaluation, an implemented version of Pearson and Spearman methods, provided by the competition website is employed. Given a run file, one can run the evaluation code using the following command:

```
python evaluate.py <number-of-pairs> <file-prediction-1> <file-gold-1> ..... <file-prediction-n> <file-gold-n>
```

The results for the mentioned IR methods is put in table 1. In this table, the obtained results using different joy database is demonstrated. From all VSMs and LMs, only three methods are picked, since the other three showed unreasonable results. For instance, lnc.ltc did not work well for this project. A possible explanation for this could be the fact that lnc.ltc designed for a normal IR problem, in which the term frequency matters. However, here, the problem seems to be a Boolean retrieval problem. In other words, what matters for a tweet to be happy is the fact that it has a couple of happy words, not whether the term frequency of "A word" is high or low. In addition, taking logarithm from tf is not a good idea when we are dealing with only 140 characters, and we cannot have high tfs. In addition, as it can be seen from the table, increasing the document's size using a meaningful data, not only *does not* improve the results, but also causes lower correlations in the results. This is probably because of the fact that adding more vocabulary to the database, will add more noisy data to the system. And, because no noise rejection mechanism is employed in this project, the results are noisy and unreasonable. Among all these methods, *Laplace Smoothing* showed a better result in average.

5 Conclusion

In this research study, two different approaches to Information Retrieval were compared; Vector Space Model and Regression.These two methods have two completely different points of views in looking at a problem, one does not take any sort of feedback from the world, and only works on what we *assume* to be true about IR. However, the other one only works based on user's feedback.

Despite the first guesses, IR methods worked better here. The main reason for the bad performance of SVR was the poor quality of the features. In machine learning approaches, the most important key element of the solution is features.

Here, a large majority of potential features were neglected. Here, a long list of good features and filters that can help to solve this problem more effectively can be found.

6 Contributions

Both members contributed to the project academically. Shayan was more focused on the coding and packaging side of the project, and Soheil contributed to coding and writing this report.

References

[1] Semeval-2018 task 1: Affect in tweets. https://competitions.codalab.org/competitions/17344.

[2] Semeval-2018 task 1: Affect in tweets. https://competitions.codalab.org/competitions/17333#learn_the_details-overview.

[3] Gianni Amati, Cornelis Joost, and Van Rijsbergen. Probabilistic models for information retrieval based on divergence from randomness. Technical report, 2003.

[4] Vladimir Andryushechkin, Ian Wood, and James O' Neill. Nuig at emoint-2017: Bilstm and svr ensemble to detect emotion intensity, 2017.

[5] Alec Go, Richa Bhayani, and Lei Huang. Twitter sentiment classification using distant supervision. 150, 01 2009.

[6] Kun-Lin Liu, Wu-Jun Li, and Minyi Guo. Emoticon smoothed language models for twitter sentiment analysis. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, AAAI'12, pages 1678–1684. AAAI Press, 2012.

[7] Mohammad Salameh Saif M. Mohammad, Felipe Bravo-Marquez and Svetlana Kiritchenko. Semeval-2018 task 1: Affect in tweets. *Proceedings of International Workshop on Semantic Evaluation (SemEval-2018*, 2018.