Emotion classification from text (chat)
Course Project, CS-725
Spring 2015-16

Ravi Shankar Mondal, 133059003
Saurabh Jain, 143050048
Srikrishna Khedkar, 143050055
Sushant Mahajan, 133059007

April 30, 2016

Contents

List of Figures

List of Tables

1 Project Description

1.1 Problem Statement

Detecting emotional state of a person by analyzing a text document written by him appear challenging but it is also essential many times due to its various application area. Recognizing the emotion of the text plays a key role in the human-computer interaction. Emotions may be expressed by a person's speech, face expression and written text known as speech, facial and text based emotion respectively. Athough sufficient amount of work has been done by researchers to detect emotion from facial and speech information but recognizing emotions from textual data is still a fresh and hot research area. In this project, we have addressed the problem of emotion detection on the basis of text data only.

1.2 Why is it interesting

- Psychologists can better assist their patients by analyzing their session transcripts for any subtle emotions.
- Reliable emotion detection can help develop powerful human-computer interaction devices. For example,a computer can analysis the emotion during a text chat session and according display proper emoticons or a computer can read a movie review and then analyze the sentiment of the reviewer.
- Deep emotional analysis of public data such as tweets and blogs could reveal interesting insights into human nature and behavior

Consider the following scenarios:

Sentence	**Emotion**
•A close person lied to me	Sadness
•A colleague asked me for some advice and as he did not have enough confidence in me he asked a third person.	Sadness
•I got a present today.	Joy

Table 1: Some examples

2 Papers

We will be closely following the paper Taner Danisman and Adil Alpkocak titled **Feeler: Emotion Classification of Text Using Vector Space Model**

3 Dataset

The paper mentions the ISEAR dataset, which is composed of around 7500 annotated sentences. The dataset has around 7500 sentences. We will use $2/3^{rd}$ of the dataset for training and cross validation and the rest for testing.

4 Different Methods Applied to this problem

4.1 Vector Space Model

- As mentioned in section 2, in line with the paper, we will be using VSM for out initial implementation.
- Put simply, we will have representative vectors for each of our emotion classes, **joy, anger and sadness**. Then we will transform our input sentence in the same way we found our representative vectors and find the cosine similarity with each of them.
- These similarities will be used to calculate probabilities of the input being in some emotion class and the most probable class will be output.
- The vectors will comprise of weights in terms of **tf-idf** frequencies. We could simply use word frequency instead but:
  - **tf-idf** was chosen to provide weights as some words occur sparsely in the document, but provide definitive information about the emotion being portrayed. For example, the word *devil* occurring in a sentence tips the scale towards negative emotion.
  - Had we chosen only frequencies, this could pollute the result.
  - Similarly, some words occur very frequently like *cat*, but do not provide much information. Frequentistically speaking these words will get higher weight.
  - **tf-idf** balances these disparities by multiplying the term frequency with inverse document frequency. The latter meaning the ratio of total number of documents and the number of documents containing that term.
  - For low frequency words, the **idf** will be high and vice-versa.

High level algorithm Training

- The training set is cleaned, in that:

  - Stop words are removed, after comparing with standard stop word lists.
  - Apostrophes are expanded.
  - Words are reduced to their root using some stemming technique.

- We build a lexicon from $2/3^{rd}$ of the above cleaned data.
- Each term in the vector corresponds to a term in our lexicon. The lexicon is an ordered set of all words in the cleaned dataset.
- We build document vectors using weights assigned to each term. These weights are calculated using the **tf-idf** technique. These weights are further normalized.

$$w_{ki} = c(t_k, d_i) = \frac{tf_{ik} log(N/n_k)}{\sqrt{\sum_{k=1}^{n} (tf_{ik})^2 [log(N/n_k)]^2}}$$

  where,
  t_k = k^{th} term in document d_i
  tf_{ik} = frequency of the word t_k in document d_i
  idf_k = $\log(\frac{N}{n_k})$, inverse document frequency of word t_k in entire dataset.
  n_k = number of documents with word t_k.
  N = total number of documents in the dataset.

- Now each emotion class is a set of the corresponding vectors, calculated as above. We then find the mean of all these vectors to find a representative vector of the said class.
- Finally, we'll get s vectors corresponding to the distinct emotion classes.

Testing

- Each input sentence is cleaned, and transformed into a vector with weights calculated using **tf-idf** technique.
- The similarity is calculated as:

$$\text{sim}(Q, E_j) = \sum_{k=1}^{n} w_{kq} * E_{kj}$$

  where,
  Q is query vector,
  E_j is an emotion vector.

- Answer is calculated as:

$$\text{VSM}(Q) = \text{argmax}_j(\text{sim}(Q, E_j))$$

4.2 Support Vector Machine

Quite similar to the above approach, this class also composes tfidfhelper.py. There is no need here to build emotion class wei ght vectors, hence that method is not used. However, "fit" and "predict" are present and work exactly the same way. We have directly used Linea rSVC from sklean API and tuned the parameters.

4.3 Gaussian Naive Bayes

Since text classification is a popular application of naive baye's algorithm, we used it for our predictions too. However, our data is continuous (tf-idf) so we have used Gaussian Naive Baye's to classify the text. Again as in case of svm we have similar API and used Ga ussianNB from sklearn libraries.

5 Ideas Used From Class

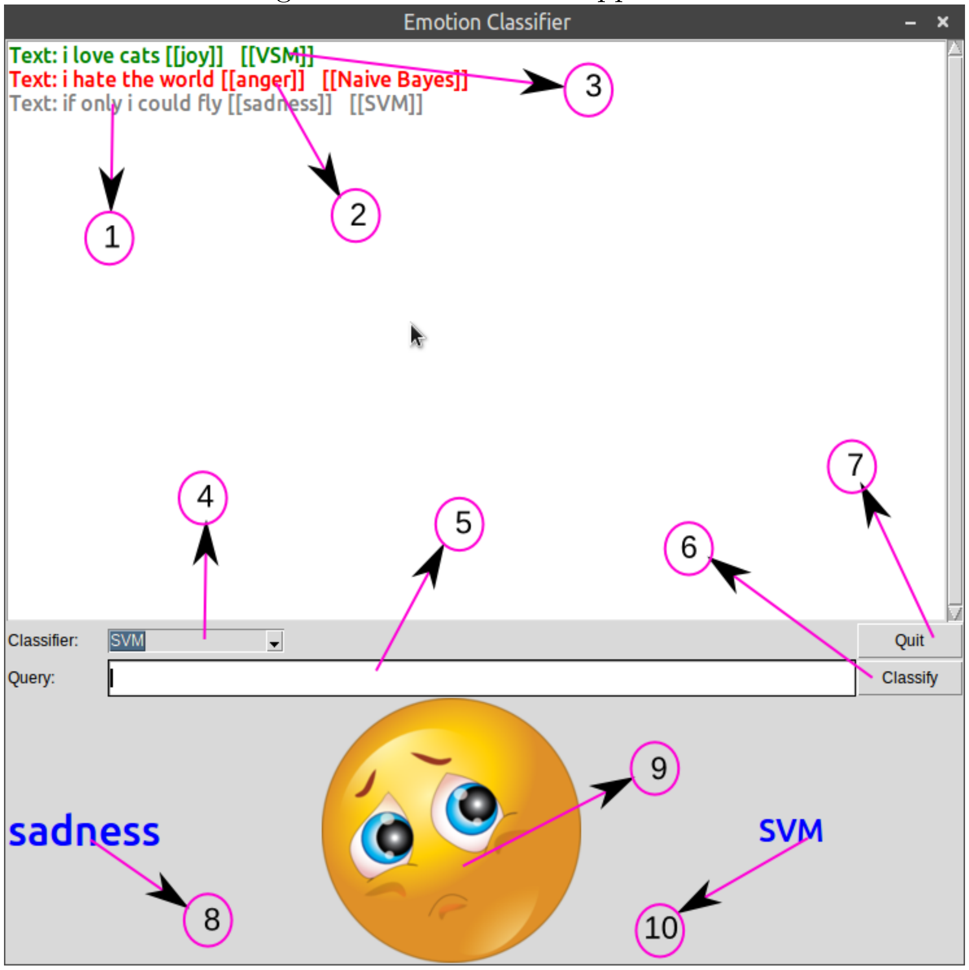
- Apart from Vector Space Model, we have also used SVM and gaussian naive bayes classifier to solve this classification problem.
- We have measured the overall F1 score of the prediction to compare the performance of different classification algorithms.

6 Concepts Learned in this project

- Learnt Vector Space model which is an algebrical model for representing text document as a vector of index terms. This is an widely used technique in information retrieval.
- Although we studied SVM and Gaussian naive bayes classifier in our class, in this project we learnt how to employ them in a practical problem like detection of emotion from text.
- During the course of this project, we also became familiar with tf-idf (term frequency-inverse document frequency) weighting scheme. This is an well known technique in information retrieval which gives us an idea about the importance of an word to a document in a corpus.
- We also had a hands on experience with Weka which is a popular suite of machine learning algorithms. We used this tool mainly to visualize the training data and to test some standard classification algorithm.

7 Application Details

Figure 1: Text classifier application



Text: i love cats [[joy]]   [[VSM]]
Text: i hate the world [[anger]]   [[Naive Bayes]]
Text: if only i could fly [[sadness]]   [[SVM]]

No.	Element	Explanation
1	Text (String)	Text to be classified.
2	Text (String)	The prediction made by classifier.
3	Text (String)	Classifier that made the prediction.
4	Combobox	Drop down showing available classifiers.
5	Entry (String)	Data entered by user to query.
6	Button (String)	Button to classify and print on the textbox.
7	Button (String)	Button to exit the application.
8	Label (String)	Large sized font - predicted emotion.
9	Label (Image)	A smiley depicting the emotion.
10	Label (String)	The classifying classifier.

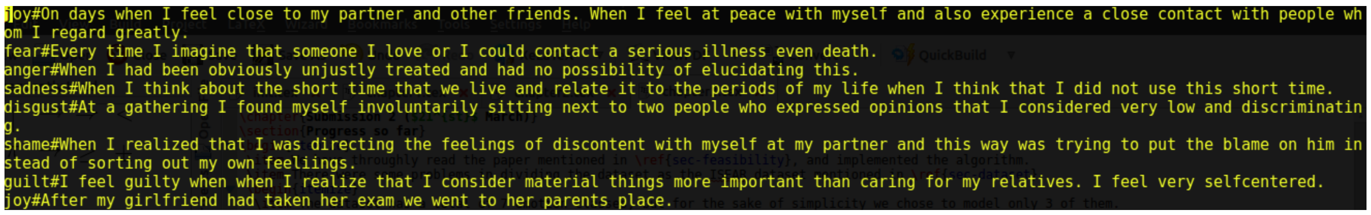
Table 2: Explanation of app features

8 Results

- We have throughly read the paper mentioned in 2, and implemented the algorithm.
- There were some problems in dividing the dataset as the ISEAR dataset mentioned in 3.
  - Has a total of 7 emotion classes, but for the sake of simplicity we chose to model only 3 of them.
  - Out of the 7 classes, 6 are negative emotions (**sadness, guilt, shame, fear, anger, disgust**) and only 1 positive (**joy**).
  - We had 2 choices now, either to divide the leftover data for 4 classes into the 2 negative ones, we've chosen (**anger,sadness**).
  - Doing this caused a lot of misclassification, so after some research, we've currently left out 2 emotions - **guilty,fear**.
- We wrote Python code to correctly implement the vector space model algorithm mentioned in 4.1. We are working collaboratively and have set up a github repo to house all the data, report and code. The link is `https://github.com/sm88/mlproject`.
- Due to the above mentioned reason for less data, we are looking into some more datasets, mentioned in the paper namely, the Wordnet-Affect dataset and the Semeval dataset.
- The wordnet database is not available easily as we need to fill out a form, which after review will enable us to download.
- We have throughly cleaned the dataset, which initially had many problems. A few major ones that we solved are:
  - Many sentences were of form "No Response", which provides no information whatsoever.
  - There were some scattered braces and square brackets as well as some non-ascii characters, that took a while to catch.
  - Some sentences were repeated for multiple classes.

8.1 Implementation related results

Following are the results of applying the predictions to training data itself. Each row specifies the distribution of the samples belonging to a specific class. We trained on on 8397 sentences (documents) and then tried predicting them using our classifiers.8.1).



```
joy#On days when I feel close to my partner and other friends. When I feel at peace with myself and also experience a close contact with people wh
om I regard greatly.
fear#Every time I imagine that someone I love or I could contact a serious illness even death.
anger#When I had been obviously unjustly treated and had no possibility of elucidating this.
sadness#When I think about the short time that we live and relate it to the periods of my life when I think that I did not use this short time.
disgust#At a gathering I found myself involuntarily sitting next to two people who expressed opinions that I considered very low and discriminatin
g.
shame#When I realized that I was directing the feelings of discontent with myself at my partner and this way was trying to put the blame on him in
stead of sorting out my own feeliings.
guilt#I feel guilty when when I realize that I consider material things more important than caring for my relatives. I feel very selfcentered.
joy#After my girlfriend had taken her exam we went to her parents place.
```

Figure 2: Cleaned partial ISEAR dataset

	anger	**sadness**	**joy**	*Accuracy %*
anger	1745	469	201	72.25
sadness	265	1832	258	77.70
joy	60	1832	151	85.48

Table 3: Confusion matrix for training set using vsm

	anger	**sadness**	**joy**	*Accuracy %*
anger	2342	65	8	96.97
sadness	70	2263	22	96.09
joy	12	18	1424	97.94

Table 4: Confusion matrix for training set using svm

	anger	**sadness**	**joy**	*Accuracy %*
anger	1829	422	164	75.73
sadness	5	1987	363	84.37
joy	0	1	1453	99.99

Table 5: Confusion matrix for training set using nb

Following are some numbers based on unseen data for the 3 classifiers:

	f1 score	**accuracy %**
vsm	0.436	44.24
svm	0.403	40.93
nb	0.459	47.67

Table 6: f1 score and accuracy on unseen data

9 Conclusions

- We found that gaussian naive baye's outperformed the other 2 classifiers. VSM came in a close second.
- The result is a bit expected as NB is the goto algorithm for text classification. Since we are using continuous valued attributes (tf-idf weights) GNB was an obvious choice.
- In our opinion SVM performed poorly because of the nature of the data. First of all it was sparse and secondly there was quite a bit of overlap in the data.
- If a word is not present in the lexicon, the prediction accuracy suffers highly.
- In VSM we tried other measures of similarity but again due to the nature of data reverted back to cosine similarity, as it is considered best for sparse data. Other measures like euclidian distance, rbf-kernel are also used but cosine similarity outperforms them over sparse data.

10 Future Work

- We can try and find more patterns in the dataset like the one shown in the paper. There they've created new words from negative verbs. For example, the input sentence "I don't love you." is transformed to "I do not love you." which is again transformed to "I do NOTlove you". Here we see that finally a new word *NOTlove* has been created. Authors claim this helps increase accuracy.
- We can find set differences between various classes on basis of words (words unique to each emotion class), and remove them from stop words list for accurate predictions.
- We can implement naive baye's classifier using bi-gram, tri-gram etc approaches.
- We can also do sentiment analysis of our dataset where we will just check the accuracy of negative and positive emotions(binary classification).
- We can apply some unsupervised learning technique in this problem which will be useful when availability of large annotated dataset is very low.

11 Pointers to literature

- The wordnet-affect dataset can be accessed from `http://wndomains.fbk.eu/wnaffect.html` although we need to fill out some forms before we are granted a download link.
- The Semeval task 14, is a medical dataset available at `http://alt.qcri.org/semeval2015/task14/index.php?id=data-and-tools`.
- Multinomial naive baye's using **tf-idf** technique can be found at `http://sebastianraschka.com/Articles/2014_naive_bayes_1.html`.
- As mentioned in section 10, SVM could be useful for classification. The paper at `http://www.cs.cornell.edu/people/tj/publications/joachims_98a.pdf` presents a good use case.
- For naive baye's we could also add the chi-squared test for confidence computation. Some details are outlined in `http://www.dis.uniroma1.it/~leon/didattica/webir/IR11.pdf`.
- Code listing is attached and is publicly available at `https://github.com/sm88/mlproject`.