Mood - Based Activity Recommendation System

Prajakta Gaydhani (pag3862)
Virtee Parekh (vvp2639)

ABSTRACT

In this application, a chatbot is developed that detects the user's mood and suggests corresponding activities like jokes, books to read, movies to watch, shopping portals etc. so as to improve the mood of the user (if the user is in a negative mood) or to enhance the mood of the user(if the user is in a positive mood). The application is developed on IBM Bluemix and uses IBM Watson's services like Conversation and Tone Analyzer.

1. INTRODUCTION

The primary aim of this project is to detect the mood of the user by analyzing the input text and recommend various activities to improve (for negative emotions) or enhance (for positive emotions) the mood. The recommendations include suggestions on books to read, movies to watch, jokes to lighten the mood and e-commerce websites for retail therapy. For example, if the system detects that the user is sad, he/she will be prompted with suggestions for light hearted movies like comedies, motivational books, short jokes. If the user's mood is detected as happy, he/she will be recommended websites for shopping, short jokes, movies and books. If the user is in a good mood, the system will not recommend books or movies that belong to genres like war, dystopia, horror etc. Examples from genres like drama, comedy, fantasy, romance will be suggested.

The project is developed using IBM's Platform-as-a-Service platform Bluemix that deploys Watson to develop a question-answering machine. We used the Tone Analyzer [4] API from Watson that uses cognitive linguistic analysis to detect and identify the mood of the input text. Tone Analyzer detects three types of tones, including emotion (anger, disgust, fear, joy and sadness), social propensities (openness, conscientiousness, extroversion, agreeableness, and emotional range), and language styles (analytical, confident and tentative) from text. In our project, we are focusing solely on the emotions recognized by Tone Analyzer. We also made use of the Conversation API provided by Watson. Conversation [3] uses machine learning and natural language processing to deploy a chatbot that can interact with the user.

In the upcoming sections, we first talk about motivation for this project. In Section 3, we explore other applications that deal with the concept of mood detection. In Section 4, we talk about datasets used and data preprocessing techniques used. The implementation details are explained in Section 5 followed by the challenges and limitations faced while implementation. This is followed by ethical considerations regarding the concept of emotion detection, future work in this domain and the conclusion.

2. MOTIVATION

Experiencing negative emotions is not a rare thing. We have all faced dozens of situations that have a surge of negative emotions like hatred, anger, fear, frustration etc. in our minds. There are steps that can be taken to control our reactions to keep the negativity at bay. However, not every individual can stay emotionally calm all the time and even the calmest of people do get upset. So the main issue is to be able to deal with the negative vibes and to be able to achieve a state of calmness that allows us to make rational decisions. Negative emotions are toxic and extreme negative emotions can affect the physical health adversely. Impacts of negative emotions include hypertension, cardiovascular diseases, high blood pressure etc. Negative emotions are detrimental to an individual's productivity and rationality.

This application is a small step in trying to help the user in alleviating his misery or enhancing his positive mood. Recommendations are made in such a way that they will provide a decent distraction for the user and will allow the user's mind to calm down. If the user is already in a positive mood, recommendations are tailored such that, it reinforces the user's positive attitude.

3. RELATED WORKS

The concept of emotion detection is getting immense popularity not only in the field of sentiment analysis but also in the field of health care, wearables , education and customer satisfaction [7]. Emotions can not only be detected from text, but also from face, speech, photos and videos. Main types of emotions that are detected from these sources are angry, fear, disgust, joy, sadness, surprise and Contempt [7]. This paper mainly focuses on identifying emotion from text.

One approach in extracting emotion from text is using the Receptiviti Natural Language Personality Analytics API. It uses basic of AI, Machine learning and Natural Language processing concepts to detect the emotions. The Receptiviti uses a set of emotions and words to detect sentiments from the text [7].

Another approach uses BiText [1] to analyze emotions from the text. BiText can identify only positive and negative polarities. While assigning different polarities, BiText takes into consideration the structure of the

sentence. If two sentences contains 'not sad' and 'happy' words respectively, both these sentences are assigned a positive polarity by BiText. Along with polarity, BiText assigns a sentiment score to every sentence [1]. For assigning sentiment score, BiText takes into consideration the intensity of words in the sentences. The sentences that contains words such as very, strongly are given a high sentiment score. Along with this Bitext identifies the main topic of the sentiment [1]. Bitext is widely used in detection emotions of the users from twitter tweets [1].

Another approach uses a Microsoft Azure Cognitive Services to analyze emotions in the text document [2]. The model analyses the emotion of overall document and not of a particular text in the document. The system takes text document as a input and returns a score in the range of 0 (negative) to 1 (positive) [2]. The model is trained by providing a set of text documents. Before that, the text in the documents is preprocessed using stop words removal and parts-of-speech tagging. For calculating the score for each sentence, it uses a classification algorithm. If the sentence is evaluated to a score closer to 1, then that sentence is indicated as a positive sentence otherwise the sentence is considered to be a negative sentence [2]. Overall emotion score for text document is then computed from the individual score of the sentences that constitutes the entire document.

The next approach uses a emotion ontology to detect the emotions from a text [13].The method maps a hierarchy of emotions class such as fear, disgust, anger, surprise, joy and happy to emotion ontology [13] . The emotion ontology contains a class and sub-class structure [13]. Emotion classes that are at low level in the hierarchy forms a sub-class in the emotion ontology and remaining forms the parent class. The actual emotions are detected using an emotion detection algorithm [13]. The algorithm calculates weight of the emotions at every hierarchy and then the emotion having highest weight is assigned as the emotion of the text [13].

Another approach uses Qemotion API's [14] to detect the emotions in the text and is widely used for customer satisfaction. The model identifies the emotion from text in the form of temperatures [14]. If the emotion detected is anger, it assigns a temperature in the range of -14 degree to -6 degree. If emotion detected is joy, it assigns a temperature in the range of 31 degree to 40 degree [14].

4. DATA DESCRIPTION AND PREPROCESSING

There are three data sets that are used in the application. The MovieLens [8] dataset is used to recommend movies to the users according to the genres. The MovieLens dataset comprises of around 9000 movies. The MovieLens dataset has three attributes - movie id, movie name and movie genre. The movie-id gives a unique id to every movie. The movie name specifying the name of the movie and movie genre specified the type of movie (example : Comedy, Horror or Romance). The dataset was all cleaned with no missing values and noise and so cleaning was required. Detailed description of the attributes is given in the table 1.

Another dataset that we are using in our application is a joke dataset [5]. This dataset had originally around 200,000 jokes but for the scope of this project, we considered only top 1000 jokes from the dataset. The joke dataset has two attributes joke id and joke. The dataset obtained was cleaned with no missing values and noise so cleaning and preparation was required. Detailed description of the attributes is given in the table 2.

Table 1: Attribute Description of Movie Dataset

ID	uniquely identifies the movie
Name	Name of the movie
Genre	genre of movie

Table 2: Attribute Description of Joke Dataset

Attribute	Description
Joke Id	Uniquely identifies the joke
Joke	Actual Joke

The third dataset that we are using is a books summary [6] dataset. The book summary dataset has around 12,000 books and had originally 7 attributes - wikipedia id, freebase id, book title, author, publication date, book genre (Freebase ID:name tuples) and plot summary. The books dataset was originally in .txt format separated by a tab. We converted .txt file to .csv file. For this we used excel-> Data -> Import option. After converting this file to a .csv file, the data obtained was not cleaned. The attributes like freebase id, plot summary, publication were not relevant for our project were removed. The genre column was composed of Freebase ID:name tuples. To remove freebase id from genre we used python's regular expressions. Detailed description of the attributes is given in the table 3.

Table 3: Attribute Description of Books Dataset

Attribute	Description
Wikipedia id	ID of book in wikipedia
freebase id	ID of book in the database
title	Title of the book
author	Authors of the book
publication date	Date of publication
genre	Genre of the book
plot summary	Overview of the book

For the online shopping website recommendations, we created a small list in python's consisting of popular websites URL's in categories like clothing, beauty, gadgets and household items.

5. IMPLEMENTATION

There are three main components to this application. Firstly, the chatbot conversation needs to be built in IBM Bluemix's Conversation Dialog. Secondly, using the Python API, a program is written that merges Tone Analyzer and Conversation. Lastly, in the program, at the correct instances, data is fetched from the datasets for activity recommendation and displayed to the user. All the steps are explained thoroughly in the following sections.

5.1 Training the Conversation Dialog

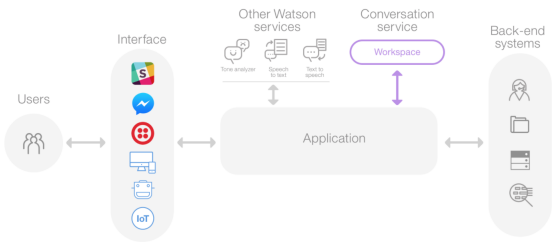

Figure 1: High Level View of Watson Services Implementation [12]

Watson Conversation allows users to build chatbots by defining intents, entities and dialog.

5.1.1 Intents

An intent is used to define the and identify the nature and purpose of the user's input [10]. An intent must be defined for each kind of user request that the application should support. An intent name is preceded by the hashtag symbol #. Multiple synonyms are provided so that Watson Conversation can identify the type of user input.

Consider the following scenario in our application. Any conversation starts with a greeting. We need Conversation to be able to identify the different types of user greetings. We define an intent called #Greeting. We then provide possible user examples like "Hi", "Hello", "Hey", "Yo", "Namaste", "Hola", "Howdy", "What's up", "Bonjour" etc. The more examples we provide, Conversation will detect the greeting more efficiently.

5.1.2 Entities

Entities represent objects that are relevant to the user's input and purpose [9]. They can be used to define the flow of action the Conversation must follow. Entities are defined with an @ symbol preceding the entity name,

Consider an flight booking application. Entities will be objects like @location, @date, @flightName etc. Conversation also provides built-in entities like @sys-currency (to detect currency values), @sys-date and @sys-time for date and time, @sys-person to identify user's name etc.

Our application does not make use of any entities.

5.1.3 Dialog

The Dialog uses intents and entities to provide a response to the user's input [11]. Dialog consists of nodes that contain a condition and a response. The condition can consist of an entity or an intent. If the condition is met, the Dialog responds with the responses that are provided by the user. After the response is printed, we configure Dialog as to what it should do now. We can either configure it to wait for a user input, go to the next node or jump to another dialog node.

Consider the #Greeting intent defined above. When the Conversation detects a greeting, we want Conversation to reply with an appropriate greeting. We create a Dialog node called, GreetingResponse. #Greeting forms the condition. The responses include "Hey! How was your day ?", "Hey! How's it going ?", "Yo! How's life?", "Hey! How have you been ?". While responding Dialog can randomly choose either of the responses. And finally, it waits for user input before proceeding.

5.2 Watson Developer Cloud and Python API

Once the user inputs how his/her day has been, the input needs to be collected, passed to the Tone Analyzer and values need to be fetched from the database accordingly. This is all done with the help of Watson Developer Cloud and Watson's Python API.

Watson Developer Cloud can be installed locally and it gives the developer access to all Watson services. It can be installed by using the following command:

sudo -H pip install –upgrade watson-developer-cloud

We then create Conversation and Tone Analyzer services on IBM Bluemix. Once the services are created, we obtain the credentials for accessing these services locally. Credentials include username, password, url, version and workspace id.

An instance of these is created in a Python program using the credentials. Instances are created using watson_developer_cloud.ConversationV1 and watson_developer_cloud.ToneAnalyzerV3.

There are two main methods used:

- tone() : This method returns a JSON object that contains the various tones detected in the input text and their respective scores. The JSON objects consists of all tones that were detected along with their scores. We only store the emotion that has the highest score and discard the other emotions. Scores range from 0 to 1, with 1 being the highest. For the input text "I am mad at her", the following values are returned by tone()

Table 4: Output Values of tone() for "I am mad at her"

Tone Name	Score
Anger	1.0
Disgust	0.931034
Fear	0.931034
Joy	0.0
Sadness	0.916667

- message() : This method returns a JSON object that contains details like Conversation's response to the input text, intent of the input text etc.

5.3 Recommending the Appropriate Activities

We get the intent from message() and mood from tone(). Activities are suggested based on the above two features. For example, if the input sentence has negative connotations and the mood detected is "fear" , then first suggestion is a joke. Then books and movies are suggested. Suggestions are done according to the emotion detected. In this case, the emotion is "fear". Therefore, all books or movies having genres like horror, thriller and war will be avoided and genres like comedy and drama will be chosen.

If emotion detected was "joy" and if the sentence has a positive connotation to it, corresponding activities will be suggested. To keep the user in the joyous state, suggestions for jokes and shopping portals are given. Books and movies

belonging to genres like comedy, romance and fantasy are suggested and genres like war, dystopia and horror are avoided.

6. LIMITATIONS

There were multiple challenges faced and we discovered quite a few limitations in using Watson's services.

- Failure to Recognize Negation in Input

  Watson's Conversation and Tone Analyzer fail to interpret sentences that have negation in them. An input sentence like "I am not feeling good" returns no results when used with both the services. This is a major loophole when it comes to Watson's natural language processing skills. Using negative logic in a sentence is quite common but Watson's services fail at recognizing it.

- Correct Grammar Required

  Watson's services require sentences to have correct grammar and words should not be misspelled. This is not a major limitation but not every user will have the domain knowledge about English grammar. Slight tolerance must be allowed, at least in the case of grammar.

- Issues with Watson's Conversation Dialog

  Watson's Conversation does not allow the same examples to be put in multiple Intents. Words like "okay" might belong to different input cases but for the scope of this project, we had to put it in one intent.

7. ETHICAL CONSIDERATIONS

When interacting with such a system, we give the system access to our emotions. Usually, our emotions guide our actions. It is unto the developer to not manipulate these emotions. The recommendation systems can serve a variety of purposes, both ethical and unethical.

Like the developed application, if the aim of the system is to relieve users of their sufferings, such recommender systems prove to be quite useful. Mood detection systems can be used to identify users suffering from mental disorders and suggest activities or remedies that will ease their pain and help them. These systems can also be built for the general public and recommendations can vary from foods to eat to keep you healthy to physical fitness tips.

However, these recommendations can be manipulated too. Consider a political party that is trying to turn the votes in their favor by finding out flaws in the actions of the opposing party. The party can use mood detection and find out users on social media that are upset with the opposing party. These users can then be targeted and flooded with information that will make them turn against the opposing party. The information provided to users can be true or just trolls.

Similarly, such sentiment analysis can be used by companies to analyze customer satisfaction. They may then limit the userâĂŹs experience by recommending products from the same brand over and over again and displaying ads everywhere on the userâĂŹs social media accounts, which can be considered unethical.

8. FUTURE WORK

We have developed a simple application that takes in user input, detects emotion of the user and recommends simple activities. This application can be used as boiler plate for several other applications. An helpful improvement that can be made is to improve the dialog flow and to create a more complex dialog. This would require creating a complex structure in Watson's Conversation. Suggestions of books or movies could be altered according to the user's choice of genres. this would also require modifying the Dialog to meet the need of complex queries. Other activities like foods that help reduce stress can be suggested or physical activities suitable to the user can be suggested.

9. CONCLUSION

Thus, the mood based recommendation system is a question answering based recommendation system that uses IBM Watson Tone Analyzer and Conversation API to detect emotions of its users in form of textual conversation. Depending upon the mood of its users, the system recommends different activities such as Movies, Jokes, Books and online shopping websites to enhance their mood without hurting their sentiments.

10. REFERENCES

[1] Bitext - nlp to enhance the understanding between humans and machines.

[2] How-to sentiment analysis in text analytics rest api (microsoft cognitive services on azure).

[3] Watson conversation.

[4] Tone analyzer - ibm bluemix, May 2016.

[5] Amoudgl. amoudgl/short-jokes-dataset, Mar 2017.

[6] D. Bamman and N. A. Smith. New alignment methods for discriminative book summarization. *CoRR*, abs/1305.1319, 2013.

[7] B. DOERRFELD. 20 emotion recognition apis that will leave you impressed, and concerned | nordic apis |, Aug 2016.

[8] F. M. Harper and J. A. Konstan. The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.*, 5(4):19:1–19:19, Dec. 2015.

[9] IBM. Defining entities.

[10] IBM. Defining intents.

[11] IBM. Dialog overview.

[12] IBM. Ibm cloud.

[13] S. N. Shivhare and P. S. Khethawat. Emotion detection from text.

[14] Sifium. Top five emotion / sentiment analysis apis for understanding user sentiment trends., May 2017.

APPENDIX

A. INDIVIDUAL CONTRIBUTIONS

- Virtee Parekh
  - Training the Conversation Dialog.
  - Developing the application using Python API.
- Prajakta Gaydhani
  - Developing the application using Python API.
  - Developing UI for the application.

APPENDIX

A. USER INSTRUCTIONS

- Create an account on IBM Bluemix.
- Create services of Conversation and Tone Analyzer and save the credentials.
- Train the Dialog, Intents, Entities section in Watson's Conversation.
- Install Watson Developer Cloud on your machine.
- Use the Python API provided by Bluemix, to execute the conversation and suggest activities based on user's mood.