Twitter News Chatbot – Public Event Recommendation System

Master Thesis Proposal

Petar Petrov 11729352

Abstract

The question, that I seek to answer with my thesis is whether social media could be used to aggregate news and inform users about topics they may be interested in. More specifically, I will focus on future event updates and attendance tracking for events, happening in Vienna. The social media of interest is Twitter and the format, that I will provide for the solution will be in the form of a chatbot interface, which will be hosted on Telegram. The main idea is to ask the specific user some questions, build understanding of his/her wishes and interests and present to him the required information about the nearby events. In order to achieve this, the solution will include a recommendation engine, which will be powered by the prior searches made by the specific user, but also from trends in the area. The final output will comprise of several recommendations, which the user may like or dislike, which will help the recommendation engine learn and understand the user better and provide better and better event proposals on each subsequent run.

Introduction

Nowadays the usage of different social networks is at its peak and shows no signs of slowing down. There are so many different kinds of social networks ranging from simple dating applications to complex and diverse systems like Facebook. However they all share one thing in common – the ability to interact between their users using different means of communication. An important aspect of these networks is the real-time information sharing of news and events, by updating statuses. I am going to focus on only one of all the available networks – Twitter. It has so much influence, that there is already a word, which refers to the status update of a specific user in the network – *tweeting.* Here is a definition: a message posted on the Twitter social media service and website: the message may include text, links, photos, or videos. [1]

Tweets have a maximum length of 280-characters (recently updated from 140) which makes them really easy to write and don't require much effort and time to post an update about what is on one's mind. This makes the tweet also a really appropriate format to share news and events happening nearby from non-journalists. The advantage of this, is that the news will spread faster and the people will be better informed about many different topics, not only from mainstream media.

Thesis Statement

The aim of the thesis is to develop a user interface in the format of a chatbot, which is appropriate for the subject matter. It is easy to work with and there is already a lot of research and developments in the sphere. The hypotheses, that will be tested are:

H1:
The chatbot interface is appropriate for the purpose of requesting and presenting data to the user. The feedback from the user is sufficient to update the recommendation model.

H2:
It is possible to use Twitter to be notified about public events happening nearby. The information that the user receives is sufficient and is relevant to him/her.

Related Work

Before I tried to come up with a solution for the problem, I have introduced myself into the topic by reading some research papers regarding the subject matter. Most of the approaches, that are available online are describing event detection, but I am focusing mainly on event extraction [1]. Two of the more interesting papers, that I have encountered were:

- Open-domain extraction of future events from Twitter – Kunneman and Van den Bosch (2016) [2]. The approach they used was mainly focused on the **explicit references to the start time** of the upcoming event. They refer to the word event as follows: 'An event is a significant thing that happens at some specific time and place', where 'significant' is defined as '[something that] may be discussed in the media'. Furthermore, in order to distinguish public from private events, they rely on the assumption, that an event is significant, if many people have tweeted about it (setting a threshold of 5). Initially they start with a wide selection of tweets, which is then shrunk based on the number of people, that have tweeted about the same event and the same time. In order to decrease the duplicate output, they use clustering to combine similar tweets into clusters. The final results of the model were evaluated by showing the top-250 ranked events found to human annotators. Eighty percent of the candidate events were indeed assessed as being an event by at least 3/4 of the human annotators and 63% by all four. The recall was evaluated using maintained calendars on the Web. The top-250 events recall was 0.20 and 0.40 for all retrieved events.

- Social Event Detection on Twitter – Daehnhardt et. al. (2012) [3]. Their approach for the filtering of future events focuses on **a Naïve Bayes classifier**. The 1st task in this approach is to find specific Twitter users, which are mainly concerned with event publishing and broadcasting. Their tweets are assigned as events, when training the classifier and tweets published by other users are initially assigned as the Other class. Then in the training phase, all of the tweets, which are classified as Other, are filtered for the presence of event-related aspects (time, persons involved, locations). If such aspects are present in the tweet, it is classified as an Event. This is done, in order to enable automatic identification of events by using a supervised learning technique, without the need of manually labeling the training datasets.

---

1 – One must note the difference between event extraction and event detection: Event extraction is regarding future events, event detection is primarily focused on past events, which are having effect on social media. [2]

By reading these and also other related papers, I came up with a solution, that may suit the purpose of this thesis best by combining features from different papers and mixing them up. I have decided not to use the approach proposed by Daehnhardt et. al. because of the classification part, which I find too cumbersome and will prefer to use unsupervised learning techniques in order to solve the problem. Therefore, I will adopt some of the methods described by Kunneman and Van den Bosch, namely the time references, the ranking of events and the clustering to remove the duplicate and redundant tweets. Furthermore, I will provide a set of event locations – a gazetteer, as proposed by Zhang et.al. (2018) [3]. Each location will have a specific event type category label, which will be used to filter the tweets. In the next section, I will explain roughly how the system will work.

Approach

The thesis will consist of two conceptual parts – the development of the software and then evaluation of the results obtained from running the software. Here is my proposed solution to this problem:

The first element, that needs to be developed, is the Database server, which will serve to store the user information, the downloaded information from the API and the keywords, which will be used for matching and recommending. The DB of choice is MongoDB, because of the format, in which the tweet data is received – JSON. MongoDB makes it easy to work with JSON files and therefore, I will use it as my DB engine.

The next component is the Recommendation engine, which will be built on Python and will try to find appropriate events, which will be interesting to the user. In order to create recommendation engine, I will need to have a list of locations and label them into specific categories. In order to receive a sufficient amount of locations, I will use:

1. DBpedia – the Wikipedia open database
2. HERE Technologies API – provides mapping and location data

The gathered location information will be labeled into categories and stored in the database for further usage. Next will come the time standardization, which is a rather complex task, because of the different types of time representation tweets may have. For example one tweet may mention the time explicitly (dd.mm.yyyy), other may mention it again explicitly, but in different format (mm/dd/yyyy), or another tweet may possibly be using words (tonight, tomorrow, etc.). Furthermore the task is more difficult, because time expressions should be matched using two languages – German and English. This phase will require usage of different NLP techniques and python time expressions libraries. As a final step, before sending the results to the user, a clustering will be performed in order to remove duplicate tweets related to the same event.

The Recommendation engine will query the Twitter API, based on the values, that were passed by the user, using the Chatbot and were stored in the database, namely the location or the category type. The received tweets will be stored to the database and an analysis will be made by the Recommendation engine.

The last step of the implementation is the Chatbot interface. It is the way the user will interact with the system and will build the recommendation engine. It will be hosted on Telegram in order to make it public and accessible from all around the world. The language of choice for the development is Python and will include various machine learning algorithms. There will be an option to register a new user, or to login with an existing registration.

Here is a sample workflow:

1) Login/New User Registration
2) Greet the user
3) Depending on whether the user has an account:
   a) **The user has an account:** The system has already recommended events for the specific user and outputs them. If the user is not satisfied, he/she will be asked a new set of questions and they will be stored in the database for further usage.
   b) **The user doesn't have an account:** The chatbot asks generic questions – about the current location, areas of interest etc. and stores the information in the database.
4) The Chatbot starts the Recommendation Engine
5) The Recommendation Engine retrieves the relevant information from the DB (the location and event types, in which the specific user is interested in)
6) The Twitter API is fed with the relevant search keywords and finds tweets matching the pattern. The bodies of the tweets are stored in the database.
7) The Recommendation Engine filters the tweets, looking for duplicates and irrelevant ones.
8) The Recommendation Engine prepares a list of relevant events and sends them to the Chatbot
9) The Chatbot outputs the information to the user.

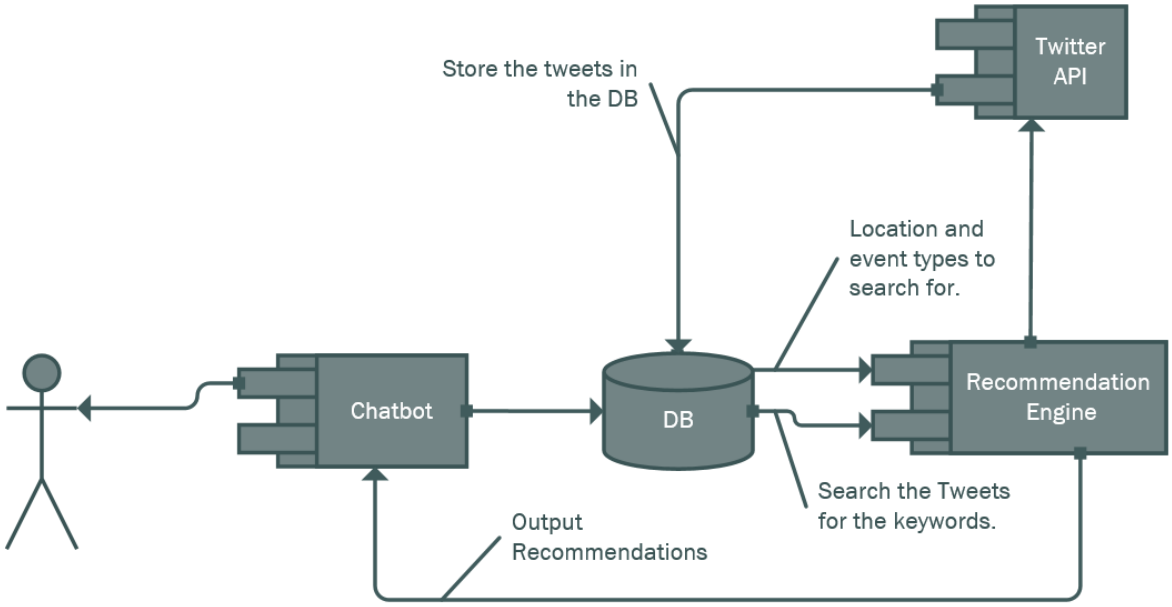

Fig.1 – Workflow and interaction between components

After the entire system is up and running, I will focus more on the improvement of the recommendation engine by introducing different machine learning algorithms and trying to improve the performance. The performance of the model will be measured by calculating the recall and the accuracy of the output. Another thing that will definitely need some tweaking is the chatbot interface, which can always be trained better with different data and various Natural Language Processing techniques. In the next section, I provide details on the evaluation of the results.

Evaluation of Results

In order to accept or reject the hypotheses, that have been defined earlier, I will have to introduce some metrics and to compare the results of my model with them:

- H1 Evaluation – The chatbot functionality has to be evaluated by a non-biased person, who will try to assess the output, based on the following metrics:
    - Robustness to unexpected input - percentage of successes, the robot will continue executing properly after being given unexpected input. (acceptable is over 75%)
    - Correctness of information transferred to the Recommendation engine – does the entered data by the user correspond to the queries, that are executed by the Recommendation engine component? – acceptable over 90%
    - Provides greetings, pleasant personality – True/False (True acceptable)
    - Guides the user through the entire process of Recommendation – The chatbot must ask the required questions and keep the conversation alive until being said to stop. – acceptable over 80%
- H2 Evaluation – The Recommendation engine will be evaluated by several non-biased users, who will mark the recommendations they receive with Interested or Not Interested.
    - The acceptable percentage threshold for True Positive responses will be 60%. It is usually hard to develop an understanding about the wishes of a user, without having much information initially, therefore it would take some time for the system to output more precise recommendations.
- Furthermore, each entry in the database, which is labeled as an event, can be evaluated using the precision and recall metrics. I will test for them with an approach adopted by Kunneman and Van den Bosch (2016):
    - In order to calculate the precision, a sample of arbitrary events will be extracted and the evaluators will go through it, while marking each one as a True or False, depending on whether they consider it an event or not. The acceptable precision rate threshold will be set at 80%.
    - In order to calculate the recall, for arbitrary number of event locations, an official event calendar will be downloaded from the Web (https://www.events.at/) and the output from the recommendation engine will be compared to the full list of events. The acceptable recall rate, I will set at 20%, because I would prefer to achieve better accuracy and have proper public events, than having large number of non-public events existing in the dataset and recommending them to the users.

Work Plan

Based on the number of components, that are part of my solution, I will have to focus on different parts during different time. The first step is to setup the database, then setup the Twitter API in order to be able to receive the information, followed by the extraction of

locations. The Recommendation Engine, along with the API tweaking will be the next stage and I assume, it will be the longest one. There I will have to tryout different algorithms and approaches in order to achieve maximum accuracy and prove my hypotheses. Then a simple chatbot interface will be implemented in order to have the entire system working. Finally, I will focus more on debugging and improvement of the Chatbot interface.

1) Database – End of November
2) Recommendation Engine – End of January
3) Chatbot for Telegram – End of February
4) Debugging and improving – End of March
5) Wrapping up the paper – 15th April

Implications of Research

The main goal of this project is to output user-specific recommendations about various event types, which are occurring nearby. This will be achieved in an interactive and user-friendly way with a natural language conversation between the user and the chatbot. Furthermore, not only the user of the solution will be satisfied from it by being informed about more events, but also the organizers of these events will benefit by having their events shared with a bigger audience.

References

[1] https://www.dictionary.com/browse/tweeting

[2] Open-domain extraction of future events from Twitter – Kunneman and Van den Bosch (2016)

[3] Social Event Detection on Twitter - Elena Daehnhardt, Claudia Hauff, Ilknur Celik, Fabian Abel, Geert-Jan Houben (2012)

[4] Yihong Zhang, Claudia Szabo, Quan Z. Sheng, Xiu Susie Fang: SNAF: Observation filtering and location inference for event monitoring on twitter. World Wide Web 21(2): 311-343 (2018)

[5] Jeffrey Dalton, Victor Ajayi, Richard Main: Vote Goat: Conversational Movie Recommendation. SIGIR 2018: 1285-1288