Department of Computer Science, University of Leicester CO7201 Individual Project

JokeBot: An AI Comedian Twitter Bot

Aakash Chahal
ac879@student.le.ac.uk

Preliminary Report

Project Supervisor: Dr Paula Severi
Second Marker: Muhammad Kazim

Word Count: 1345 words
Date of Submission: 03/03/2023

Declaration

All sentences or passages quoted in this report, or computer code of any form whatsoever used and/or submitted at any stages, which are taken from other people's work have been specifically acknowledged by clear citation of the source, specifying author, work, date, and page(s). Any part of my own written work, or software coding, which is substantially based upon other people's work, is duly accompanied by clear citation of the source, specifying author, work, date, and page(s). I understand that failure to do this amount to plagiarism and will be considered grounds for failure in this module and the degree examination as a whole.

Index

1. Motivation and Objective

The motivation behind this project is to build a Twitter Bot that can instantly produce humorous tweets. The bot will create its own tweets using natural language processing after being trained on a library of jokes and humorous content. The objective is to create a bot that can generate amusing and engaging tweets that are creative, witty, and original.

A simple way of creating a bot account would be to simply gather a large amount of comedic dataset (jokes) and reading them into an appropriate data structure (for example dictionary in Python) and we could simply get a random joke out of the dataset and tweet it using the twitter API but the jokes tweeted wouldn't be unique and may potentially tweet same joke more than once, but with the recent rise in AI and Machine Learning we can use and train an AI language model with the help of a large comedic or jokes dataset which would help us generating new and unique jokes in real time and it would also remove the problem of getting the same joke more than once. This gives a new motivation for the project to be an AI bot that can generate and tweet jokes in real time.

Creating an AI-based bot that can create amusing material for social media platforms is the major goal of this project. The following objectives of the project are:

- A dataset of hilarious content should be gathered and prepared.
- Train a language model, like GPT-3, using the dataset to produce original humour.
- Test the fine-tuned/trained model with appropriate techniques, and fine-tune the model based on the results.
- Integrate the learning model with twitter API to enable the bot to tweet on its own.

2. Requirements and Challenges

Essential:

- The bot must be able to generate original jokes in real time and tweet them.
- The bot must be able to understand and respond to user interactions which includes likes, comments, retweets and mentions on twitter.

Recommended:

- The bot should be able to go through trending topics and hashtags and should be able to generate jokes related to those topics.
- The bot should also be able to handle normal conversation in comments instead of just replying with a joke.

Optional:

- The bot could be trained on additional dataset to improve the comedic abilities. Additional datasets being datasets of jokes from real conversations which can be obtained using various transcripts/data available from the movies or tv shows.
- Additional model can be trained to interpret normal conversation.

Challenges:

- Compiling a vast and varied dataset of jokes and humorous material.
- Improving the language model to produce unique, and original jokes.
- Seamless integration of the language model and Twitter API.
- Maintaining and tracking the bot's engagement and performance.

3. Technical Specifications

The AI Comedian Twitter Bot will be utilizing natural language processing technique and machine learning algorithms to generate humorous jokes and tweet them. Specifically, the project would involve:

- Pre-processing the dataset of comedic content to prepare it for training the language model.
- Training the language model (such as GPT-3) using Python.
- Develop / Integrate the model with Twitter API.
- Optionally launch or host the bot on a cloud server like AWS or firebase.

Following are some of the specific details about the project:

Programming Language: Python
Libraries / APIs: OpenAI API, Twitter API
IDE: Visual Studio Code
Version Control: SVN (Subversion)

4. Requirements Evaluation

The following metrics can be used to evaluate and gauge the success of the project:

- The coherence and humour of the tweets generated by the bot, as measured by user feedback and engagement metrics such as likes and retweets.
- The bot would be tested with appropriate Machine Learning algorithms to classify if the jokes are funny or not.

More evaluation plans would be incorporated depending upon further research that needs to be done which includes Twitter API and what sort of user data we can gather using Twitter API.

5. Background Research and Reading List

Since the project includes a very vast topics if machine learning and AI, it is really important to do enough reading and background research on relevant materials and similar projects that has been successfully developed in the past, following are some of the documentations, books and sources that are and shall be used for the background research for this project:

- "*Natural Language Processing with Python*" by Steven Bird, Ewan Klein, and Edward Loper, Available online at: https://www.nltk.org/book/
- What is a Twitter Bot?, Available at: https://www.techtarget.com/whatis/definition/Twitterbot
- OpenAI API Documentation, Available at: https://platform.openai.com/docs/introduction
- Twitter API Documentation, Available at: https://developer.twitter.com/en/docs
- *"Conversational AI: Dialogue Systems, Conversational Agents, and Chatbots"* by Michael Mctear.

NOTE: The above reading list is as per the current research and knowledge of the project there might be some additional resources that can be added to the list accordingly.

6. Work and Risk Plan

Detailed Work Plan

Following tasks and sub-tasks will be undertaken in order to complete the project:

- Gather and pre-process dataset (week 1-2)
    - Collect various and diverse datasets of comedic contents preferably jokes.
    - Pre-process the collected data to remove any irrelevant content and transform into a suitable format for the learning model.
- Train the language Model (week 3-4)
    - Selection of a suitable language model architecture, such as GPT-3 and fine-tune or train it on the processed dataset of jokes or comedic content collected at the beginning.
    - Evaluate the performance of the model and make any changes if necessary.
- Generate Jokes and Test the Model (week 5-6)
    - After training the model, it's time to generate new jokes using it.
    - Test the generated content using appropriate algorithms to see if the generated joke or content is funny or not.
    - User testing can also be done at this stage where generated jokes could be shown to some users and evaluate of they find it funny or not.
- Integration with Twitter API (week 7-8)
    - Following the testing of the learning model, we'll integrate the learning model to a twitter account using Twitter API.
    - Develop appropriate responses for user interactions like retweets, and comments.

- Evaluation and Optional Features (week 9-10)
    - After launching the bot on twitter, evaluate the bot's performance using user feedback which can be obtained using likes, retweets, comments or even followers.
    - Update or make any changes if necessary according to the feedback.
    - Work on optional requirements or any other requirements that are left behind or might have been dropped for some reason.

Risk Plan

The project might potentially have the following risks and their respective mitigations:

- Difficulty obtaining a quality dataset: Mitigate the difficulty by expanding the source of datasets, collecting a large amount of data, and carefully pre-processing the dataset.
- Overfitting the language model: Regularization techniques and performance evaluation on a validation set can be used to reduce the risk of overfitting the language model.
- Technical difficulties with the integration with Twitter: Mitigate by thoroughly testing and debugging the integration with twitter API.
- Insufficient computing resources for training on additional datasets: Mitigate by minimising the size of additional datasets or by using cloud computing resources.