Manuel Salado Alvarado

Kyle Mullings

6320.001

5/5/2025

Comet News Report

**YOUTUBE AND GITHUB**

https://www.youtube.com/watch?v=a-7R2ZnCE_s

mxs170018/cometNews

**Base Model**

Base model had a zookeeper/Kafka combination server that hosted two topics. Those topics were used for inter-process communication by reading and writing to a consumer and producer, respectively. The base model also had a newsAPI set up so that it would trawl recent news articles for important subjects. It did so in a rule based manner, and it was pretty bad because it allowed multiples of the same entity, it would have dates and numbers and sometimes even links. It would then visualize the data as a bar graph (figure 1). In fig. 1 you can se that "Monday" is on there, when it shouldn't be. "Second", which is another non-sensical entry. "Lawrence Summers Trump" refers to the journalist Lawrence Summers that did an article on Trump. This was for a class on databases, so the NLP aspect was not a priority, having the servers communicate was. As 'future works' for that homework, I mentioned that some better NLP would do this homework justice. Kyle and I decided that we would use this homework as the base model of the project to implement NLP techniques we have learned in class and hook it up to a chatbot.

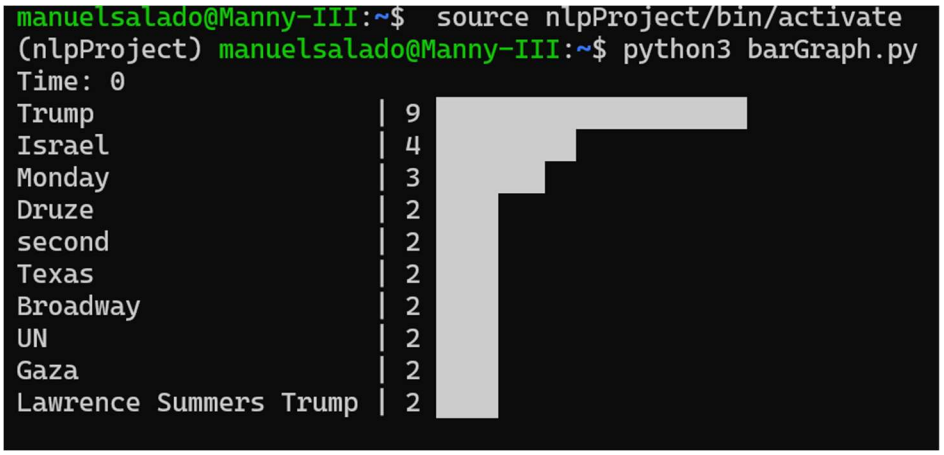


*Figure 1: Base Model*

## Contributions

**MANUEL** did the following: We decided to ditch the rule based approach and instead use REGEX to get rid of any numbers and links. We then fed this semi clean text into SPACY's en_core_web_sm model to tokenize, remove stop-words, and do a basic named entity recognition pass. This was fine, but it left a lot to be desired, so we instead used the en_core_web_trf to make use of a larger model that uses TRANSFORMERS to get the benefits of ATTENTION. We then also added a few checks to exclude words that had LABELS that were marked as cardinal, ordinal, quantity, date, time, money. We decided to still add a RULE that explicitly included some words that were still getting through the filter. After we had this cleaner text, we used the spacy model to get all the NAMED ENTITIES that were not in the excluded labels or words. We iterated through all the ents and made a ENTITY UNIGRAM COUNT DICTIONARY. This is done every three minutes with the articles that are coming in through the newGrabber.py which sends the most popular news as raw text every three minutes. The clean entity count dict is sent to cometNews.py. In that file we take the top 10 entities with the most counts, and randomly picked five entities (not from the top 10). This was done to avoid always having a bias towards American politics (President Donald Trump has always been the most talked about entity) and include some variation of news that may be important, but isn't being talked about everywhere. **KYLE** did the following: These are displayed as before, but on a GUI and now we also offer buttons that correspond to these entities which feature a live lookup. It gets the news from which they were grabbed, it looks them up on Wikipedia, it CLEANS the text and then sends it to a LARGE LANGUAGE MODEL (openai API) to provide a summary (RAG) based on contextual past (Wikipedia) and current events (the news). It CACHES responses for the same entity for five minutes, so that the user doesn't have to wait when cycling between topics. We do assume that after 5 minutes, a new article could've broken, so the live lookup is performed again. *We both worked on everything together, but this is how we split the workload.*

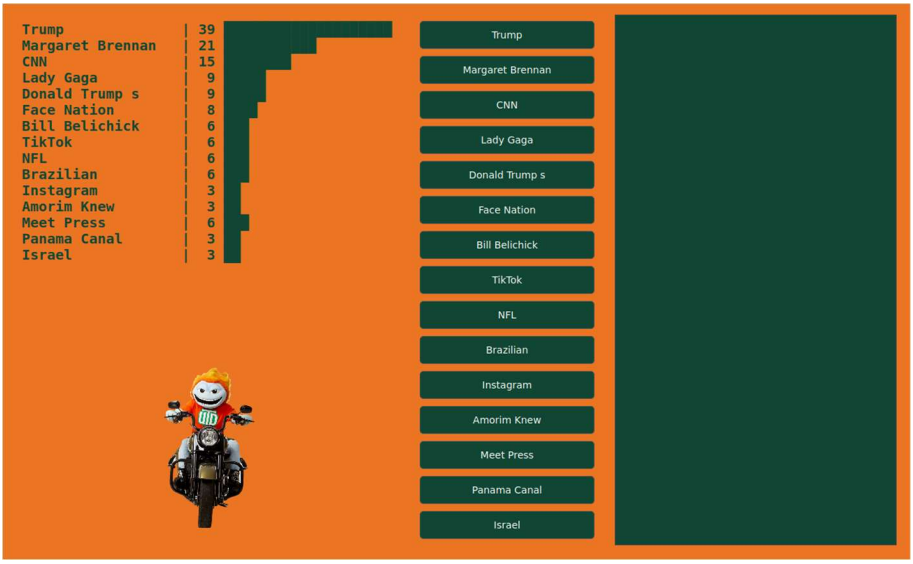


***Image 2: CometNews GUI***

**Lessons Learned**

We initially wanted to have everything done locally with OPEN SOURCE SOFTWARE via OLLAMA and DEEKSEEK-R1, but we realized that that would require 500GB of space for some of the more competent models, and the small models were just too slow and not good enough. Especially since we did all our developing and testing on a virtual machine where the GPU was not available. On the virtual machine, we said hello to the smalled deepseek r1 model and it took 2 minutes to respond with "Hello, how can I assist you?" while making the VM chug. If we had access to more powerful machine with a dedicated GPU, we might have been able to implement this locally (except for the news and Wikipedia, obviously). Spacy with transformers works so much better than the normal small English core which I have used for all of my homework and projects so far. It is a little larger, but it's a total of 2GB which isn't too bad. Also getting rid of the labels to avoid bogging down the entities helped a lot. Making a GUI is a lot harder than it seemed, and getting it to plug in to everything was tough. If we really wanted to improve this project we would store all user generated requests so that they could see their logs. Another note is that we could've used some transformers to better host RAG on the initial news we grab, instead of waiting for a lookup when a user request it.

**Team Member self-scoring**

**Manuel Salado Alvarado:**

70 points – significant exploration beyond baseline:

Accomplished much more than described in base model

20 points – innovation or creativity:

Used REGEX, SPACY, TRANSFORMERS, NAMED ENTITY RECOGNITION, UNIGRAM COUNTS amongst other things to trawl through news.

10 points – highlighted complexity:

Improved on NLP from base model using the aforementioned techniques in the contributions.

10 points – discussion of lessons learned and potential improvements:

We both sat together and discussed this for an hour and summarized it above.

8 points – visualization:

The project looks a little poor, but it gets some extra points for the school spirit charm.

10 points – discussion of testing outside of the team:

discussed with roommates, parents/family, girlfriend, and a couple members from previous projects.

-1 points – money made:

spent money ($20) on openAI API.

**Kyle Mullings:**

70 points – significant exploration beyond baseline:

Accomplished much more than described in base model

20 points – innovation or creativity:
used HUMAN LANGUAGE TECHNOLOGIES to create a GUI. Hooked up a LARGE LANGUAGE MODEL to do RAG given CURRENT CONTEXT AND PREVIOUS CONTEXT.

10 points – highlighted complexity:
Hooked up many different API's for live lookup using LLM along with caching techniques amongst other things.

10 points – discussion of lessons learned and potential improvements:

We both sat together and discussed this for an hour and summarized it above.

8 points – visualization:

The project looks a little poor, but it gets some extra points for the school spirit charm.

10 points – discussion of testing outside of the team:

Discussed with parents/family, friends from other fields (nurse, doctor, and accountant), and a couple members from previous projects.

0 points-made money:

Used QT which is free for the GUI.