IST 707- Applied Machine Learning

Unsupervised Sentiment Analysis of Tweets on GPT-4

Abirami Rajalingam
(arajalin@syr.edu)
Danila Rozhevskii
(drozhevs@syr.edu)
Venkata Sai Mani Lakshmi Kavya Darsi
(vdarsi@syr.edu)
Lahari Chowtoori
(lchowtoo@syr.edu)

IST 707:Unsupervised Sentiment Analysis of Tweets on GPT-4

ABSTRACT

The GPT-4 is a potential successor for the GPT-3, a cutting-edge language model for tasks involving natural language processing. We will be compiling a dataset of tweets from Kaggle in order to assess the performance of unsupervised sentiment analysis on GPT-4. We then preprocess the tweets to subject the unsupervised sentiment analysis techniques.Each tweet is labeled with a sentiment score ranging from -1 (negative) to 1 (positive) based on the text's polarity**.** Our K-means model achieved reasonable results on an unsupervised sentiment analysis task of tweets about GPT-4 based on the benchmarks from existing pre-trained models such as BERT and VADER.

INTRODUCTION

In this machine learning project, one of the most well-liked and easily accessible sources of social media data is the sentiment analysis of tweets, and our goal is to investigate K-means potential in this area. We will look at the intricate and nuanced language of tweets in order to better comprehend and address user opinion on GPT-4.
We created a k-means sentiment analysis model that categorizes tweets as positive, negative, or neutral using a combination of natural language processing methods.We used pre-trained models to generate true labels and used Cohen's Kappa to compare the model's performance to those benchmarks.

Overall, this project offers a fascinating chance to investigate the k-means potential to perform accurate unsupervised sentiment analysis.

RESEARCH QUESTIONS:

- Based on the tweets, how does the GPT-4 impact society? Is the public opinion mostly positive, neutral or negative?
- How well can the K-means method perform unsupervised sentiment analysis on tweets?

EXPLORATORY DATA ANALYSIS:

DATA DESCRIPTION

The data used for our project was collected from Kaggle. The dataset is a collection of tweets on GPT-4,which is an upcoming language model from OpenAI.The dataset contains around 28000 tweets,which were collected using the Twitter API.The tweets were collected from 14th March 2023 to 12th April 2023 and cover various topics,such as politics,sports,entertainment and technology. The dataset contains over 28,710 rows and 12 columns.Each tweet in the dataset is associated with metadata such as the username, user location and user description.

DATA CLEANING & FILTERING

1. **Remove spam and unrelated tweets**
   Due to the abundance of repeated spam and unrelated tweets observed during data exploration, those tweets should be removed to not spoil the final word vector dictionary. They generally include tweets from spam bots, tweets

2. **Remove rows with duplicate values in the tweet's column**
   To make a final filter, duplicate rows on the "text" column are removed. They might be the result of both spam and retweets.

After cleaning the final number of tweets used further is 20,465.

DATA EXPLORATION

Exploring the data is a crucial step, and in the case of Twitter datasets based on GPT-4 tweets, there are several ways in which data analysts can use visualizations to identify patterns and trends.
We first analyzed the trend based how many number of tweets per hour were tweeted based on the #gpt4 tweets.

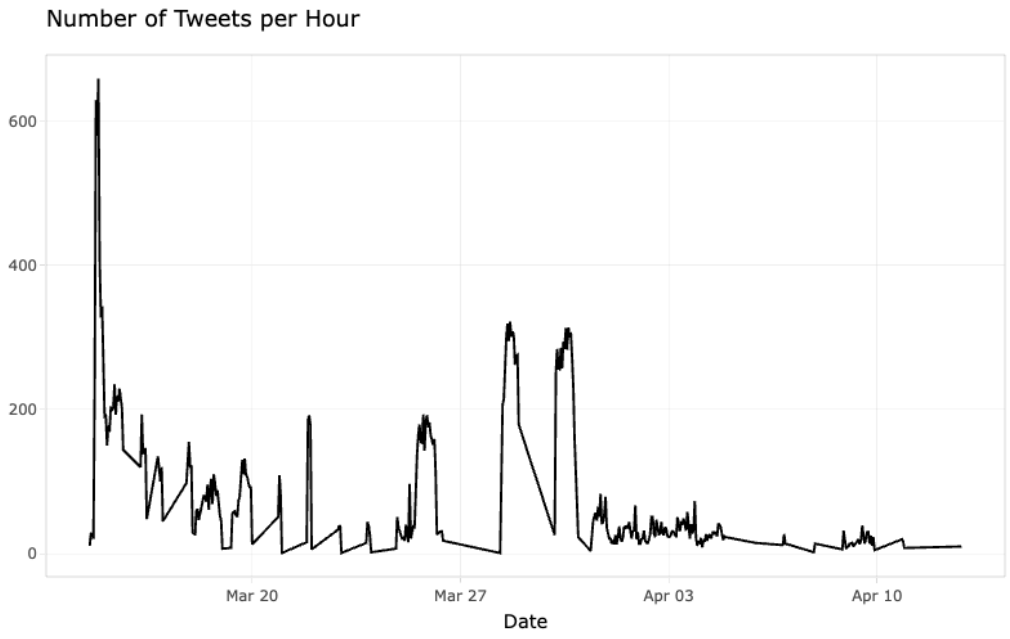

Figure 1 - Trend Plot for Number of tweets per hour

Based on the above trend plot we inferred that on the day of GPT-4 release, there were more tweets tweeted with hashtag GPT-4.

To find the most common hashtag mentioned in the Twitter dataset we visualized using the word cloud plot. From the plot we inferred that GPT4 was one of the most frequently used hashtags.

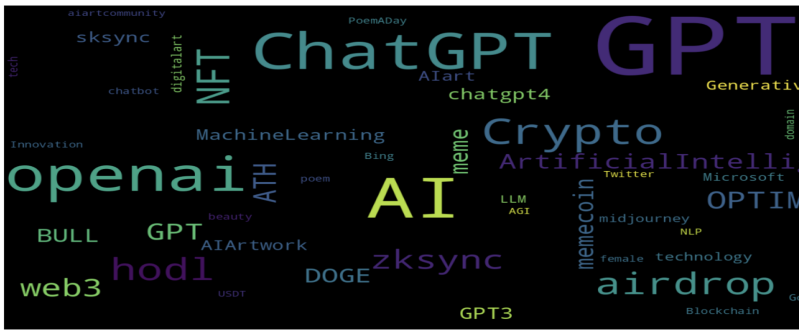

Figure 2 - Word Cloud for Hashtag

DATA PRE-PROCESSING

In the data preprocessing step, there are main 2 steps: filtering and tokenization.

Filtering:
For filtering, the function was used on the raw Tweet column that turns text into lowercase and removes mentions, hashtags, links, punctuations, non-alphabetical characters, and extra spaces, The same function also performs the lemmatization of the words, which leaves only the core of it in order to create a proper dictionary from the corpus. The last step of the function is to create a list of tokens from the remaining words to later feed into a neural network and create embeddings.

Tokenization:
After filtering, the original Tweet column is split into tokens_tweet and cleaned_tweet. The first column is the result of the filtering function applied to the raw text, and the second column is the resulting list of tokens combined into a single string. The latter column is used to get the sentiment from the pre-trained models later that only accept complete sets of sentences and not tokens.
The last tokenization step is to use the Gensim package's Phrase function to create a list of sentences, which include both single-word tokens as well as bigrams. Since some words often come together in combinations, such as names and companies, we need to consider both unigrams and bigrams in our tokenization process.

MODELING

The modeling step consists of two parts: embedding generation, the K-Means model, and pre-trained models.

Embeddings:
After turning tweet data into separate lists of unigram and bigram tokens, the Word2vec model is used to vectorize each tweet. Most of the neutral network settings are left at default, except the vector size which is set to 300 and is optimal when vectorizing tweets. The vectorization takes two steps: creating a corpus or vocabulary from the input text data and then training the mode on that data in order to learn each word's representations. For checking purposes, there is a function that allows users to search for similar words in semantic meaning using newly created embeddings, and this is how they are tested for relative correctness.

K-Means:
The corpus of word vectors is then fed into the K-Means algorithm configured for 3 clusters: positive, negative, and neutral. However, it is worth mentioning that words are divided semantically, which means if the word "good" is often used in sentences with negative context, it is considered a sign of a negative cluster by K-Means. The neutral cluster's goal is to collect all the junk and irrelevant words which say nothing about either sentiment or semantic meaning.

After training the model, the clusters are indicated and labeled by hand, manually looking through the first top 200 words closest to each cluster center. This can give the best idea if the cluster is positive, negative, or neutral. Additionally, the wrongly placed words in clusters are manually reassigned to appropriate clusters to increase the accuracy of the model.

Finally, get_sentiments() function is mapped on the Tweets tokens column, which uses the K-Means model to identify the sentiment of every word and then averages the combined score of all tweets into a single value that is an approximation of the tweet's sentiment.

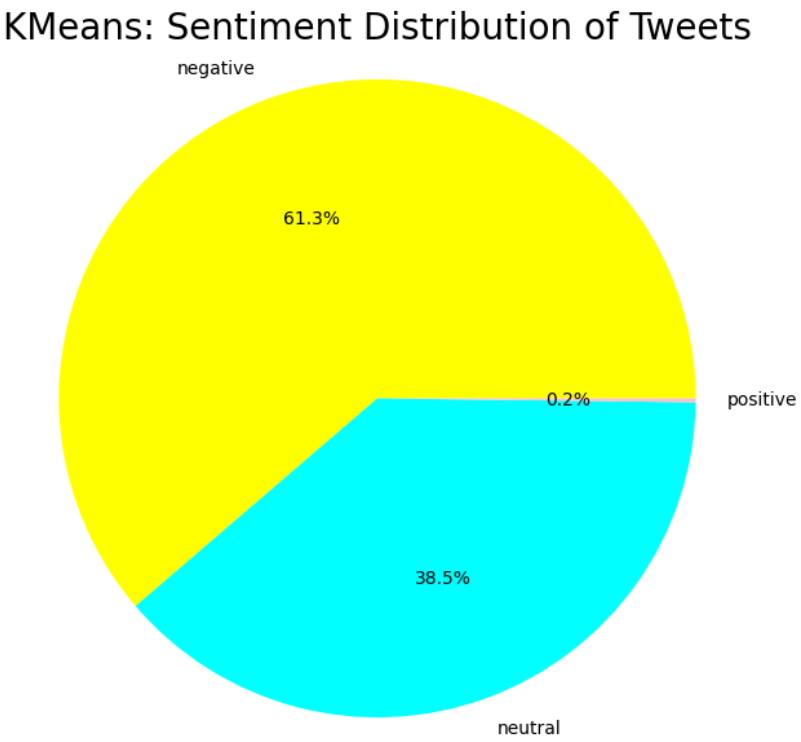

Figure 3. The sentiment distribution of words.

From Figure 3, it is clear that a majority of words were clustered in the negative and neutral categories by K-Means, which means there is a big saturation of tweets on the particular two topics. The negative cluster looks very small in comparison to the other two, which means it could actually be the positive cluster and not the negative one.

Pre-trained models:

Since the project is about unsupervised sentiment analysis, there are no true labels for the dataset. That is why pre-trained models such as BERT (Large Language Model) and VADER (keyword-based model) are used to provide close-to-the-true labels.

BERT is Bidirectional Encoder Representations from Transformers, a family of masked-language models introduced in 2018 by researchers at Google. This pre-trained is able to accept raw text (cleaned_tweet column) as input, perform vectorization on its own and conclude the sentiment based on the training data it has. However, it only has positive and negative sentiment output.

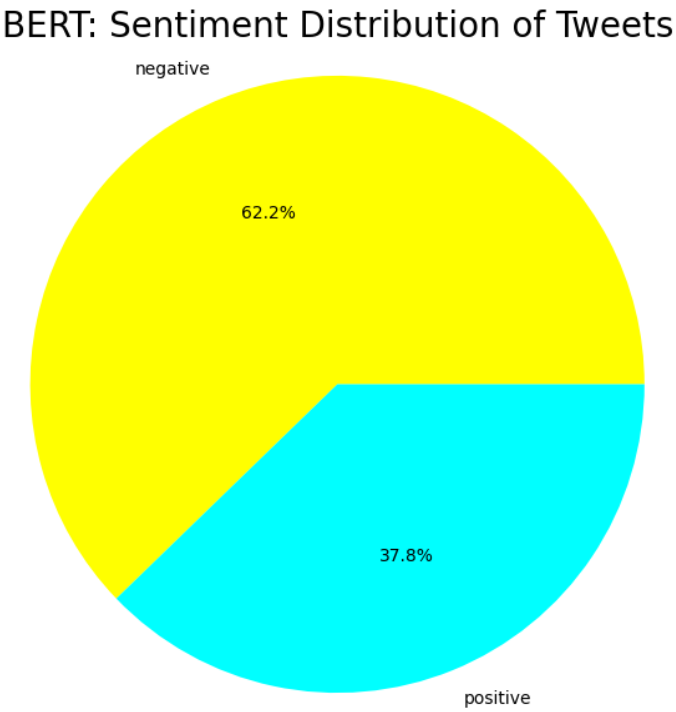

Figure 4. Sentiment distribution of tweets by BERT.

From Figure 4, there is an even distribution of sentiment with negative tweets dominating the ratio. Interestingly enough, the ratio is very close in exact percentages to the results from Figure 3, K-Means result. In the discussion section, there is an explanation of why it is probably not the case of similar accuracy.

VADER:

Valence Aware Dictionary for Sentiment Reasoning is an NLTK module that provides sentiment scores based on the words used. It is a rule-based sentiment analyzer in which the terms are generally labeled as per their semantic orientation as either positive, negative, or neutral. The rule-based means that this method has a pre-determined semantic dictionary that is able to evaluate the average sentiment of a tweet based on the average score across all words.(Google) For that reason, unlike BERT or K-Means, this algorithm can not account for the context of the tweets, only the rough estimate baked off keywords found.

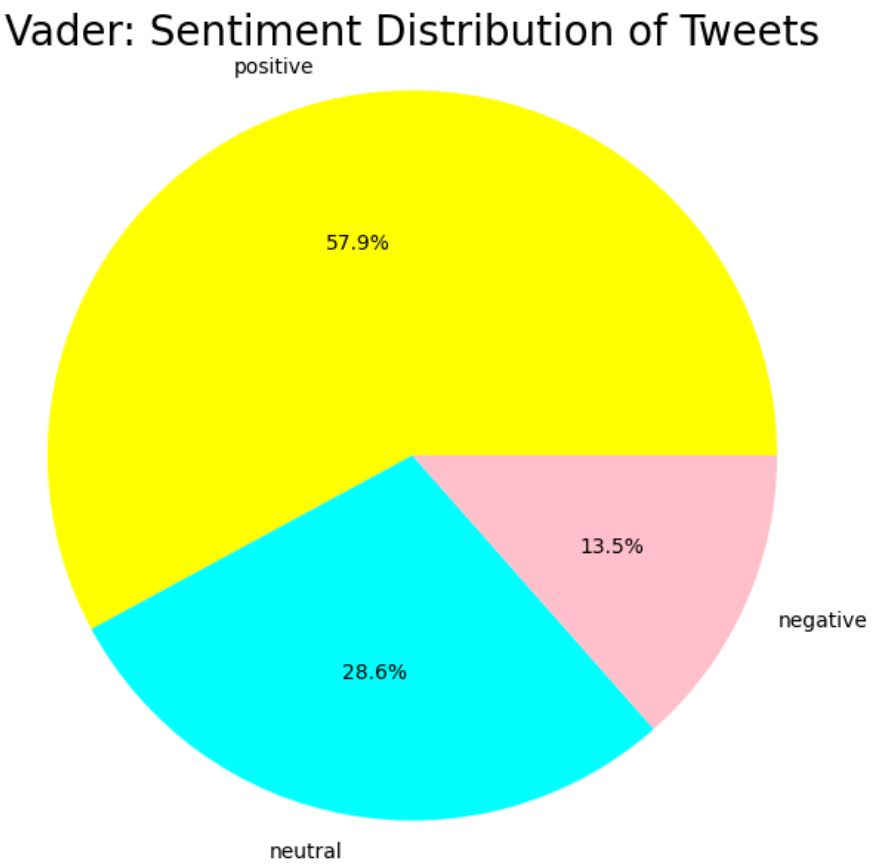

Figure 5. Sentiment distribution of tweets by BERT.

From Figure 5, the pie chart looks the opposite from the first 2 pie charts. It looks like, according to VADER, negative tweets accounted for only 13.5% in comparison to around 60% estimated by both K-Means and BERT. The positive tweets still lack a lot in numbers, which makes the result seem out of context. Since the algorithm is key-based, it is possible that text on a narrow topic can be easily misunderstood by the algorithm. More analysis is given in the discussion section.

EVALUATION

The evaluation part consists of 2 stages: collecting true labels and comparing them to the results from K-Means, as well as other models using Cohen's Kappa Score.

True labels:
Since there are no true labels, 30 sample tweets from the data were chosen as a test set. The members of the project team each gave their own opinion on the sentiment of each tweet.

Cohen's Kappa Score:

Cohen's kappa coefficient is a statistic that is used to measure inter-rater reliability for qualitative items. It is generally thought to be a more robust measure than simple percent agreement calculation, as κ takes into account the possibility of the agreement occurring by chance. (Wiki)

Cohen suggested the Kappa result be interpreted as follows: values ≤ 0 as indicating no agreement and 0.01–0.20 as none to slight, 0.21–0.40 as fair, 0.41– 0.60 as moderate, 0.61–0.80 as substantial, and 0.81–1.00 as almost perfect agreement.

In pairs of raters, K-Means is compared to the pre-trained models' outputs as well as each of the team member's outputs, and the team's average sentiment. Also, as a point of interest, the most accurate model for sentiment analysis, BERT, is evaluated against the true labels of team members, and the team's average.

Results

K-Means vs	**Cohen's Kappa Score**
BERT	-0.153846
VADER	0.037736
Abirami	0.050000
Lahari	-0.045455
Venkata	0.050000
Danila	0.062500
Team Average	**0.000000**

Table 1. Cohen's Kappa scores of K-Means model vs pre-trained models and human benchmarks.

BERT vs	**Cohen's Kappa Score**
Abirami	0.000000
Lahari	0.017857

Venkata	0.000000
Danila	0.078341
Team Average	**0.061033**

Table 2. Cohen's Kappa scores of BERT model vs pre-trained models and human benchmarks.

Discussion

From the table results, it is obvious that both K-Means and BERT models are struggling to have a correlation or some kind of similarity to real human sentiment labels. That might be happening due to the server factors that are discussed below. First of all, the tweets dataset was collected considering only #GPT-4 hashtag, which increases the scope of topics a single tweet could be about. Secondly, the actual tweet sentiments might be very positively or negatively saturated, which means there could be an equal distribution of opinions due to the fact that it is a hot topic. Finally, even with cleaning and filtering steps, the resulting dataset seemed to have repetitive tweets and spam content which made it very hard to evaluate the sentiment of some tweets.One of the best ways to train a better model and achieve clearer results is to wrangle the existing data more and try to make it more clean and representable of actual content on GPT-4. Another idea is to scrape a new dataset more carefully and test the approach again.
The answer to the first research question stays ambiguous as the algorithm needs to be further adjusted and enhanced to make sensible conclusions about public opinion. Based on BERT results, the closest to accurate benchmarks, around 62% of tweets were positive.

Conclusion

The project served us a great experience with utilizing machine learning techniques learned in class. We were able to set a research goal, did data exploratory analysis, and build a machine learning pipeline that achieved desired results. Even though the accuracy of the final model is not the best and can be speculated about, we were able to identify the most likable causes and provide solutions for those in the future.

References

1. Agashini V Kumar, Meera K. N. (2022) Sentiment Analysis Using K Means Clustering on Microblogging Data Focused on Only the Important Sentiments.ICETET-SIP-22
2. Pierre Megret (2018). Gensim Word2Vec Tutorial. From: https://www.kaggle.com/code/pierremegret/gensim-word2vec-tutorial/notebook

3. Rafał Wójcik (2019). Unsupervised Sentiment Analysis. From: https://towardsdatascience.com/unsupervised-sentiment-analysis-a38bf1906483
4. Kurtis Pykes (2020). Understanding Cohen's Kappa coefficient. From: https://towardsdatascience.com/cohens-kappa-9786ceceab58
5. Mary L. McHugh (2012). Interrater reliability: the kappa statistic. From:https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3900052/#:~:text=Cohen%20suggested%20the%20Kappa%20result,1.00%20as%20almost%20perfect%20agreement.