

# Analyzing Current Media Sentiment Towards A.I.

Harvard University Extension School

"Introduction To Functional and Stream Programming for Big Data Systems"

CSCI E-88A, Spring 2023

Daniel Lebedinsky

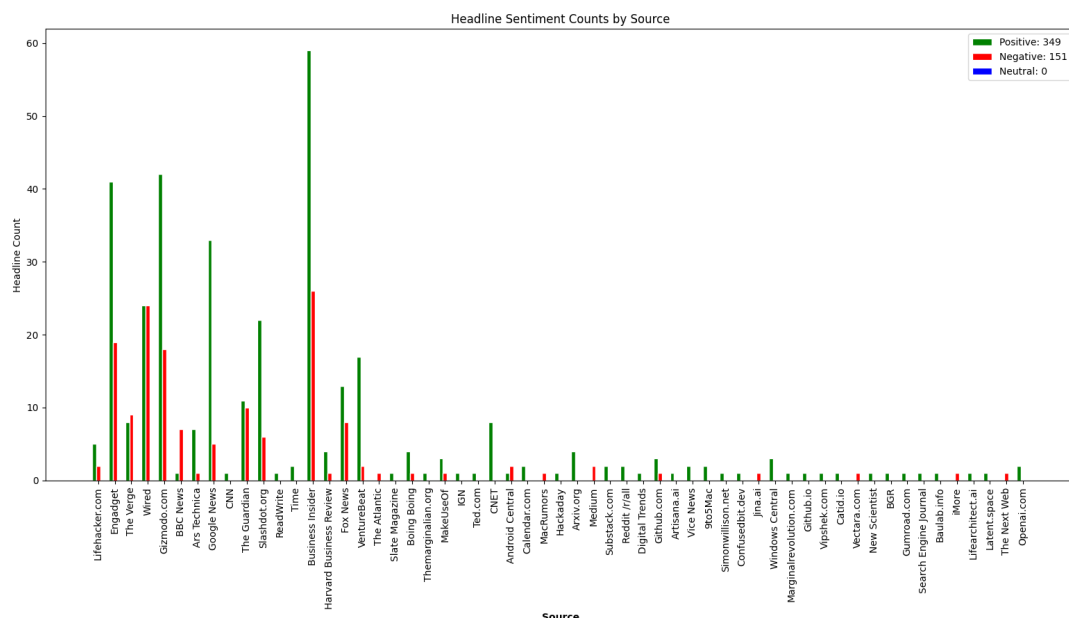
Link to video: [https://youtu.be/Bffazy\\_LHIY](https://youtu.be/Bffazy_LHIY)

The goal of this project was to collect news headlines over the past month, to perform an exploratory analysis of how the subject of Artificial Intelligence is currently being covered between different English-language news sources. The data on this subject were obtained with the free tier of the [Python News API client](#). Multiple queries related to AI were used to download news articles to a static file.

The following technologies were crucial to the analysis:

- I chose the Apache Beam Pipeline as the core technology for the analysis, for its scalability. When Beam reads the data from the static file, it creates a distributed data structure (PCollection) to store the results, which can be contributed to or processed by multiple workers.
- OpenAI's completion API was used in the function classifying Headline Sentiment. The text of the article headline is categorized as either positive, negative, or neutral. Instructions to categorize it as "positive, neutral, or negative" were passed to Davinci, a low-cost variant of ChatGPT that is optimized for rapid queries.
- A Python virtual environment running on Google Cloud Platform (GCP) was the runtime for the Data\_Collection.py and Sentiment\_Analysis.py scripts. Google's service allows the Beam pipeline to run in parallel across multiple servers and/or threads.
- The output text file of the pipeline was downloaded to a local machine, for the sake of user-friendliness, and used for data visualization with Matplotlib. If one prefers using Jupyter notebooks, either locally or on Google Colab, it is very straightforward to convert the visualize.py script to this format.

In the end, I produced the following bar chart:

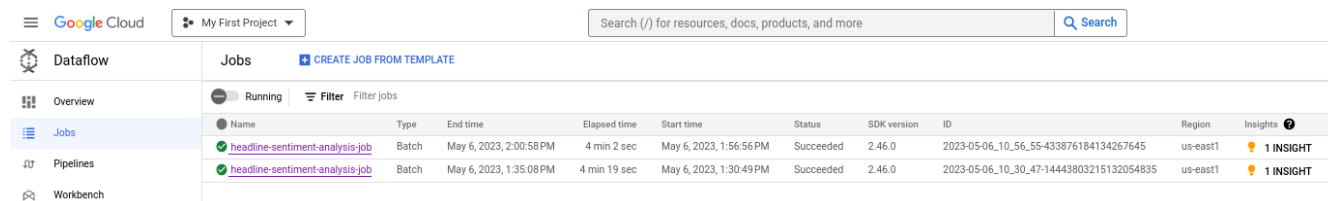


To set up the environment for this project, I activated a GCP account, which allowed me to receive \$300 worth of credits, for a free trial. I opened a GCP shell, uploaded the Data\_Collection and Sentiment\_Analysis scripts, and created a python virtual environment. Then I installed the GCP version of apache beam, and the openai and newsapi libraries. To check the performance of the code, and to verify the number of workers, I enabled GCP's Dataflow API, which created a dashboard to view relevant information and allowed the code to run on multiple workers in the Eastern US.

The purpose of the Sentiment\_Analysis pipeline is to take a file of news articles as input, and return a flat file with the following format: "Outlet Name | sentiment | 24-hour period | count". The input is a JSON file containing a list of dictionaries, where each dictionary represents an article, and the relevant keys are for the headline, the outline, and the time of publication. For each article, the pipeline first parses the above information, then applies fixed windowing of size 24 hours with no offset, to group the current headline with all others published in the same 24-hour period. Then, it applies the Sentiment function, to obtain a value of "positive, negative, or neutral". It is common for one outlet to publish multiple AI-related articles per day, so in each window, the pipeline aggregates by source and sentiment. Finally, it writes each aggregation containing the outlet name, sentiment, window start and end time, and total to a text file. The resulting text file is then used to create a bar chart, with the total number of positive/negative/neutral headlines for each news outlet, over the course of the past month.

The above bar chart shows several interesting results. The news outlets that covered Artificial Intelligence most frequently were Engadget, Wired, Gizmodo, Google News, and Business Insider. This is not surprising, as these outlets are primarily dedicated to covering technology and industry news. AI also received wide coverage from mass media sources, such as Fox News, BBC, and The Guardian. Almost every outlet, with the notable exceptions of BBC, The Verge, and The Guardian, appeared to give mostly positive coverage to AI. Interestingly, not a single news outlet in the past month appeared to cover AI from a purely neutral point of view. Subsequent analyses and testing could explore this. For instance, it is possible that my Headline Sentiment function could have undefined behavior, or it could be that it is standard within the sampled news outlets to sensationalize headlines. A more detailed query to NewsAPI, or other Big Data sources on journalism, could explore factors that influence positive or negative coverage of artificial intelligence, such as the outlet's target demographic, ownership, or political leanings.

In conclusion, I was able to run the Apache Beam pipeline both on my machine, for convenient testing, and on GCP. I ran the pipeline on GCP twice, to compare the effects of parallelization. The first run (see image below, second row) did not include the lines of code related to WorkerOptions, so Beam determined the number of workers on its own, which turned out to be 1. The second explicitly set the number of workers to 2, which resulted in a very slight improvement in speed. This is to be expected. Given that the input data file is relatively small, the speed gained by parallelization would be insignificant, and largely offset by the overhead associated with splitting the data across multiple threads or CPUs. If I were to test the efficiency of this code's parallelization, I would need to obtain a much larger input file, using the enterprise version of NewsAPI. I predict that if it were to contain at least 1GB of news articles, Beam would start multiple workers by default.



The screenshot shows the Google Cloud Dataflow console. The 'Jobs' tab is selected, displaying a table of job runs. Two jobs are listed, both with a status of 'Succeeded'. The first job started at 1:56:56 PM and the second at 1:30:49 PM, both on May 6, 2023. Both jobs used SDK version 2.46.0 and ran in the us-east1 region. Each job has an 'INSIGHTS' icon next to it.

Name	Type	End time	Elapsed time	Start time	Status	SDK version	ID	Region	Insights
headline-sentiment-analysis-job	Batch	May 6, 2023, 2:00:58 PM	4 min 2 sec	May 6, 2023, 1:56:56 PM	Succeeded	2.46.0	2023-05-06_10_56_55-433876184134267645	us-east1	1 INSIGHT
headline-sentiment-analysis-job	Batch	May 6, 2023, 1:35:08 PM	4 min 19 sec	May 6, 2023, 1:30:49 PM	Succeeded	2.46.0	2023-05-06_10_30_47-14443803215132054835	us-east1	1 INSIGHT