Comparative Study of Sentiment Analysis of Reddit posts with AWS and Google Cloud

Anagha Bapat
MS Computer Science
Santa Clara University
Santa Clara, CA
Abapat@scu.edu

Manas Sadhwani
MS Computer Science
Santa Clara University
Santa Clara, CA
msadhwani@scu.edu

Siddarth Palagrahara Chandrashekar
MS Computer Science
Santa Clara University
Santa Clara, CA
spalagraharachandras@scu.edu

Yash Patki
MS Computer Science
Santa Clara University
Santa Clara, CA
ypatki@scu.edu

Abstract - Sentiment evaluation is the computational look at reviews, attitudes, opinions, sentiments, and feelings expressed within the text. It basically refers to a classification problem where the focus is to show the polarity of phrases and then classify them into positive or negative or mixed sentiment. Reddit sentiment analysis gives us a broad and effective manner to measure the public's feelings about any post about the platform. The usage of cloud applications is increasing drastically for enterprise and analytics. Cloud computing offers benefits such as virtualization, resource management, and the elimination of cluster setup cost and time. Using services offered by cloud providers offer benefits like scalability, ease to apply, cost-effectiveness, and performance. There are a number of cloud providers that offer many natural language processing techniques. But choosing which cloud providers to go with would need a little research. In this paper, we purpose to offer a detailed comparative look at leading cloud service companies Google and Amazon for their NLP offerings taking into account metrics like accuracy, cost, features, and computation time.

Keywords—Social Media; Sentiment Analysis; Natural Language Processing; AWS; Google Cloud; Cloud Computing; Opinion Mining

I. Introduction

In the world where people express their opinion on the internet which ends up generating huge chunks of data most of which is actually unstructured. Making sense of this data and analyzing the sentiment of the people towards a certain person/object of attention can help taking important business decisions. However, today almost 3 quintillion bytes of data is generated every day and to make sense of such kind of data can get tricky. Due to the increase in storage capacity and improvement of technology when it comes to working with large amounts of data there are both opportunities and challenges when it comes to analyzing the humongous amounts of data we generate.

Sentiment analysis is a part of Natural Language processing that creates a way to extract the feelings regarding a certain topic from the given text. It involves the use of natural language processing tools and computational linguistics to firstly identify and then extract the data sending it through a function to quantify and provide a sentiment score to the text. The use of sentiment analysis is in many fields from marketing a product to analyzing the opinion of the people towards a politician, analyzing social media responses to customer service provided by companies.

II. Problem Analysis

A. Motivation

There has been a lot of research and development when it comes to natural language processing. As sentiment analysis is a part of NLP there has been tremendous research when it comes to sentiment analysis. The amount of data that is available now thanks to the growth in social media and access to the internet, people post about a variety of topics, celebrities, products and politicians on their social media. Using sentiment analysis we get insights about the data on a wide scale in order to know the pulse of the people. The opinion that is obtained from the use of such algorithms can be used to improve the services provided to the people. The dataset gathered from these operations provide the mood of the public whether it is positive, negative or neutral regarding a topic. Supposedly the opinion is turning negative, measures can be taken by the business to avoid it from getting worse and when it’s positive it can help the businesses to know what's right about their marketing strategy.

B. Problem Statement

When it comes to companies who want to know what the people think about their product or a politician who wants to know how the people respond to their interviews and speeches. The challenge of doing sentiment analysis when there are loads of comments being posted for a post at a rapid pace, you need to have the bandwidth, memory and the processing power to process these comments too on a stand alone system. In order for the system to work there must be natural language processing which requires machine learning acumen and serverless computing which is event driven to handle the requests. As the data enters it must trigger the natural language processing functions to analyze the data. There should be negligible downtime and minimal fault tolerance on these systems.

C. Role of Cloud Services

Sentiment analysis can be performed in a traditional way where there is no need to set up the cloud to analyze and store the data, but cloud services provide many benefits. Nowadays businesses and industries have moved to the cloud, as it offers benefits of virtualization, availability of resources on demand and no set up cost in the initial phase. When it comes to analyzing the data, cloud has benefits like scalability, ease of use, cost benefits and improved performance. In the cloud the customer pays for only what they use and it is good for AI and machine learning workloads. The businesses can test their capability and scale up when it goes in production. There is no need for special skills in data science and AI to use the available Natural language APIs offered by cloud services. Amazon, Google and Microsoft provide options which don’t require deep knowledge of Artificial Intelligence or ML theory.

D. Contribution

We aim to provide a comparative study of two leading cloud service providers Google and Amazon for their NLP services while comparing them with respect to various metrics like accuracy, cost, performance and features. We will also provide a comparison with a traditional approach for sentiment analysis and NLP using NLTK library in Python and provide the comparison on accuracy with preexisting labelled data and performance by using the compute services provided by both these cloud providers. We will also compare the cost of using these services for different cases and scenarios.

We aim to build a real-time Reddit Sentiment analysis application on Google cloud and AWS platform to evaluate the sentiments of comments for a specific subreddit.

III. PROPOSED SOLUTION

We have come up with a system that utilizes Google Cloud Services/Amazon Web Services to get the sentiment of the replies to a post on Reddit. We leverage the offerings of instances that can scale the application by supplying adequate resources like memory, bandwidth, and processing speed to run the service. AWS and GCP incorporate a highly powerful natural language processing API called Amazon Comprehend and Cloud Natural Language, respectively. This API is triggered based on the entry of data into the memory buckets. To perform event-driven serverless computing, we incorporate Lambda functions and cloud functions.

A. Proposed Solution on Amazon Web Services

Step 1.Data Collection:

This step involves retrieving comments to a Reddit post, this is done using PRAW, a Python Reddit API Wrapper. A python script is used for this purpose and the address to the Reddit post will be passed as an URL to this script. Along with retrieving the comments, the script also pre-processes the data to remove stopwords, emojis, special symbols, and URLs.

Step 2.Storing the pre-processed data:

In this step, we are trying to store the data in a storage space which would make it convenient and easily accessible to further operations on it. This is achieved by using Kinesis firehose delivery stream which makes sequential processing of data easier. Once the data is retrieved and pre-processed, this data is then transferred to a bucket in S3 using the firehose delivery stream.

The reason to use Kinesis firehose is because it automatically scales to match the throughput of data and requires no ongoing administration. It can also batch, compress, transform, and encrypt the data before loading it, minimizing the amount of storage used at the destination and increasing security. The data collected is available in milliseconds to enable real-time analytics.

Step 3. Getting the sentiment of the data stored in S3:
As soon as the data is stored in an S3 bucket the system is configured to trigger the AWS Lambda function where the actual sentimental analysis is performed. The Lambda function takes the data stored in the S3 bucket as the input and uses AWS comprehend services to get the sentiment of the comment. The output of the lambda function will have the general sentiment of the whole post, indicating if its positive, Negative or Neutral and a score associated with each comment indicating if its positive, negative or neutral.
Amazon Comprehend is a natural language processing (NLP) service that uses machine learning to find insights and relationships in text.
AWS Lambda is a serverless compute service that runs the code in response to events and automatically manages the underlying compute resources. It is also "stateless" so it can rapidly launch as many copies of the function as needed to scale to the rate of incoming events.

Step 4: Data Visualization:
The output of the lambda function is loaded into Elastic search, which can power extremely fast searches that support data discovery applications. Elastic search also provides tight integration with Kibana which is an open source data visualization and exploration tool which is helpful for visualising the data loaded into ElasticSearch.

B. Proposed Solution on Google Cloud Platform

Step 1. Data Collection and VM Instance setup: A VM instance is created on google cloud platform and a python script is executed to retrieve replies on a reddit post passed as an URL to the script, by accessing a Python Reddit API Wrapper. The script screens and processes the data collected by removing special symbols, emojis, URL and stop words. This setup enables us to create a robust system that can scale according to the application needs. The VM memory and processor specs are defined to cater to required data pipeline and speed requirements.

Step 2. Data Ingestion: The retrieved Reddit comments are then published to Google Pub/Sub. Google Pub/Sub is an asynchronous delivery service for event-driven messaging for data ingestion and data delivery based on subscribing to a topic. It supports the publisher-subscriber paradigm.

Step 3. Sentiment Analysis: We make use of Google Cloud Functions to carry out Sentiment analysis. Cloud Functions is Google Cloud's event-driven serverless computing platform. It provides automatic scaling, and it is fault tolerant and highly available.A Google Cloud Function will get triggered by the publish event from Pub/Sub which will run in the background. The Google Cloud Function will call the Natural Language Processing API. The Natural Language API provides powerful pre-trained models which allow us to work with natural language to perform and understand sentiment analysis. The sentiment analysis results will be stored in the form of logs.

Step 4. Data Storage and Visualization: Beats is a lightweight client which is used as a data-shipper to send data from multitude of sources including Pub/Sub to elasticsearch. We have used pubsubbeat which is an elastic beat for Google pub/sub for our implementation. This beat subscribes to a topic and ingests messages. In our case it will ingest the google cloud function logs and send them to elasticsearch ingest nodes. We use Elasticsearch to store all the log information. Elasticsearch provides the capability to store, search and analyze huge volumes of data quickly and provide results in milliseconds. We use ingest pipeline to transform some log information as a prior step before storing logs into elasticsearch. We use Kibana to visualize the results and classify the sentiments and build reports.

IV. IMPLEMENTATION

A. Solution Using Amazon Web Services

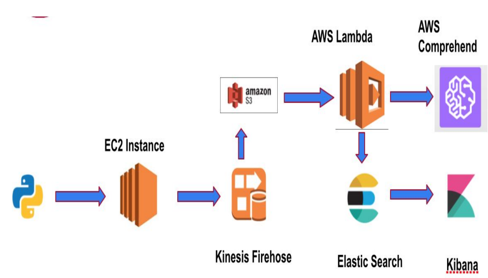

Figure 1:AWS Setup for Sentiment Analysis

Step 1.Getting the reddit comments:

Since we are finding the sentiment of the comments for a reddit post, retrieving the comments for the post is the first step. We extract the comments from a post by using PRAW. Client application is registered with reddit by providing reddit ID keys are environment variables in the script. This establishes a streaming session with the application. The Data retrieved is then cleaned to remove all the stop words, emojis, URLs and special characters. This is done as these carry no sentiment value. This makes the results more accurate.

Step 2. Data Ingestion and Lambda Invocation:

The comments ingested into a delivery stream created using the AWS Kinesis Firehose. This is done using the boto3, an AWS SDK toolkit for python. The data is ingested using the Client.PutRecord API. The data from the delivery stream is then put into a bucket in S3. As soon as the data reaches the bucket, this triggers the lambda function to run on the date in the bucket and invokes the Amazon comprehend API to get the sentiment of the comment.

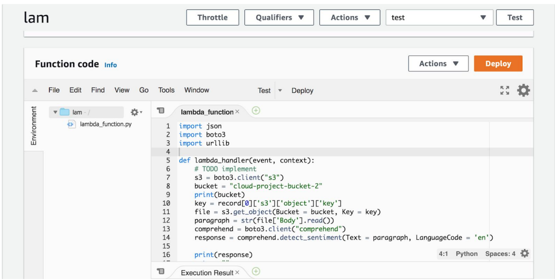

Figure 2: lambda function which is triggered when data is put into S3

Step 3. Performing the sentiment Analysis:

With the Lambda function triggered, the AWS Comprehend is invoked. AWS comprehend is a sentiment analysis tool by amazon which uses machine learning to help you uncover the insights and relationships in your unstructured/ structures data. AWS comprehend deduces the generic sentiment of data as either positive, negative or neutral. Along with getting the overall sentiment of the data, it also gives the sentiment score which is basically the average of the positive and negative sentiment of the comments.

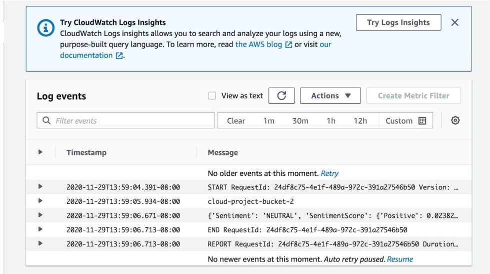

Figure 3: AWS cloud watch logs stored in AWS Cloud Watch

Step 4. Data Storage and Visualization:

The data logs from the lambda function can be viewed in the cloud watch. CloudWatch collects monitoring and operational data in the form of logs, metrics, and events, providing you with a unified view of AWS resources, applications, and services that run on AWS and on-premises servers. The logs from the Cloudwatch are then pushed into ElasticSearch. From here with the tight integration with Kibana virtualisation of the output data becomes easier. Kibana, an open source data visualization plugin for Elasticsearch is used to visualize the data in various forms of graphs.

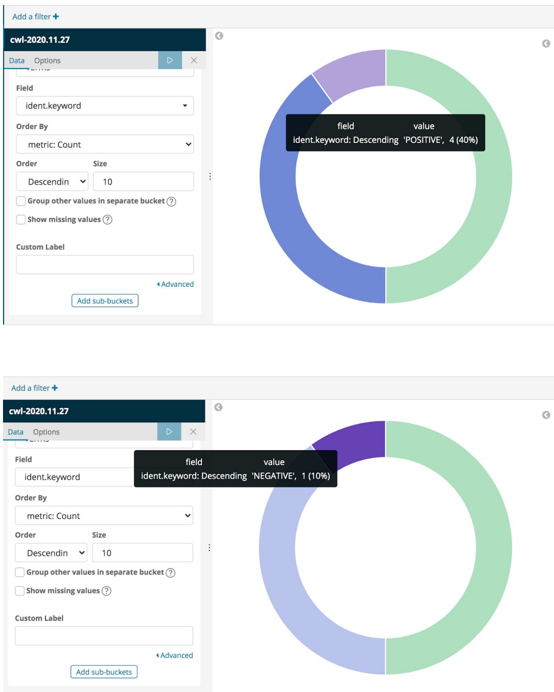

Figure 4: Data Visualization in Kibana on the basis of scores of sentiment analysis.

B. Solution Using Google Cloud Platform

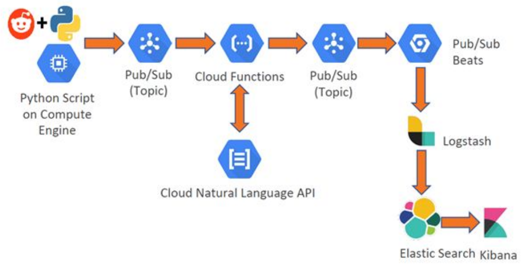

Figure 5:Google Cloud Setup for Sentiment Analysis

Step 1. Data Collection and the Instance: We extract the Reddit comments belonging to a post by using Python Reddit API Wrapper. The registered reddit ID keys are initialized in the python script which helps to establish a streaming session with the application. Data retrieved undergoes the process of data preparation by removing all the stop words, emojis, URLs and special characters. This prepares the data to carry out accurate sentiment analysis.

Python3 is installed on the VM instance using pip command. Other necessary installations like jdk, cloud-sdk and google-cloud-pubsub are performed on the VM instance. Python Script captures the Reddit comments for a specific post passed as an argument on the Google Cloud Instance (Compute Engine). The python script acts as a publisher and publishes all the captured comments to google Pub/Sub topic.

```
def publish_to_pubsub(rpost):
    # Data must be a byte string
    rpost = rpost.encode('utf-8')
    # When you publish a message, the client returns a Future.
    message_future = publisher.publish(topic_path, data=rpost)
    message_future.add_done_callback(callback)
```

Figure 6: Publish Reddit comments in script to Pub/Sub

Step 2. Data Ingestion: Cloud Pub/Sub provides a staging location for data based on event (Reddit data streaming) on its journey towards storage as logs. The publisher application creates and publishes messages to a topic. Subscriber applications create a subscription to a topic to receive messages from it. Communication can be one-to-many (fan-out), many-to-one (fan-in), and many-to-many. We have created "analysereddit-topic" topic in Pub/Sub. The Reddit comments are published to this topic based on the event of their arrival. A subscription "gcf-sentiment_cloudfunction-us-central1-analysereddit-topic" is created on the mentioned topic when the cloud function receives this data.

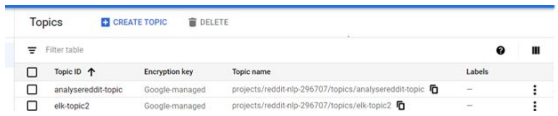

Figure 7: Topics defined in Pub/Sub

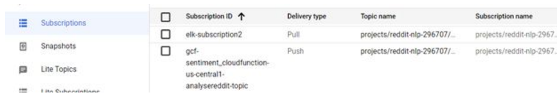

Figure 8: Subscriptions defined in Pub/Sub

Step 3. Sentiment Analysis: Google Cloud Function is an event-driven serverless execution environment where the user's function is attached to the event of Reddit comments published to Pub/Sub. Cloud function invokes Cloud Natural Language API where staged data is converted to a structured JSON format with the fields "post", "score", "magnitude". The score ranges from -1 to +1, but the user has the flexibility to define a custom range. Google defines 0.25 to 1.0 as positive sentiment, -0.25 to +0.25 as neutral sentiment, and -1 to -0.25 as negative sentiment; however, we can achieve finer granularity by classifying the reddit comments as highly positive, positive, neutral, negative and highly negative based on the score and magnitude values.

The results of the sentiment analysis from Google Cloud Function will be stored in the form of Logs.

Figure 9: Google Cloud function with trigger set on topic analysereddit-topic

```
if 'data' in data:
    try:
        post = base64.b64decode(data['data']).decode('utf-8')
    except Exception:
        post = data['data']
        print('not base64 encoded')
        pass

# print('Hello {}!'.format(post))
"""Run a sentiment analysis request on text within a passed filename."""
client = language_v1.LanguageServiceClient()

text = post

document = language_v1.Document(content=text, type_=language_v1.Document.Type.PLAIN_TEXT)
encoding_type = language_v1.EncodingType.UTF8
annotations = client.analyze_sentiment(document=document)


score = annotations.document_sentiment.score
adjusted_score = (score + 1) * 5
magnitude = annotations.document_sentiment.magnitude
import json
#dic = {"post": str(post), "score": str(adjusted_score), "magnitude": str(magnitude)}
dic = {"post": str(post), "score": (adjusted_score), "magnitude": (magnitude)}
print(json.dumps(dic))
```

Figure 10: Cloud Function code snippet invoking Google cloud NLP

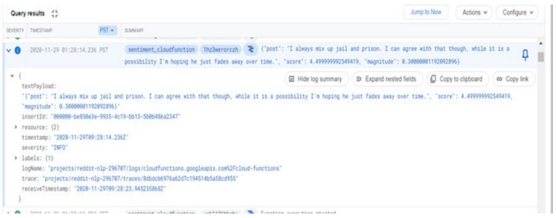

Figure 11: Google Cloud logs created

Step 4. Storage and Visualization: PubSubBeat is a type of beat that serves as an open-source platform for a lightweight data shipper subscribed to Pub/Sub. Beats can send the data directly to Elasticsearch. In our implementation 'pubsubbeat' which is an elastic beat is subscribed to the topic – "elk-topic2". A "sink" is created in Log Router on this topic chosen as destination which will include logs from our Cloud Function. This will help us to export the logs from the Google Operation Logs to this topic. The 'pull' subscription on this topic will enable the ingestion of Google Cloud Logs consisting of sentiment analysis results and send the data to Elasticsearch.

Figure 12: Data Flow to Elasticsearch and visualization in Kibana

Kibana is used to visualize the data present in Elasticsearch. In order to enable the visualization of the log data in Kibana, we have defined and uploaded an ingest pipeline which processes JSON structure string to create JSON objects (for score and magnitude) to Elasticsearch. These JSON objects can be used to create visualizations in Kibana to carry out various analyses. Whenever the Sentiment analysis logs from Google Cloud function are received by Beats, they will first get passed through this ingest pipeline and then will be indexed in Elasticsearch.

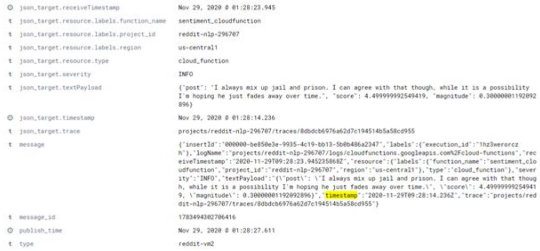

Figure 13: Raw log message stored in Elasticsearch

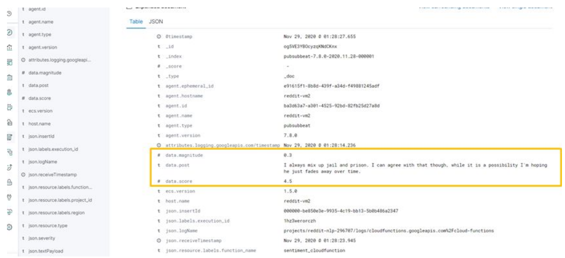

Figure 14: Fields for score and magnitude in Elasticsearch

Visualizations can be created and viewed in Kibana based on the score for sentiment analysis.

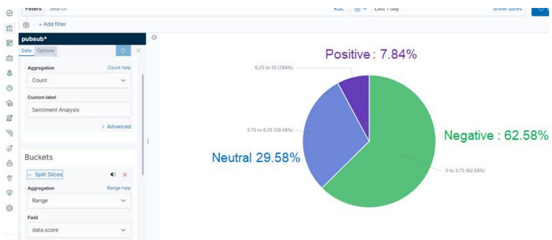

Figure 15: Data Visualization in Kibana on the basis of scores of sentiment analysis.

V. Evaluation

A. Performance

For performance testing, we tested the data on different subreddits with varying numbers of comments. We executed the code with the AWS Comprehend service, Google NLP and (Natural Language Toolkit) NLTK. The data using AWS Comprehend was run on an AWS EC2 instance, the Google NLP was run on Google Compute Engine and we ran NLTK locally for comparison and to set benchmarks.

Comments	GCP	AWS	NLTK
365	46.48s	29.08s	4.21s
730	1m 39s	1m 2s	16.39s
4,200	11m 8s	Error	2m 36s
1771	4m 40s	2m 48s	1m 7s

Table 1: Performance Comparison

We noticed that AWS Comprehend running on AWS EC2 was faster in every case compared to Google NLP. NLTK was faster than both since it was running locally and there were no latencies. We also noticed that AWS has a limit for the input size not exceeding 5000 bytes. GCP allows content size of 1,000,000 bytes. Google limits the number of requests per minute to 600 and per day to 800,000. This is not ideal when number of API requests/min are high as it leads to time out error.

Cost Analysis

AWS Comprehend requests Sentiment Analysis, are measured in units of 100 characters, with a 3 unit (300 character) minimum charge per request.The free tier includes 50K units of test (5M characters).

Up to 10M units	From 10M-50M units	Over 50M units
$0.0001	$0.00005	$0.000025

Table 2 : Price per unit for AWS Comprehend

Google Natural Language API usage is calculated in terms of “units,” where each document sent to the API for analysis is at least one unit. Documents that have more than 1,000 characters are considered as multiple units, one unit per 1,000 characters. The table below provides the price per unit based on the total number of units analyzed during the billing month.

0 to 5K	From 5K to 1M	1M to 5M	5M-20M
Free	$0.001	$0.0005	$0.00025

Table 3: Pricing for Google NLP

Comparison of the costs of AWS and GCP

Example Use Case	Records	Characters per record	Amazon	Google
Customer Review Sentiment Analysis	10,000	550	$6	$10
Twitter Sentiment Analysis	1,000,000	75	$100	$1,000
Document Topic and Sentiment Analysis	5,000	10,000	$51	$100

Figure 16: Price Comparison

Thus, we can see that although AWS Comprehend is cheaper in most cases, in cases with a large data set size and lesser requests GCP NLP costs comparatively less and AWS Comprehend costs less for a smaller dataset size and more requests.

B. Accuracy Comparison

There are sentiment analysis training sets available online for testing the accuracy of the sentiment analysis services. These datasets have been previously verified by a sentiment analysis professional. We test the accuracy of Google NLP and Amazon Comprehend on these models provided.

SubReddits on Indian Prime Minister Election 2019

The subreddit dataset by Kaggle had comments of people expressing opinions about the 2019 Lok Sabha elections in India. The nature of the tweets was unstructured with grammatical mistakes, there were errors in spelling and diction therefore making it difficult to be analyzed.

- Among the total number of comments on the subreddit 41.28 percent were positive, 22.22 percent were negative and 35.28 percent neutral comments as per the analysis by the dataset provider.

- Google NLP: 57.1 percent of the comments matched correctly whereas 42.9 percent matched incorrectly when compared to the given sentiment.
- Amazon Comprehend: 45.7 percent of the comments matched correctly whereas 54.3 percent matched incorrectly.

On Comparison between the sentiment provided by Google NLP and Amazon comprehend the trend was that Google NLP gave a conviction on negative sentiments and did a better job at identifying the negative comments whereas Amazon Comprehend having a mixed sentiment score for certain negative comments which goes on to show that Google NLP is highly polar in nature and provides results in that manner. When it comes to structured data Google NLP does a better job with 93 percent of matches compared to 80 percent matches on Amazon Comprehend.

Sarcasm on reddit: When it comes to reddit statements there may be some form of sarcasm in these comments. When testing with Google NLP and Amazon Comprehend for Sarcastic comments both the services struggled to provide the correct result but Google NLP did a much better job than Amazon Comprehend in getting a match.

We can see that each service may perform better or worse depending on the type of input we provide to them and the nature of the comments. The goal of the sentiment analysis would help in deciding which service would work better for the dataset being used.

If the dataset has comments which are highly inclined towards a positive or negative sentiment i.e highly polar in nature, in this case Google NLP does a better job. If the dataset has comments which are less polar in that case Amazon Comprehend is a better choice. When it comes to faster execution AWS is a better choice.

Comparing these results to NLTK we see more accuracy with 65 percent matches and 35 percent mismatches. But this highly depends on the nature of the dataset and how polar the comments are in the dataset. However, NLTK failed to perform when it came to Sarcasm detection in the comments. Cloud services would also have certain advantages over a locally run solution. Some of the advantages of using cloud services for sentiment analysis would be higher scalability with the ability to scale up or scale down our operation and storage needs quickly to suit your situation, allowing flexibility as your needs change. Rather than purchasing and installing expensive upgrades yourself, your cloud computer service provider can handle this for you, efficiency in collaboration with the ability to communicate and share more easily outside of the traditional methods as well as access to automatic updates where the models of these services will keep improving over time using methods like machine learning and deep learning.

VI. DISCUSSION AND RELATED WORK

In [1], the authors have proposed an open source sentiment analysis SaaS tool on the cloud which extracts distinct subjective features and calculates the polarity. They are classified into positive, negative and neutral. The authors have leveraged cloud services to make the tool scalable, available and secure. Various APIs are mapped to their advantage in the tool and it has been provided as a cloud solution. In [2], the author has analysed the sentiment of tweets obtained from twitter using R and R Studio. It has utilised cloud solution AWS S3 to store the data. It has given the various advantages and challenges faced using the services provided by cloud providers while performing sentiment analysis on the cloud. In [3], the authors have proved a comparative study of using Azure and AWS for sentiment analysis of Twitter data. They have given emphasis on the polarity and emotion classification using the Naive Bayes Classifier after collecting and preprocessing the data. It gives us a perspective of the advantages, challenges and the features provided by AWS and Azure and their comparison. The authors of [4] have created a tool called “RIVA : A Real-time Information Visualization and Analysis Platform for Social Media Sentiment Trend”. They have provided sentiment analysis on Twitter data and used a Spark cluster for streaming, processing and storage. They have used a web crawler to get the data from twitter. Data visualization and analysis is provided using Zeppelin. They have provided a correlation between the web news and Twitter sentiment for certain topics.

VII. CONCLUSION

In this project, we have developed a SaaS solution for providing real time sentiment analysis on Reddit data using the services provided by AWS and GCP. AWS Comprehend and Google Natural Language API were compared on the following metrics.

Speed: Although both the platforms have certain tradeoffs, we can conclude saying that to achieve a system with a higher speed, AWS is a better option.

Accuracy and Granularity: Google Cloud Platform's CLoud Natural Language API is more accurate than AWS Comprehend. It is better for more polar data and handles sarcasm the best compared to AWS Comprehend and NLTK. The user's have the ability to define the values of the score in the cloud function and can make the classification granular by using these scores.

Compatibility with 3rd Party Apps: Often the user would like to integrate the service with third party apps. One such case would be to store the ingested logs in a stash. The stored logs could be further indexed into key value pairs and used to visualize in external services. Configurability with AWS is fairly easy because of the simplicity of json response. Kibana is built-in within Amazon Web Services, and the ingested data could be easily directed to Kibana and then to Logstash. In this aspect, integrating AWS with ElasticSearch and Kibana is more seamless compared to GCP as they do not provide a lot of options for the same.

Visualization: The integration of AWS with Elastic, makes it fairly simple to visualize the ingested logs as dashboards and reports. Google Cloud Platform does not have a default visualization tool. Incompatibility of nested json results from GCP's logging system, makes visualization with Kibana a tedious process. AWS is a clear winner in this regard.

Reliability: Both Google Compute Engine and AWS EC2 both have SLAs which provide a monthly uptime percentage of at least 99.95%. With Google Compute Engine, we can use Stackdriver to monitor the health of the system. CloudWatch in AWS is a counterpart of Stackdriver.

AWS: The user has the ability to get different machines within their multiple availability zones per region.

Google Cloud has the ability to live migrate virtual machines. Live migrations allow you to resolve patching issues, repairing the software and updating the hardware and software without the overhead of rebooting.

Price: In cases with a large data set size and lesser requests GCP NLP costs comparatively less and AWS Comprehend costs less for a smaller dataset size and more requests.

REFERENCES

[1] Omkar Sunil Joshi, Garry Simon, "Sentiment Analysis Tool on Cloud: Software as a Service Model" International Conference On Advances in Communication and Computing Technology (ICACCT), 2019

[2] Amisha Tiwari, "Sentiment Analysis in R on AWS cloud", International Research Journal of Engineering and Technology, 2017

[3] Laila M. Qaisi, Ibrahim Aljarah, "A Twitter Sentiment Analysis for Cloud Providers: A Case Study of Azure vs. AWS" International Conference on Computer Science and Information Technology (CSTT), 2016

[4] Yong-Ting Wu, He-Yen Hsieh, Xanno K. Sigalingging, Kuan-Wu Su, Jenq-Shiou Leu, "RIVA : A Real-time InformationVisualization and Analysis Platform for Social Media Sentiment Trend", 2017

[5] https://www.deducive.com/blog/2018/6/02/using-r-for-sentiment-analysis-with-aws-comprehend-google-cloud-natural-language-ibm-watson

[6] https://mindcraft.ai/automated-nlp/

[7] https://github.com/googlearchive/pubsubbeat

[8] https://cloud.google.com/functions/docs/tutorials/pubsub

[9] https://cloud.google.com/natural-language/docs/basics

[10] https://www.deducive.com/blog/2018/6/02/using-r-for-sentiment-analysis-with-aws-comprehend-google-cloud-natural-language-ibm-watson