Real-time Twitter Sentiment Analysis with AWS and Google Cloud

Inchara Raj
School of Engineering
Santa Clara University
Santa Clara, CA
iraj@scu.edu

Nitya Navali
School of Engineering
Santa Clara University
Santa Clara, CA
nnavali@scu.edu

Priyanka Satish
School of Engineering
Santa Clara University
Santa Clara, CA
psatish@scu.edu

Suraj Bagaria
School of Engineering
Santa Clara University
Santa Clara, CA
sbagaria@scu.edu

***Abstract*— Sentiment analysis is the computational study of opinions, sentiments, evaluations, attitudes, views and emotions expressed in text. It refers to a classification problem where the focus is to predict the polarity of words and then classify them into positive or negative sentiment. Sentiment analysis over Twitter offers people a fast and effective way to measure the public's feelings over any topic of interest. Cloud applications are increasingly being used in industries for business and analytics. Cloud computing offers the benefits of virtualization, the elasticity of resources and elimination of cluster setup cost and time. Using cloud service for analyzing the data offers several benefits like scalability, ease to use, cost effective and efficiency. There are number of cloud service providers who offer many natural languages processing services. It could be difficult for the Cloud Customer to decide the best provider. In this paper we aim to provide detailed comparison study of two leading cloud service providers Google and Amazon for their NLP services considering various metrics like accuracy, cost, features and computation time. We also present SAAS model for analyzing the sentiments of tweets in real time on Google Cloud and Amazon AWS platform**.

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

I. INTRODUCTION

The world has experienced a tremendous increase in the volume of textual data in recent years, especially for the unstructured data generated by people expressing opinions through different web and social media platforms. In a world where every day we generate 2.5 quintillion bytes of data, sentiment analysis has become a key tool to make sense of that data. Multitude of textual data initially could be equated to garbage which would need to be disposed from time to time. However, with the advancement in storage capacity accompanied by the increasing sophistication in data mining tools, opportunities and challenges have been created for analyzing and deriving useful insights from these mountains of data.

Sentiment Analysis also known as Opinion Mining is a field within Natural Language Processing (NLP) that builds systems that try to identify and extract opinions within text. Sentiment analysis refers to the use of natural language processing, text analysis, computational linguistics to systematically identify, extract, quantify, and study affective states and subjective information. It is widely applied to the voice of customer materials such as reviews and survey responses, online and social media, and healthcare materials for applications that range from marketing to customer service to clinical medicine.

II. PROBLEM ANALYSIS

A. Motivation

Sentiment analysis is a topic of great interest and development since it has many practical applications. Social media have evolved to become a source of varied kind of information. This is due to nature of social media on which people post real time messages about their opinions on a variety of topics, discuss current issues, complain, and express positive sentiment for products they use in daily life. Sentiment Analysis provides us the ability to gain insights from data obtained from social media. Opinions that are mined from such services can be valuable. Datasets that are gathered can be analyzed and presented in such a way that it becomes easy to classify the online mood to positive, negative or even indifferent. This allows individuals or business to be proactive when a negative conversational thread is emerging. Alternatively, positive sentiment can be identified thereby allowing the identification of product advocates or to see which parts of a business strategy are working.

B. Problem Statement

Sentiment analysis is essential for leveraging people's opinion into a score to categorize their emotion towards any entity. It is a challenging task to perform sentiment analysis when the input is a stream of real time data because of scarcity of resources like memory, bandwidth, processors, availability and fault tolerance on a stand- alone system. It is a necessity to have natural language processing systems which requires machine learning acumen and serverless event driven computing service to handle traffic. The entry of traffic must be a trigger to natural language processor to analyze the textual content. A persistent system which can run the application with

negligible down time must be available with minimal fault tolerance.

C. Leveraging Cloud Services

There are many existing approaches that perform sentiment analysis in a traditional way but leveraging cloud services to build one offers many benefits. Cloud applications are increasingly being used in industries for business and analytics. Cloud computing offers the benefits of virtualization, the elasticity of resources and elimination of cluster setup cost and time. Using cloud service for analyzing the data offers several benefits like scalability, ease to use, cost effective and efficiency. The cloud's pay-per-use model is good for burst AI or machine learning workloads. It makes it easy for enterprises to experiment with machine learning capabilities and scale up as projects go into production. The cloud makes intelligent capabilities accessible without requiring advanced skills in artificial intelligence or data science. AWS, Microsoft Azure, and Google Cloud Platform offer many machine learning options that don't require deep knowledge of AI, machine learning theory, or a team of data scientists.

D. Contribution

- We aim to provide detailed comparison study of two leading cloud service providers Google and Amazon for their NLP services considering various metrics like accuracy, cost, features and computation time.
- We will research on how accurately these services predict the sentiment for different dataset. We will be using a preexisting labelled data set as a benchmark.
- We aim to build a real-time Twitter Sentiment analysis application on Google cloud and AWS platform to evaluate the sentiments of tweets for a specific entity.
- We will be providing an insight into how much each cloud platform charge to provide the cloud services by considering various cases and scenarios.

III. Proposed Solution

We have proposed a system that leverages Google Cloud Services/ Amazon Web Services to ingest real time data. We take advantage of instances which can scale the application by supplying adequate resources like memory, bandwidth and processing speed to run the service. Google Cloud services/ AWS incorporates a highly powerful natural language processing API called Cloud Natural Language/ Amazon Comprehend. This API is triggered based on the entry of tweets. To perform event driven serverless computing, we incorporate Cloud Functions/ Lambda functions.

A. Proposed Solution on Amazon Web Services

Step 1. Data Collection: In this step tweets with specific hashtag will be extracted using Twitter standard search API. A python/Java script will be implemented to fetch the tweets by invoking the Twitter Search API and then tweets will be pre-processed to remove unnecessary details which might affect the result like acronyms, URLs, hash tags and emotional symbols.

Step 2. Data Streaming: The tweets collected in previous step are fed into Kinesis FireHose delivery stream so that data gets processed sequentially and incrementally on a record-by-record basis or over sliding time windows. Amazon Kinesis Data Firehose is a fully managed service for delivering real-time streaming data to destinations such as Amazon Simple Storage Service (Amazon S3), Amazon Redshift, Amazon Elasticsearch Service (Amazon ES), and Splunk. 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. Sentiment Analysis: As soon as the tweet enters Kinesis FireHose delivery stream the system would be set to trigger AWS Lambda function where the actual sentimental analysis is performed. AWS Lambda is a serverless compute service that runs the code in response to events and automatically manages the underlying compute resources. The input to AWS lambda will be the tweets and these tweets are analyzed by invoking services provided by Amazon Comprehend. Amazon Comprehend uses machine learning to help uncover the insights and relationships in your unstructured data. The output will have the tweet and a score associated with each tweet indicating if its positive, negative or neutral. Lambda functions are "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 Storage and Visualization: The result from the previous step is loaded into Elastic Search. Elasticsearch is an open-source, broadly-distributable, readily-scalable search engine. Elasticsearch can power extremely fast searches that support data discovery applications. Elastic search provides tight integration with Kibana an open source data visualization and exploration tool which will help to visualize the results in the form of charts. The tweet and the sentiment data will be stored in an Elasticsearch domain where we can visualize real time information using custom charts.

B. Proposed Solution on Google Cloud Platform

Step 1. Data Collection and the Instance: An instance is created on google cloud platform and a python script is run to capture tweets from the Twitter API. It enables us to create a system that can scale the application, a system that is robust and persistent. The instance is created to cater to the memory requirement, bandwidth and processor to scale to the required speed.

Step 2. Data Ingestion: The tweets are published to Pub/Sub. Pub/Sub is one of the cloud services that can handle real time data. Pub/Sub is a google cloud delivery service for event-driven messaging for data ingestion and data movement based on subscribing to a topic. It supports publish- subscribe paradigm.

Step 3. Sentiment Analysis: Cloud Functions is Google Cloud's event-driven serverless computing platform. It automatically scales and it is fault tolerant and highly available. Google cloud function will be triggered by publish event from Pub/Sub which would run on the background. Google cloud function will call the Natural Language API. The Natural Language API has powerful pre-trained models which allow us to work with natural language to perform and understand sentiment analysis.

Step 4. Data Storage and Visualization: Logstash is a product of elastic which enables processing server-side data processing pipeline that ingests data from a multitude of sources including Pub/Sub. Logstash performs data transformation simultaneously and stores it. It is possible to parse and transform data into a structure and store it in our choice of "stash". We use Kibana to visualize the results and classify the sentiments and build reports.

IV. Implementation

A. Solution Using Amazon Web Services

Step 1. Data collection and preprocessing: Extracted real time tweets using Python filtered on a specific entity using open source library Tweepy. Tweets of only English dominant language are filtered in the script. Client application is registered with Twitter by providing consumer keys and authorization tokens in each request to validate the identity of the source. A streaming session is established, where tweets are fetched and routed to listener which is responsible for data cleaning. Data cleaning involves removing extraneous information which can affect the accuracy of result.

Step 2. Data Ingestion and Lambda Invocation: Tweets are ingested to Amazon Kinesis FireHose delivery system using boto3 SDK toolkit for Python. Information of each tweet is wrapped in the form of JSON which includes created_date, location, message and is sent to Firehose using Client.PutRecord API. As soon as the tweet reaches Firehose stream Lambda function is invoked. Lambda function decodes the encoded data, extracts the message and calls Amazon Comprehend API to detect the sentiment of the tweet.

Step 3. Sentiment Analysis: Amazon Comprehend with the help of powerful Machine Learning capabilities deduces the sentiment of the tweet as "POSITIVE", "NEGATIVE" and "NEUTRAL". Along with sentiment we also record the sentiment score which is average of Confidence of Positive and Negative sentiment of the tweet. This will help us to classify neutral tweets and slightly polar tweets. Data is encoded and sent to Kinesis Firehose.

Step 4. Data Storage and Visualization: Kinesis delivers the records to elastic search. An "Index" is created for storing the result and a "Type" is defined for saving the output record. Kibana, an open source data visualization plugin for Elasticsearch is used to visualize the data in various form of graphs.

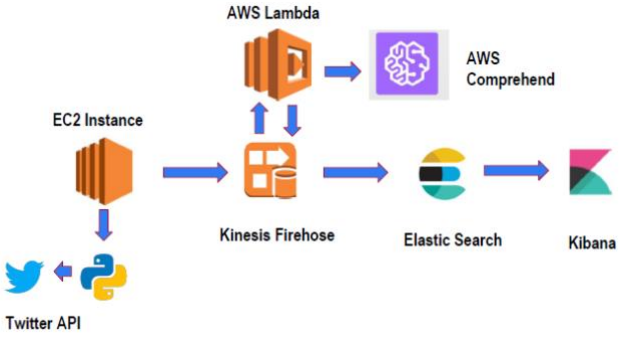

Figure 1: AWS Setup for Sentiment Analysis

```
for record in event['records']:

    dict_data = json.loads(base64.b64decode(record['data']).decode('utf-8').strip())


    comprehend = boto3.client(service_name='comprehend', region_name='us-east-1')
    sentiment_all = comprehend.detect_sentiment(Text=dict_data["text"], LanguageCode='en')
    sentiment = sentiment_all['Sentiment']
    positive = sentiment_all['SentimentScore']['Positive']
    negative = sentiment_all['SentimentScore']['Negative']
    total = (positive + negative)/2

    output_record = {
        'recordId': record['recordId'],
        'result': 'Ok',
        'data': base64.b64encode(json.dumps(data_record).encode('utf-8')).decode('utf-8')
    }


    output.append(output_record)
return {'records': output}
```

Figure 2:Lambda function invoking AWS Comprehend

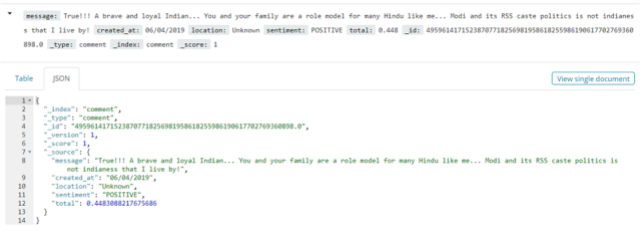

Figure 3:JSON response in Elasticsearch

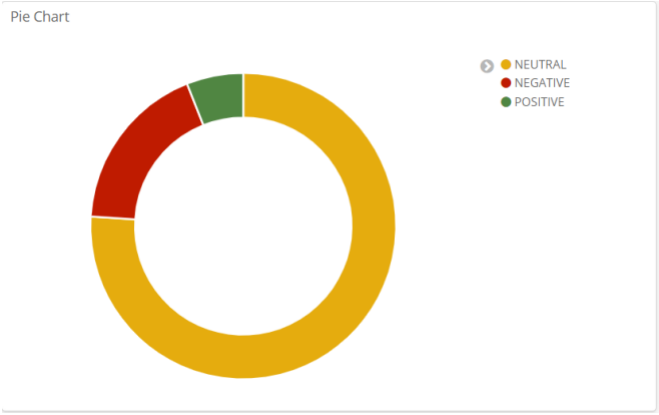

Figure 4:Classifying sentiment Score

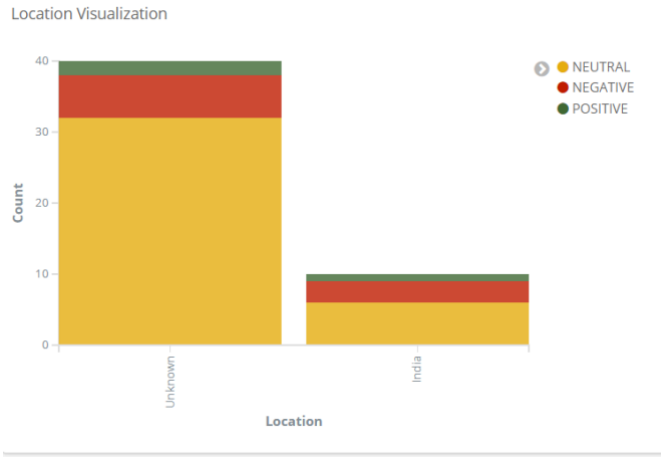

Figure 5:Scores classified by location

B. Solution Using Google Cloud Platform

Step 1. Data Collection and the Instance: Using Twitter Developer Portal, create Access token and access token secret. The Instance is setup with custom configuration to python script. Python3 is installed on the instance using pip command. Python Script containing the Twitter API Tokens runs on the Google Cloud Instance (Compute Engine). The real-time tweets are streamed and English language tweets with a filter condition are captured using the python script.

Step 2. Data Ingestion: Cloud Pub/Sub provides staging location for data based on event (tweets streaming) on its journey towards storage as logs. A 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. A topic "filtertweets" is created in Pub/Sub. The tweets are published to this topic based on the event of their arrival. A subscription "sentiment_cloudfunction-filtertweets" is created where the cloud function receives this data.

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 tweets streamed to Pub/Sub. Cloud function invokes Cloud Natural Language API where staged data is converted to a structured json format with fields "tweet", "score", "magnitude". The score can range from -1 to +1, but the user has the flexibility to define a custom range. Google defines 0.25 to 1.0 as Positive, -0.25 - +0.25 as neutral, and -1 - -0.25 as negative, however the user can achieve more granularity by classifying the tweets as Highly Positive, Positive, Neutral, Negative and Highly Negative.

Step 4. Storage and Visualization: Logstash is server- side data processing pipeline subscribed to Pub/Sub. Logstash Ingests logging data from Pub/Sub and transforms it and sends the data to Elasticsearch. Kibana could be used to visualize the data present in Elasticsearch. However, the nested json response from Google cloud platform's logging service was incompatible with Kibana's indexing mechanism. Google Cloud Platform's Stackdriver Monitoring provides visibility into the performance, uptime, and overall health of cloud-powered applications.

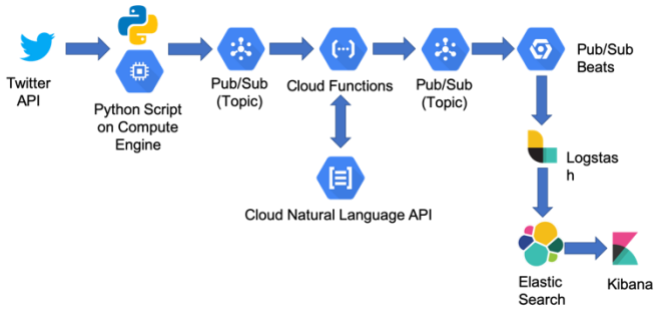

Figure 6: Google Cloud Setup for Sentiment Analysis

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

Figure 7: Publish tweets to pub/sub

```
"""Run a sentiment analysis request on text within a passed filename."""
client = language.LanguageServiceClient()
content = tweet
document = types.Document(
    content=content,
    type=enums.Document.Type.PLAIN_TEXT)
annotations = client.analyze_sentiment(document=document)

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

Figure 8: Cloud function invoking Cloud Natural Language API

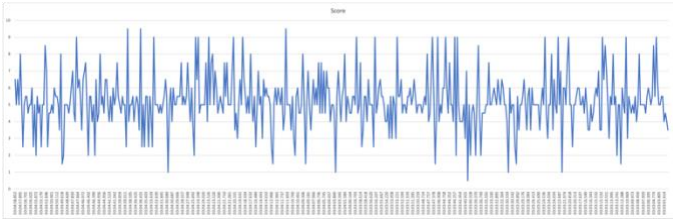

Figure 9: Score variation over time

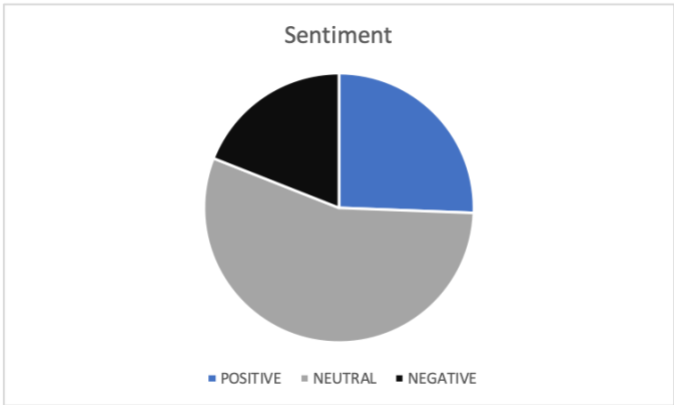

Figure 10: Classifying sentiment score

Figure 11: JSON response in logs viewer

V. EVALUATION

A. Accuracy of AWS vs GCP

Sentiment analysis training datasets are available online for testing purposes. The chunk of text in these datasets have been pre-categorized and verified by a human. We tested accuracy for Google NLP and AWS comprehend services against these pre-existing models.

Twitter US Airline Sentiment:
Twitter US Airline Sentiment by Kaggle had tweets of customers expressing their views regarding 6 popular US airlines on twitter. The tweets were highly unstructured meaning grammar, spelling and diction were highly colloquial and therefore difficult to analyze. Hence, the tweets were preprocessed and cleaned before being sent to Google and AWS API to receive back the results.

- The dataset had 14,640 tweets in total out of which 2,343 tweets were positive, 3,099 tweets were neutral, and 9,178 tweets were negative.
- Google NLP results: 9,039 tweets were matched correctly whereas 5,601 tweets failed to match. Google NLP had 62% match rate.
- AWS Comprehend results: 8,781 tweets were matched correctly whereas 5,859 tweets failed to match. AWS Comprehend had 60% match rate.
- Conclusion: With respect to the above highly unstructured dataset, Google did slightly better than AWS. We further considered only the neutral tweets and found that the failure rate for these tweets were higher in case of Google NLP with failure rate of 28% when compared to AWS comprehend which had a failure rate of just 18%.

Yelp Restaurant Sentiment:
Yelp dataset had about 1,000 reviews by yelp customers which were highly polar in nature. Out of the 1,000 reviews, 500 were clearly positive and 500 were clearly negative.

- Google NLP results: 931 tweets were matched correctly whereas 69 tweets failed to match. Google NLP had a good match rate of 93%.
- AWS Comprehend results: 784 tweets were matched correctly whereas 216 tweets failed to match. AWS Comprehend had a match rate of 78%.
- Conclusion: With respect to the above structured dataset where the reviews had highly polarity, Google did better than AWS.

Apple Product Sentiment:
Apple dataset had tweets of customers who expressed their opinion regarding Apple product and services. We found majority of the tweets in this dataset to be less polar in nature/neutral and sarcastic statements.
The dataset had 9,092 tweets in total – 570 negative statements, 5,544 neutral statements and 2,978 positive statements.

- Google NLP results: 4,012 tweets were matched correctly whereas 5,081 tweets failed to match. Google NLP had 56% match rate.
- AWS Comprehend results: 3,322 tweets were matched correctly whereas 5,771 tweets failed to match. AWS Comprehend had 63% match rate.
- Conclusion: With respect to the above dataset, AWS did better than Google.

B. Performance

US Twitter Airline sentiment dataset had customer tweets related to 6 most popular airlines of US. To compare performance, we recorded the time required to get back the sentiment analysis results of all the tweets under each airline category using both Google and AWS API.

We found that in every case, AWS comprehend is clearly faster than Google NLP.

Attempt 1		GOOGLE			AWS	
	Start time	End time	Total	Start time	End time	Total
American	15:58:19	15:58:59	40sec	13:44:25	13:44:59	34sec
Delta	15:37:35	15:38:15	40sec	14:02:59	14:03:33	34sec
Southwest	15:44:33	15:45:15	42sec	14:05:24	14:05:59	35sec
United	18:04:49	18:05:34	45sec	14:07:24	14:08:01	37sec
US Airways	15:59:35	16:00:25	50sec	14:09:26	14:10:00	34sec
Virgin Americ	18:03:55	18:04:33	38sec	13:41:49	13:42:22	33sec
Attempt 2						
American	16:00:41	16:01:20	39sec	13:50:12	13:50:45	33sec
Delta	15:38:38	15:39:19	41sec	13:51:08	13:15:42	34sec
Southwest	15:45:56	15:46:39	43sec	13:53:40	13:54:14	34sec
United	18:09:33	18:10:17	44sec	13:54:25	13:55:01	36sec
US Airways	16:01:59	16:02:48	49sec	14:48:54	14:49:27	33sec
Virgin Americ	18:06:45	18:07:22	37sec	14:57:18	14:57:49	32sec

Table 1:Runtime analysis for airline dataset

C. Overall Result

We conclude that each service may perform better or worse depending on the type of text to be analyzed. Hence our suggestion would be to carefully consider the goal and nature of your dataset before deciding which service to use.

1. If the dataset has more statements which are inclined towards positive or negative sentiment meaning highly polar in nature, Google would be a better choice.
2. If the dataset has more statements are less polar in nature or neutral statements and sarcastic statements which are difficult to analyze, AWS does better.
3. If you are looking for faster execution and performance, AWS is clearly a better choice.

VI. CONCLUSION

In the project, the team designed a Saas Model capable of performing sentiment analysis of real-time data on both AWS and GCP. Using labeled datasets, AWS Comprehend, and Cloud Natural Language API were compared on the following metrics.

Speed: For a SaaS model performing sentiment analysis the any platform chosen has certain tradeoffs. To achieve a system with high speed, Amazon Web Services clearly performs better.

Accuracy and Granularity: However, if the user desires accuracy while classifying sentiments, Google Cloud Platform's Cloud Natural Language API is more accurate than AWS Comprehend. Cloud Natural Language API's response is a json object with tweet, score and magnitude as its fields. The scores by default range between -1 to +1 but the user can define the values of the score in the cloud function. Furthermore, the user can also make the classification granular using these scores (Highly Positive, Positive, Neutral, Negative and Highly Negative).

Compatibility with 3rd Party Apps: Often the user would like to integrate the service with third party apps 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, AWS is a clear winner as GCP does not provide a lot of options to store the logs in a stash.

Visualization: Ready 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 result from GCP's logging system, makes visualization with Kibana a tedious process. AWS thus seems a clear winner.

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 to resolve patching issues, repairing the software and updating the hardware and software without the overhead of rebooting.

Price: Amazon Comprehend requests for sentimental analysis is measured in units of 100 characters, with a 3-unit minimum charge per request.

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

Table 2: Price per unit for Sentiment Analysis in AWS

The usage of the Google Natural Language API 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.

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

Table 3:Price per unit for sentiment analysis in GCP

Case1: 10,000 customer comments with 300 characters.
Case2: 10,000 customer comments with 10000 characters.

	Requests	Characters	AWS Units	Units for 10000 Requests	AWS $0.0001 per unit
Case 1	10000	300	3	30000	$3
Case 2	10000	10000	100	1000000	$100

Table 4: Price comparison for AWS

	Requests	Characters	Google Cloud Units	Units for 10000 Requests	Google Cloud 1$ per 1000 Units
Case 1	10000	300	1	10000-5000(FREE)	$5
Case 2	10000	10000	10	100000-5000(FREE)	$95

Table 5: Price comparison for GCP

Conclusion: If there is a larger number of requests for small size of data AWS costs lesser. But if there are fewer requests having huge data then Google NLP will cost lesser.

VII. DISCUSSION AND RELATED WORK

In [1], the authors have proposed an open source sentiment analysis tool on cloud to calculate the polarity by extracting distinct subjective features. The entities are classified into positive, negative and neutral. The tool provides a platform which ingests data from multiple sources like social media, news, e-commerce, stock market, product review and feedback from online blogs. The authors have leveraged cloud services to make the tool scalable, available and secure. In [1], the authors have designed their tool by mapping various APIs to their advantages and provided it as a tool on the cloud.

In [2] the authors have proposed a Real-time Information Visualization and Analysis platform. The proposed system performs data analysis on real-time data to classify them through visualization. The hashtags from several social media platform is streamed via Spark and the full text is ingested to perform sentiment analysis on Spark Engine. The proposed system leverages SQL to visualize the score obtained.

The authors of paper [3] focus on sentiment analysis with emphasis on polarity classification and agreement classification which assigns degree of positivity to the polarity. The concept

of sentiment analysis is applied to a document by filtering the irrelevant topics which might skew the polarity of the global sentiment. This proposal tries to identify the topic of interest and separate opinions associated with each topic. Using keywords is a naïve manner of performing sentiment analysis. A concept-based approach focuses on implicit meaning of the statement with natural language applications. Concept based approaches can identify subtly expressed sentiments which is superior to using purely syntactical techniques. Concept-based approaches can analyze multi-word expressions that don't explicitly convey emotion but are related to concepts that do.

The proposed work presented in [4] evaluates sentiment analysis of twitter data using StandfordNLP and twitter4j Libraries. It is implemented as a SaaS model and comes with built in scalability and reliance. The NLP program returns a value that estimates how much a tweet was positive or negative. The work describes the procedure to obtain tweets from Twitter API using OAuth.

Although most of these papers perform sentiment analysis in a similar manner, there are subtle differences in their object of focus and implementation. Few papers focus on the providing a reliable and scalable system that can perform sentiment analysis in real-time. Other papers focus more on the analysis itself, providing insight into how a score is determined from a sentence. Our paper tries to strike a balance albeit inclined towards the former goal.

REFERENCES

[1] Omkar Sunil Joshi, Garry Simon, "Sentimen Analysis Tool on Cloud: Software as a Service Model", ICACCT, 2018

[2] Yong- Ting Wu, He-Yen Hsieh, Xanno K. Sigalingging, Kuan- Wu – Su, Jenq – Shiou Leu,"RIVA: A Real-time Information Visualization and Analysis platform for Social Media Sentiment Trend", ICUMT, 2017

[3] Erik Cambria, Bjorn Shuller, Yunqing Xia, Catherine Habasi, "New Avenues in Opinion Mining and Sentiment Analysis", IEEE Intelligent Systems, 2013

[4] Hase Sudeep Kisan, Hase Anand Kishan, Aher Priyanka Suresh, "Collective Intelligence and Sentiment Analysis of Twitter Data by using StanfordNLP Libraries with Software as a Service (SaaS)", ICCIC, 2016

[5] J. Guerrero, J. Olivas, F. Romero, E. Viedma, "Sentiment Analysis: A Review and Comparitive Analysis of Web Services", Information Sciences, 2015

[6] Desheng Dash Wu, Lijuan Zheng, David L. Olson, "A Decision Support Approach for Online Stock Forum Sentiment Analysis" IEEE Transactions on Systems, Man and Cybernetics, 2014

[7] Rui Xia, Feng Xu, Chengqing Zong, Qianmu Li, Yong Qi, Tao Li, "Dual Sentiment Analysis: Considering Two Sides of One Review". IEEE Transactions on Knowledge and Data Engineering, 2015.