Paper AS08

Automated Social Media Monitoring for Pharmacovigilance using Cloud Solutions

Rohit Banga, Bayer Inc., Toronto, Canada

ABSTRACT

Social media platforms and public health forums provide important avenues for patients to discuss their health, use of medicinal products and any side effects. These avenues can provide important data points for pharmaceutical companies for safety monitoring. However, trying to manually find safety signals in vast unstructured data presents logistical challenges.

We propose a novel approach for automated monitoring of social media for safety signals using cloud solutions.

In this paper we will implement a Safety Tracking Tool for real time ingestion of data from Twitter. We will describe access patterns to ingest social media data in streaming/batch mode, extract relevant medical entities from unstructured data using a machine learning algorithm, and an automated software driven workflow that performs tasks such as sending notifications, automated reporting, and dashboard visualizations.

With the advancement in Artificial Intelligence (AI) and Machine Learning (ML) technologies and the advent of cloud solutions, we believe that there are opportunities for developing applications that ingest, comprehend and react to real-time events.

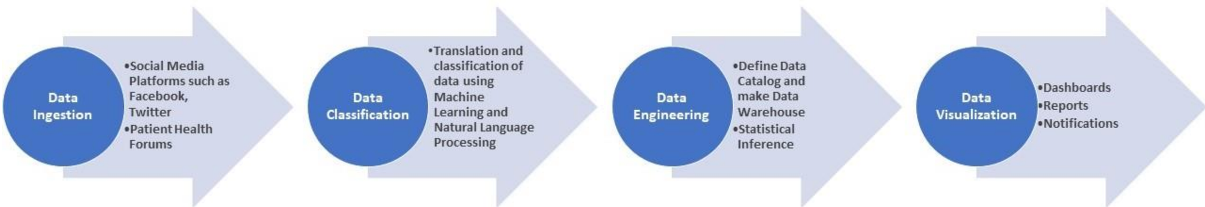

Figure 1. Process Diagram

INTRODUCTION

The World Health Organization (WHO) defines Pharmacovigilance (PV) as the science and activities relating to the detection, assessment, understanding and prevention of adverse effects or any other drug-related problem. Pharmacovigilance and Medical Device vigilance laws and regulations require Pharmaceutical companies to collect, analyze and report any suspected adverse event and/or quality issues that come to their knowledge about any products for human use (medicinal products, food supplements, cosmetics and medical devices). Pharma companies must maintain safety reports and react appropriately to quality and usability issues of their products. Pharmaceutical company personnel must also forward such safety violation or quality issue while performing job related tasks to their pharmacovigilance or quality assurance department immediately.

Social media is a promising source for new safety data and potential emergent safety signals. Social Media data has a distinct advantage over data collected using traditional AE reporting tools where under-reporting is a known limitation. Social Media data is closer to real-time occurrence of the event and it arises from direct user experience and can add to the information received from traditional post-marketing reporting methods. Many pharmaceutical companies now have an active social media strategy that proactively solicits/captures Adverse Event (AE) data on social media platforms. Companies use multiple AE reporting channels such as safety hotline, company websites and email correspondences to capture patient and consumer AEs.

Pharmaceutical companies operating in the social media space have a responsibility to document and follow-up on any potential adverse event reports communicated through such channels in compliance with the applicable regulatory guidance. However, it is important to understand that this data is essentially unstructured and obtained via uncontrolled and ungoverned processes in a non-regulated environment and is neither driven by data quality standards nor by specific business area orientation. Therefore, it is vital to carefully verify safety data obtained via social media for confirmation of the "identifiability" of both reporter and patient and address related data privacy issues and verify the accuracy of reported safety data.

This paper will describe a software solution to monitor a social media channel - Twitter in real time. It will describe a Safety Tracking Tool that will automatically ingest all tweets related to a pharmaceutical drug, extract relevant medical entities from the unstructured text of the tweet, and present the results in an interactive dashboard.

MONITORING TWITTER FOR PHARMACOVIGILANCE

The entire solution is implemented on AWS cloud and derives directly from an excellent article published on AWS Machine Learning Blog – *"Build a social media dashboard using machine learning and BI services"* by Ben Snively and Viral Desai (https://aws.amazon.com/blogs/machine-learning/build-a-social-media-dashboard-using-machine-learning-and-bi-services/). The code for implementing the solution can also be found in the above-mentioned article. Our aim with this paper is to not describe AWS technologies in detail but to educate the reader about different steps that are needed to achieve our goal and the mention the different AWS technologies that help support it.

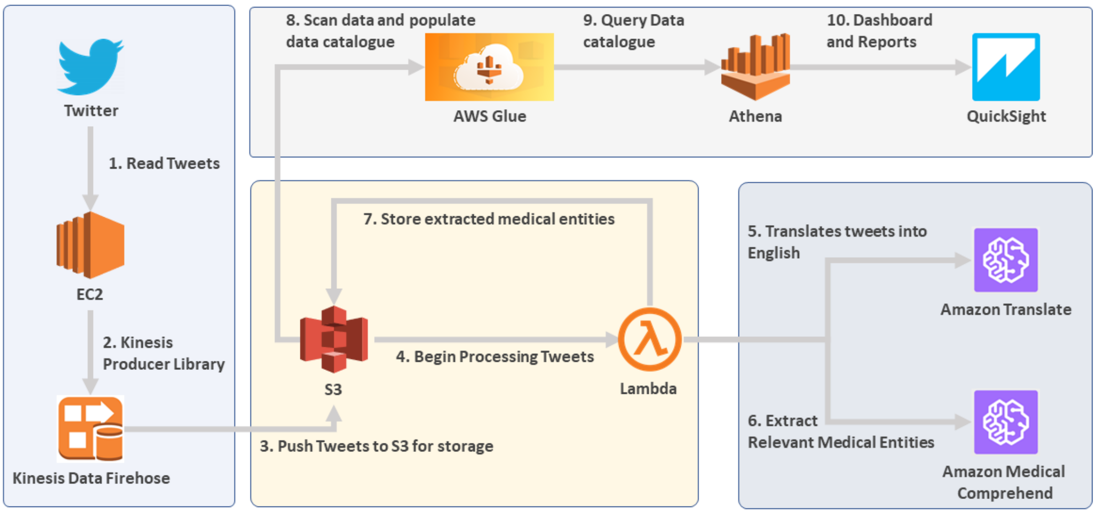

FIGURE 2 – Architecture Diagram

DATA INGESTION

Step 1 – Get Twitter API Credentials

The first step to make our Safety Tracker Solution is to get credentials from twitter to read the tweets through an API. At the time of writing this paper, one must apply for access for Twitter Streaming API at https://developer.twitter.com/.

To get access, you must tell twitter about the use case for your app. Twitter will then evaluate your use case and would give access accordingly. Once you get access, you will get a set of consumer API keys and access tokens that will be used to authorize your connection to twitter API.

Step 2- Use Twitter Streaming API to get relevant tweets

The next step is to create a twitter stream producer. This will essentially be a NodeJS application running on a Linux machine. In our example, the NodeJS application runs on an ubuntu machine hosted on Amazon EC2 (Elastic Compute Cloud). Amazon EC2 provides on-demand compute capacity in the cloud. The NodeJS application uses the credentials obtained above to connect to twitter streaming API to collect tweets that matches specific criteria (such as drug names or medicinal product names) and pushes those tweets to Kinesis Firehose.

This is the main crux of the NodeJS Application

```
var T = new Twit(twitter_config.twitter) //Make a Twit object using the API credentials

var stream = T.stream('statuses/filter', { track: twitter_config.topics , language: twitter_config.languages }); //Make a Streaming object that filters the tweets according to topics and languages that we specify

var recordParams = {};
  stream.on('tweet', function (tweet) {
                var tweetString = JSON.stringify(tweet)
        recordParams = {
          DeliveryStreamName: twitter_config.kinesis_delivery,
          Record: {
            Data: tweetString +'\n'
          }};
        firehose.putRecord(recordParams, function(err, data) {
         if (err) {…}});
        });
} //Turn the stream on and put the tweets in Kinesis Firehose using putRecord function.
```

Step 3 – Use Amazon Kinesis Firehose to push tweets to a storage service

The next step is to capture the tweets produced by the streaming API. We will use Amazon Kinesis Data Firehose for this purpose. Kinesis firehose is amazon's service to prepare and load real-time data streams into data stores and analytics tools. This service captures the tweets pushed from the Twitter Stream Producer in the previous step (using putRecord function) and further pushes the data into Amazon S3 in near real-time for storage.

Step 4 – Create S3 Bucket to store tweets

We now need to store the tweets pushed from Kinesis Firehose in the previous step. We will use an Amazon S3 (Amazon Simple Storage Service) bucket that will store the tweets as file objects produced kinesis firehose streams. S3 is an object storage service that has unlimited capacity and provides excellent security and availability of data. The main benefit of storing data in S3 buckets is that we can configure Lambda functions (talked about in the next step) that get triggered based on events in S3 buckets.

DATA CLASSIFICATION

Step 5 - Translate tweets into English

We gathered tweets not only in English but also in Spanish, German, French, Arabic and Portuguese language. We will use Amazon Translate - a neural machine translation service – from Amazon that uses deep learning models to deliver accurate and natural translation of languages. This service can be called through its API and uses the "translateText" function. The below mentioned is a snippet of the code that uses the API to pass tweet text and get translated text in return.

```
const translate = new aws.Translate();
var translateParams = {
                    SourceLanguageCode: tweet.lang,
                    TargetLanguageCode: 'en',
                    Text: tweet.text
                  };
```

```
translate.translateText(translateParams, function(err, data) {
  if (err) {…
  } else {
    tweet.originalText = tweet.text;
    tweet.text = data.TranslatedText;
    ….
  } });
```

This code needs to be called only when new tweets arrive in the S3 bucket. For this purpose, we will provision this code on an event-driven AWS Lambda Function. AWS Lambda is an amazon's revolutionary service that lets you run code without provisioning or managing servers. Lambda function will get triggered when new tweets arrive in S3, it will then call the Translate API and then store the translated text as file objects back to S3.

Step 6 – Use Natural Language Processing (NLP) to extract clinically relevant entities from tweets

This is perhaps the most important part of the entire process. We need to extract meaningful information from unstructured tweet data, and we need to do that real-time and at scale for our Safety Tracker to work in production. For this purpose, we will use Amazon Comprehend/Medical Comprehend. Comprehend is a natural language processing (NLP) service that uses machine learning to find insights and relationships in text. We can use custom classification models and plug them in Amazon Comprehend or use Amazon Medical Comprehend to extract clinically relevant information from the tweets.

The below mentioned is a snippet of the code that uses Amazon Comprehend Medical function - detectEntitiesV2 to extract clinically relevant entities.

```
const comprehendmedical = new aws.ComprehendMedical();
var params = {Text: tweet.text
                    };
comprehendmedical.detectEntitiesV2(params, function(err, data) {
                                if (err) {
                                  …
                                } else{
                                  data.Entities.forEach(function(entity){
                                      var entityRecord = {
                                        DeliveryStreamName: entityStream,
                                        Record: {
                                            Data: JSON.stringify({
                                                tweetid: tweet.id, /* required */
                                                text: tweet.text,
                                                originalText: tweet.originalText,
                                                entityID: entity.Id,
                                                entityScore: entity.Score,
                                                entityText: entity.Text,
                                                entityCategory: entity.Category,
                                                entityType: entity.Type,
                                                entityTrait: entity.Traits,
                                                entityAttributes: entity.Attributes
                                              }) + '\n'
                                          }//end of record
                                      };//end of params
                                      firehose.putRecord(entityRecord, function(err, data) {
                                        if (err) console.log(err, err.stack);
                                      });
                                  });
                                  entityCallback(null);
```

```
    }
});
```

This code is also provisioned on AWS Lambda just like Amazon Translate in the previous step. AWS Lambda function will get triggered every time new tweets arrive in S3, will get translated into English and then relevant clinical entities extracted as a JSON file and stored back to S3.

DATA ENGINEERING

Step 7 – Build Data Warehouse and a Data Catalog

Our next step is to extract, transform and load data from S3 and build a data warehouse. For this, we will use AWS Glue, that is an ETL service that can crawl S3 regularly and create/update tables in a Data Catalog. AWS Glue automatically discovers data structures and prepares data catalogs. AWS Glue would automatically discover the data structure of tweets, translated text and clinically relevant entities in our S3 bucket.

Step 8 – Query S3 data and populate data warehouse using Athena

Amazon Athena is used to query Amazon S3 data using the data catalog created by AWS Glue. Athena can be also be used to query the S3 directly (i.e. without the help of data catalogs created by AWS Glue), but if there are multiple data sources that create and dump data in S3, then AWS Glue crawlers can automatically discover the format and type of data in Amazon S3 and create data catalogs that can be queried.

DATA VISUALIZATION

Step 9 – Visualize data using Amazon Quicksight Dashboards

The last step is to visualize the data created by Athena tables in the previous step. We will use Amazon QuickSight for this purpose which would enable us to build interactive dashboards and reports and would connect seamlessly with the data warehouse. Custom solutions using R Shiny (particularly for geospatial data analysis) or Python Flask can also be used to for visualization if required.

One can also configure notifications using Amazon SNS (Simple Notification Service). For example, an email or SMS can be sent to a monitor every time an AE resulting in death is tweeted.

CONCLUSION

This paper outlines the steps that utilizes the tremendous power of cloud to build modular and scalable solutions without the need of committing to physical infrastructure. A big

advantage of using cloud solutions is that the components can be retired or replaced and custom solutions can be built over a period. It also makes advanced technologies available and pluggable with the help of APIs. Such systems can ingest and react to data in real-time and this capability opens up exciting avenues. For example, we can build a bot that can guide a user to a useful pharma drug or product if they tweet about a specific Adverse Event. We can also utilize innovative clinical trial designs that can incorporate incoming streaming data from fitness trackers for statistical analysis or gauge public sentiment related to launch of a new drug or medical device.

REFERENCES

1. Pharmacovigilance - https://www.who.int/medicines/areas/quality_safety/safety_efficacy/pharmvigi/en/
2. Build a social media dashboard using machine learning and BI services - https://aws.amazon.com/blogs/machine-learning/build-a-social-media-dashboard-using-machine-learning-and-bi-services/
3. Patient-Focused Drug Development: Methods to Identify What Is Important to Patients Guidance for Industry, Food and Drug Administration Staff, and Other Stakeholders - https://www.fda.gov/regulatory-information/search-fda-guidance-documents/patient-focused-drug-development-methods-identify-what-important-patients-guidance-industry-food-and
4. Sabrina Nour, Gilles Plourde, in Pharmacoepidemiology and Pharmacovigilance, 2019
5. Social media and pharmacovigilance: A review of the opportunities and challenges: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4594734/
6. Accessing social media information for pharmacovigilance: what are the ethical implications? https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6199687/
7. The Impact and Use of Social Media in Pharmacovigilance https://www.sciformix.com/safety-risk-management-blog/the-impact-and-use-of-social-media-in-pharmacovigilance/
8. AWS Technologies –
    a. Amazon EC2 - https://aws.amazon.com/ec2/
    b. Amazon S3 - https://aws.amazon.com/s3
    c. Amazon Kinesis Data Firehose - https://aws.amazon.com/kinesis/data-firehose/
    d. AWS Lambda - https://aws.amazon.com/lambda/
    e. Amazon Comprehend - https://aws.amazon.com/comprehend/
    f. Amazon Translate - https://aws.amazon.com/translate/
    g. AWS Glue - https://aws.amazon.com/glue/
    h. AWS Athena - https://aws.amazon.com/athena/
    i. Amazon Quicksight - https://aws.amazon.com/quicksight/
    j. Amazon SNS - https://aws.amazon.com/sns/