Documentation On

**“YouTube Data Integration and Category Prediction using NLP”**

2023

Made By:

Ritik Arora

Linkedin- https://www.linkedin.com/in/ritik-arora-001/

# Abstract

The exponential surge in online video content has ignited the necessity for robust frameworks that seamlessly manage, analyze, and extract valuable insights from vast volumes of structured and semi-structured YouTube video data. This research introduces a comprehensive project aimed at securely orchestrating the entire data lifecycle, from data ingestion to analytics and machine learning, by leveraging a suite of Amazon Web Services (AWS) cloud services. The integration of cloud-powered analytics and machine learning further advances the understanding.

In recent years, surge of online video content has driven the need for efficient and accurate video categorization techniques. This project relates to video categorization by leveraging the textual information contained in video titles. We propose a Recurrent Neural Network (RNN) architecture that effectively captures the sequential dependencies in video titles and maps them to corresponding video categories.

# Table of Contents

1. [**Table of Contents 1**](#_gjdgxs)
2. [**Abstract 2**](#_30j0zll)
3. [**AWS Architecture 3**](#_fcxqctj1y9lp)
4. [**Services Used: 4**](#_a3wf6x6hd9ze)
5. [**Dataset 5**](#_a8qi7pe05l85)
6. [**Identity Access Management 6**](#_cr1nu0ss7ro4)
7. [**Simple Storage Service S3: 7**](#_q136toskilmm)

7.1 [DataLake Creation 8](#_2lje8a5fntwm)

1. [**AWS Glue - Crawler 9**](#_5mr8sjz6fffc)
2. [**AWS Glue 11**](#_wiykbggzhe4l)
3. [**AWS Lambda 12**](#_4uv6iw87cs6u)
4. [**Lambda Trigger 12**](#_yyc44mi4bbs0)
5. [**AWS Step Functions 14**](#_oh4cthr829ua)
6. [**Parquet File format 14**](#_ppe9hvme8dp4)
7. [**AWS Quicksight 16**](#_od7aeakr9yyj)
8. [**Further Analysis: 17**](#_2c970xtjr976)
9. [**Natural Language Processing 18**](#_trxeo5ssz3nw)

16.1 [Recurrent Neural Network 19](#_sppfnwqdgd67)

16.2 [Keras 19](#_1pxezwc)

16.3 [Transfer Learning 21](#_49x2ik5)

16.4 [Softmax classifier 23](#_147n2zr)

16.5 [Pickle and HDF5 26](#_q3elsku927uk)

1. [**Code Snippet 28**](#_9k23t5kc7ovw)
2. [**References 36**](#_nmf14n)

**Table of Figures**

**FIG. NO.** **PAGE NO.**

Fig. 1 AWS Architecture 4

Fig. 2 IAM Role 7

Fig. 3 AWS CLI command for Json file 8

Fig. 4 AWS CLI command for CSV File 8

Fig. 5 AWS S3 Buckets 9

Fig. 6 AWS Glue Crawler 10

Fig. 7 AWS Glue Crawler with 10 partition regions 11

Fig. 8 Crawled Data Tables 11

Fig. 9 Lambda function 13

Fig. 10 Lambda Trigger 14

Fig. 11 Analytics S3 Bucket 16

Fig. 12 Parquet file format 17

Fig. 13 Bar Plot for Average likes by Title 18

Fig. 14 Donut chart of Sum of views by title 19

Fig. 15 Bar chart of Average of views by title 19

Fig. 16 Bar chart of average of comment count by title 19

Fig. 17 Traditional ML vs Transfer Learning 25

Fig. 18 Softmax Activation function 27

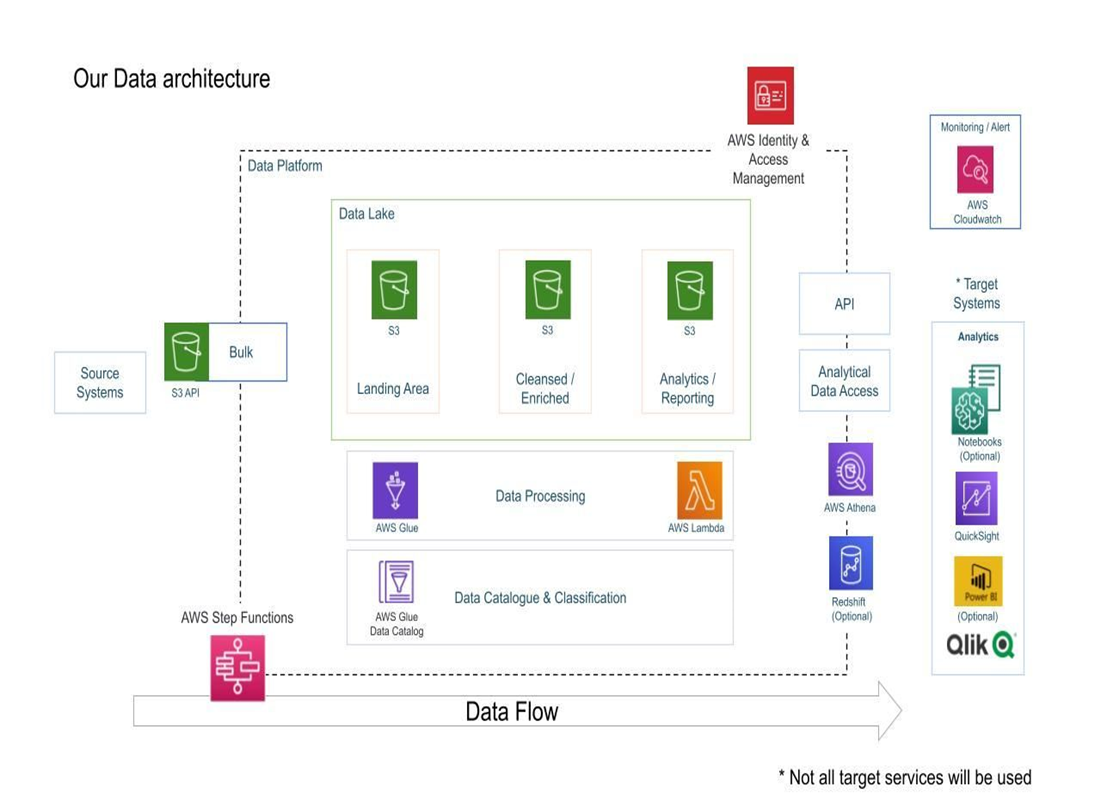
Fig. 19 graph of Softmax function 27

Fig. 20 Classification Report 36

Fig. 21 Dual line chart of Training and validation accuracy curve 37

Fig. 22 Training and Validation Loss 37

# AWS Architecture

Fig.1 Architecture

# Services Used:

1. Amazon S3: Amazon S3 is an object storage service that provides manufacturing scalability, data availability, security, and performance.

2. AWS IAM: This is nothing but identity and access management which enables us to manage access to AWS services and resources securely.

3. QuickSight: Amazon QuickSight is a scalable, serverless, embeddable, machine learning-powered business intelligence (BI) service built for the cloud.

4. AWS Glue: A serverless data integration service that makes it easy to discover, prepare, and combine data for analytics, machine learning, and application development.

5. AWS Lambda: Lambda is a computing service that allows programmers to run code without creating or managing servers.

6. AWS Athena: Athena is an interactive query service for S3 in which there is no need to load data it stays in S3.

7. AWS Sagemaker Studio Lab to run the deep learning algorithm

# Dataset

This dataset was fetched using the YouTube API and was uploaded statically to S3 bucket. The dataset includes several months (and counting) of data on daily trending YouTube videos . Data is included for the US, GB, DE, CA, and FR regions (USA, Great Britain, Germany, Canada, and France, respectively), with up to 200 listed trending videos per day. Data includes the video title, channel title, publish time, tags, views, likes and dislikes, description, and comment count columns. The category\_id field which varies between regions .

# Identity Access Management

A web service that helps to securely control access to AWS resources. Shared access to your AWS account. Granular permissions. Roles are created by AWS Management Console. Then Policies are attached to manage granular access. An IAM account is created with custom password and admin access in AWS console.

The access key and secret key are generated and the access key credentials are downloaded.

Then the AWS CLI is set up using the access key credentials and configured with the access key, secret key, and region. The configuration is verified by running a command in AWS CLI.

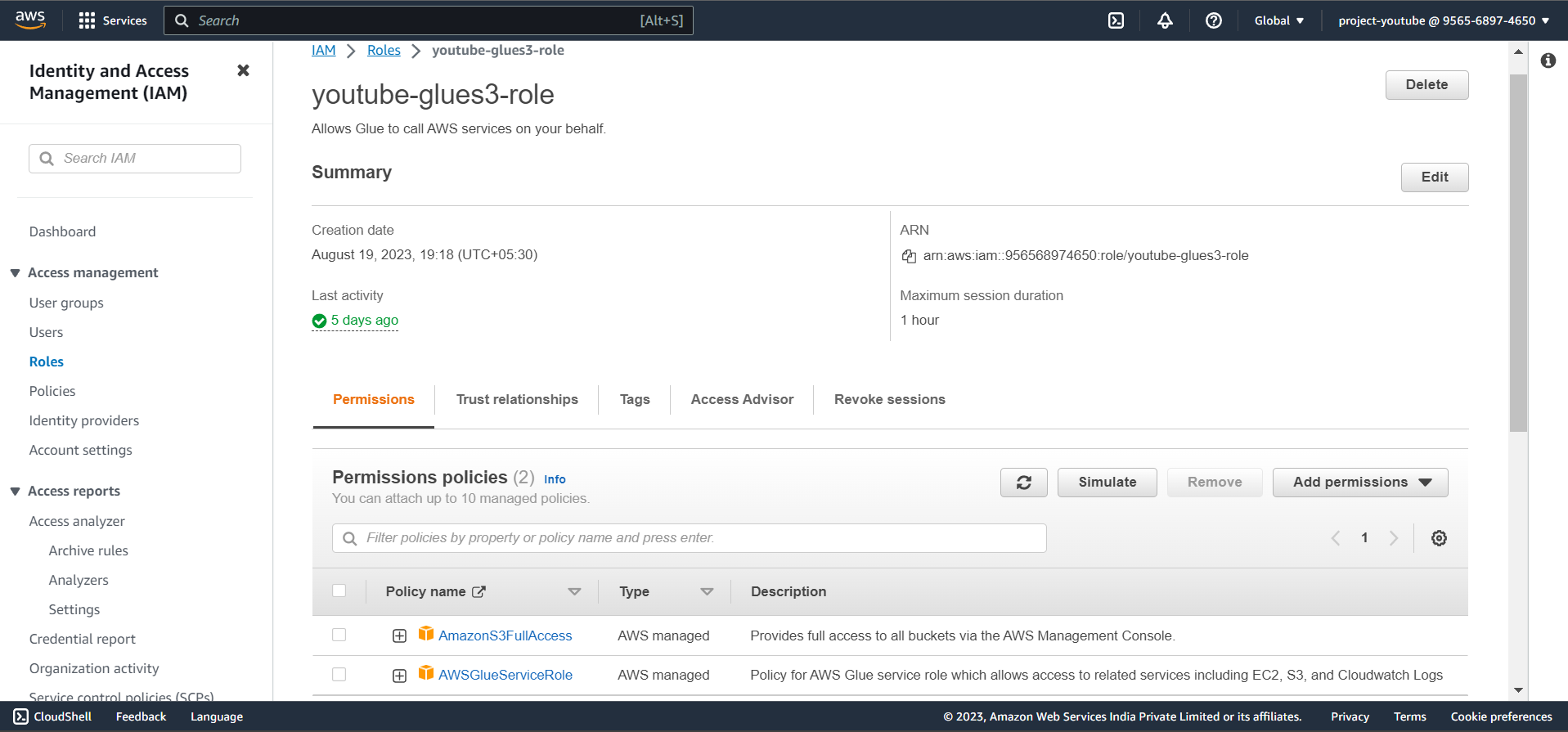


Fig.2- IAM role

# Simple Storage Service S3:

The object storage service to store and protect any amount of data.It includes use cases, such as data lakes, websites, big data analytics etc. S3 provides management features so that you can optimize, organize, and configure access to your data to meet your specific business, organizational, and compliance requirements. Seamlessly integrate and move data.

Upload data to S3 bucket using AWS CLI command.

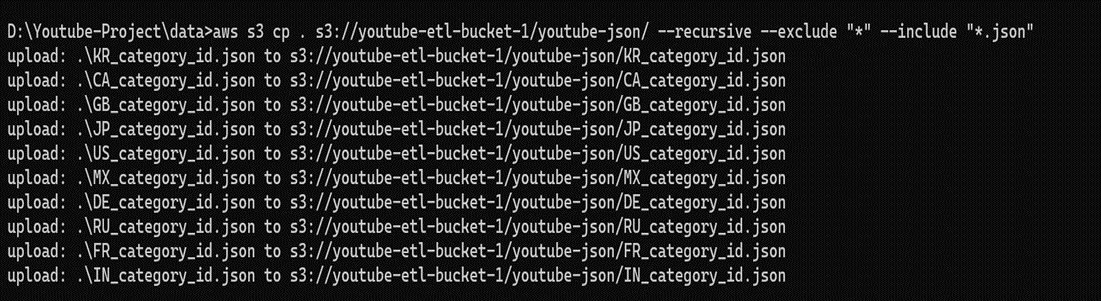


Fig.3- AWS CLI command for json

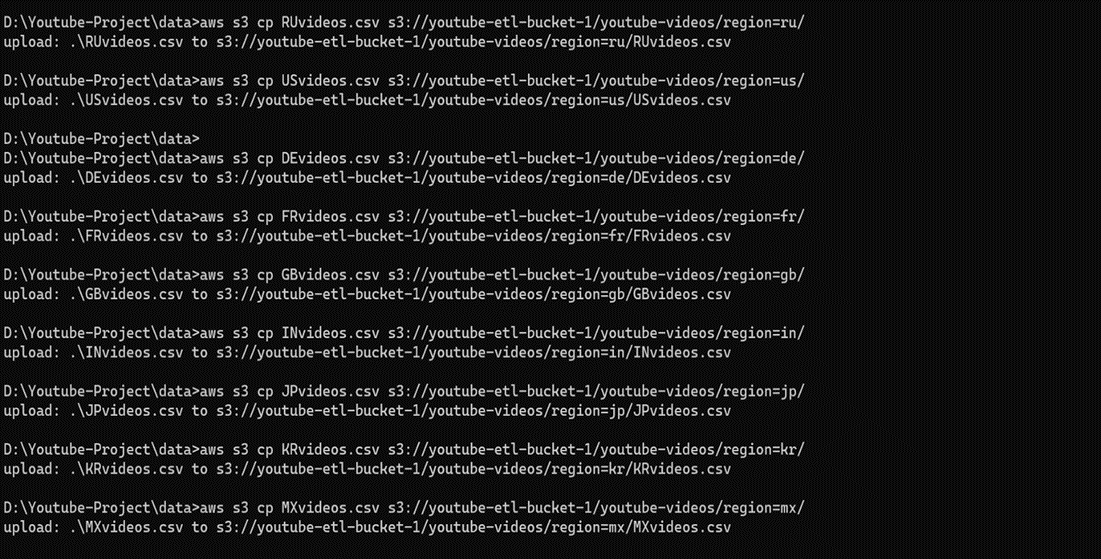


Fig. 4- AWS CLI command for csv file

## DataLake Creation

Then we created a DataLake of S3 bucket with specific naming convention and enabled server-side encryption for data protection. Upload data to S3 bucket using AWS CLI command.

Three S3 buckets for storing:

* Raw data- For storing raw .json and .csv files.
* Cleansed data- For storing cleansed .json and .csv file in parquet format.
* Cleansed data for analytics- for storing combined data to this bucket which will further used to give insights.

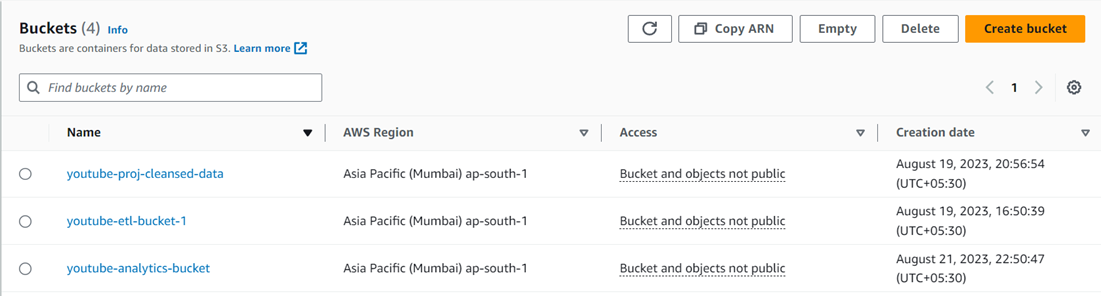


Fig.5- AWS S3 buckets

# AWS Glue - Crawler

A crawler is used to populate the AWS Glue Data Catalog with tables. Crawlers can crawl multiple data stores in a single run. ETL jobs that you define in AWS Glue use these Data Catalog tables as sources and targets. AWS Glue Data Catalog is a managed metadata repository that stores and organizes metadata. It used to define the structure and schema of your data during glue etl job .

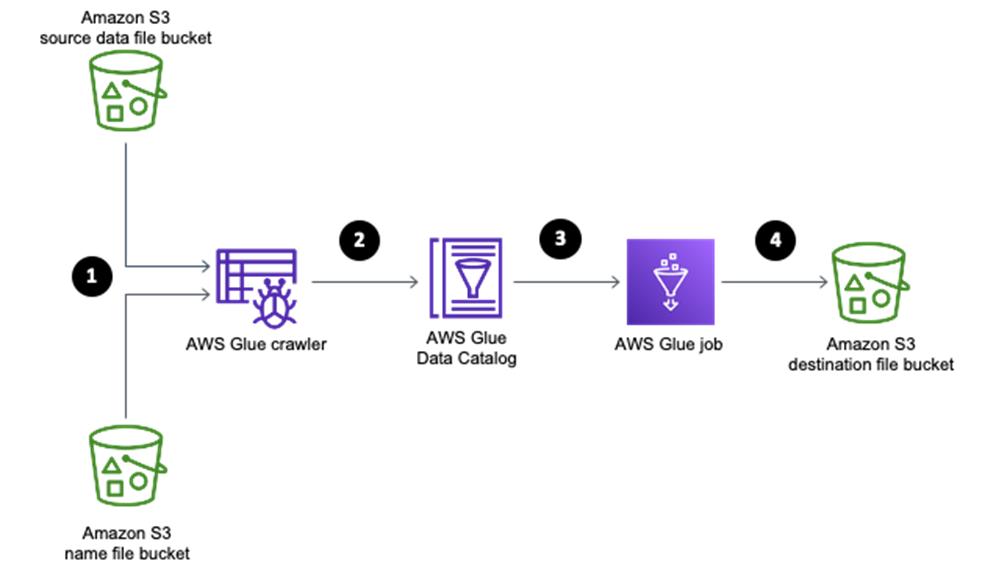


Fig.6- AWS Glue Crawler

Crawler run on S3 on csv data with 10 partition regions.

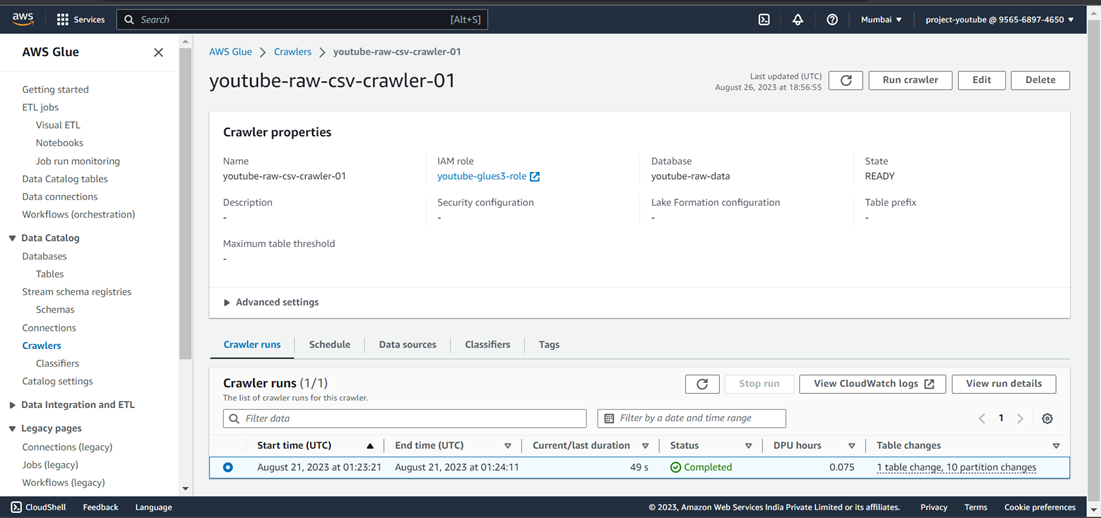


Fig. 7- Glue Crawler with 10 partition regions

Crawled Data

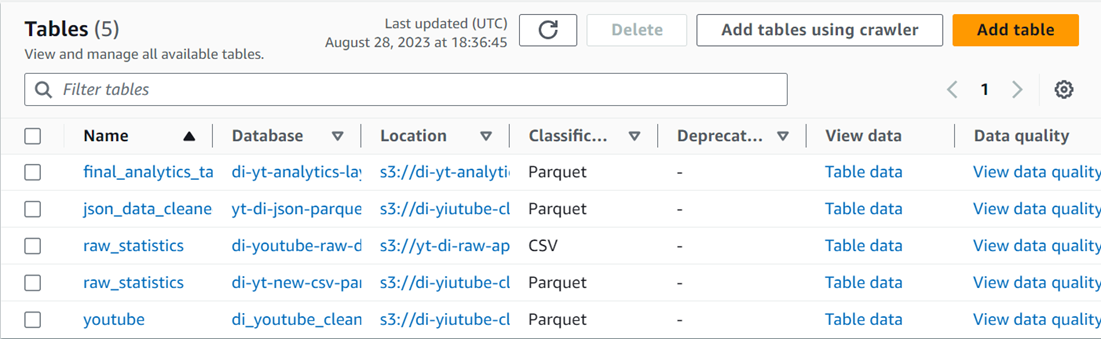


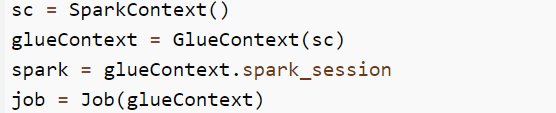
Fig. 8- Crawled Data Tables

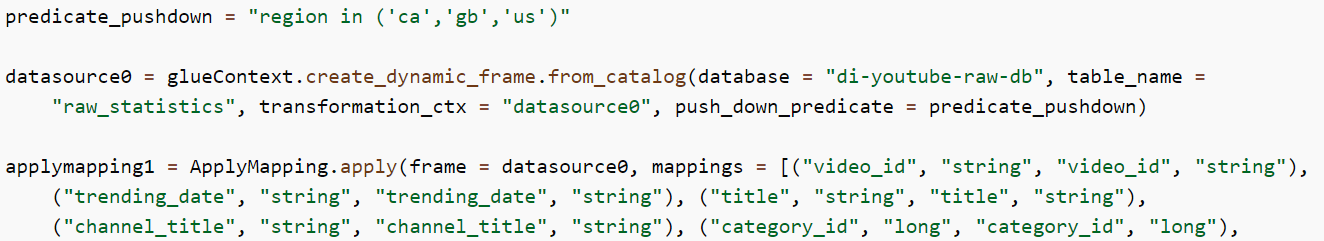
# AWS Glue

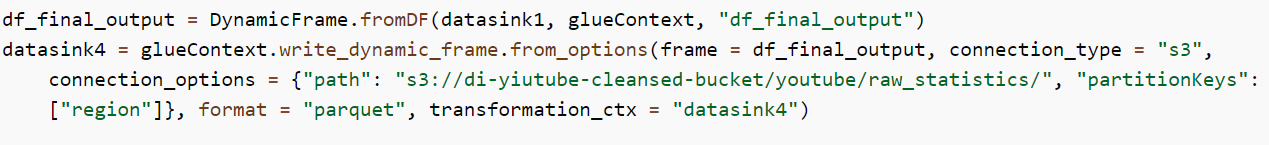
Glue is a serverless data integration service that makes it easier to discover, prepare, move, and integrate data from multiple sources. It simplifies ETL pipeline development. It supports various processing frameworks and workloads .

An ETL job was performed using AWS glue. The job extracted the raw .csv data from the s3 bucket through tables. Transformed the data by defining schema structure and predicate pushdown to select the region which filters the data before the source is extracted. It improves the ETL process and optimizes resources used. Then we loaded the transformed data to a new s3 bucket.

Following script was used to perform ETL process.







# AWS Lambda

Lambda is a serverless computing service provided by Amazon Web Services. Lambda functions can perform any kind of computing task, from serving web pages and processing streams of data. It runs code without provisioning or managing servers. Lambda creates functions, self-contained applications written in one of the supported languages and runtimes.

We created a lambda function to convert JSON data into parquet

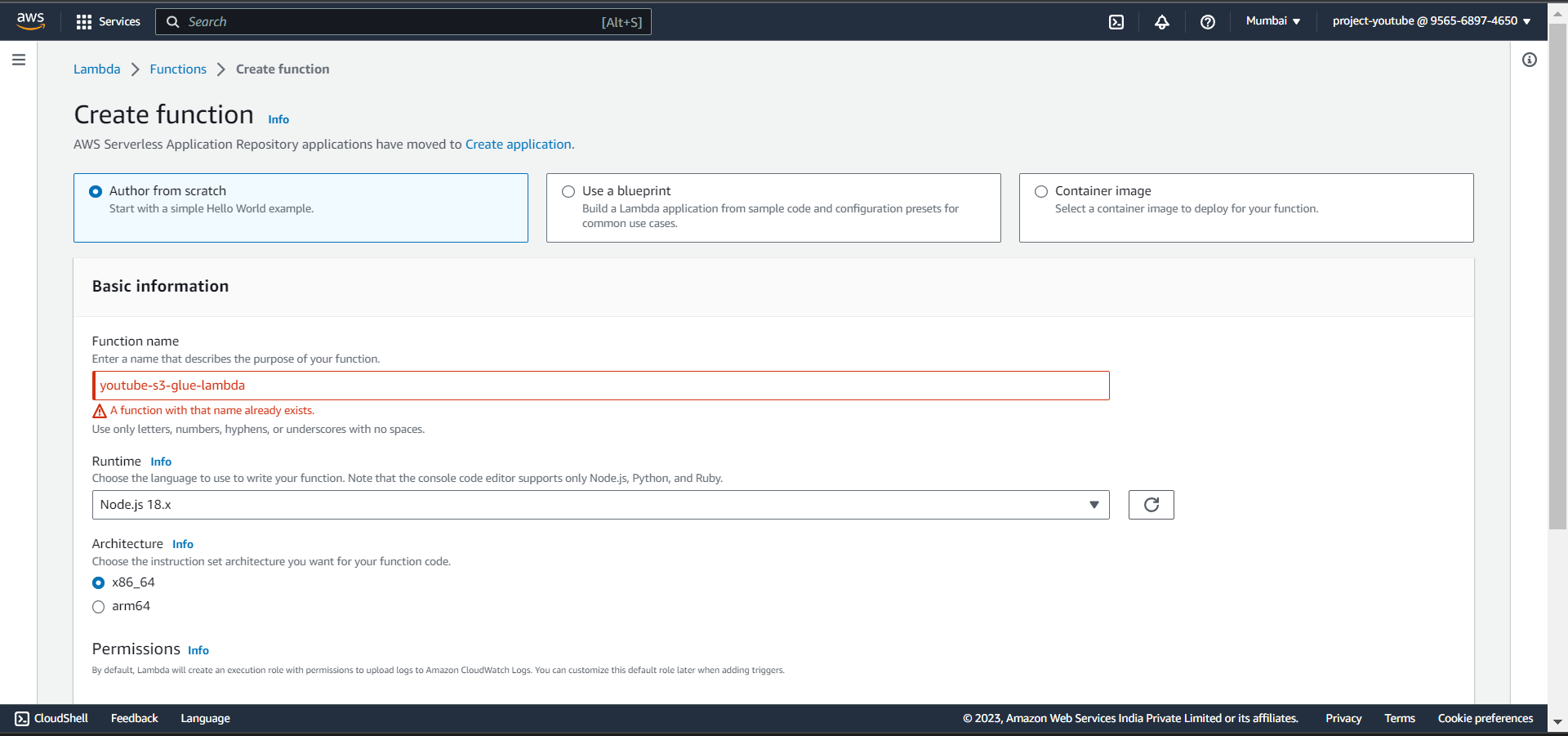


Fig. 9- Lambda function

# 

# Lambda Trigger

Lambda trigger was created to automate the ETL process which reads files from an S3 bucket whenever raw files are added to the bucket

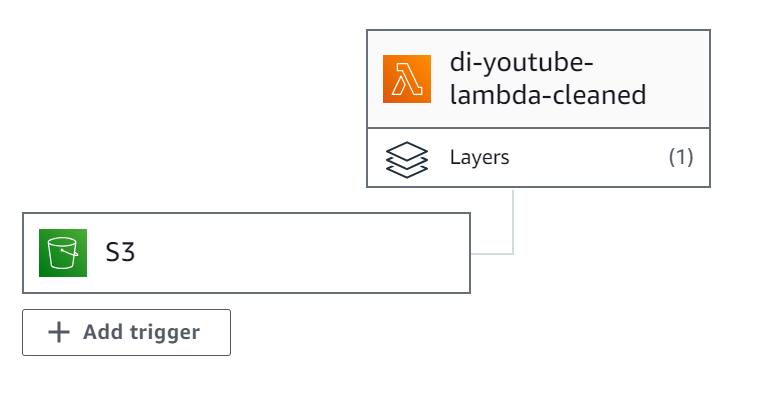
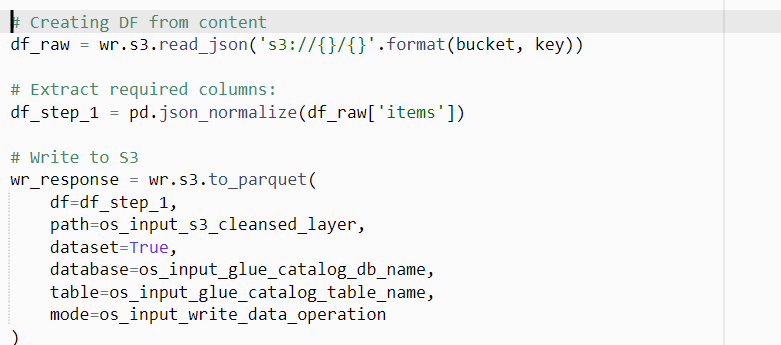


Fig. 10- Lambda Trigger

Lambda Function snippet



# AWS Step Functions

Visual workflows for distributed applications. Step Functions is a visual workflow service that helps developers use AWS services to build distributed applications, automate processes, create data and machine learning (ML) pipelines. Automate extract, transform, and load (ETL) processes . The transformed parquet files were inner joined on id column and output was stored in a S3 bucket which will serve as a data store for the analytic layer.

After running the visual workflow the data from the .json and .csv files were joined on id column and partitioned by region column and category id and finally transformed into parquet format in snappy compression.

Final Analytic bucket :

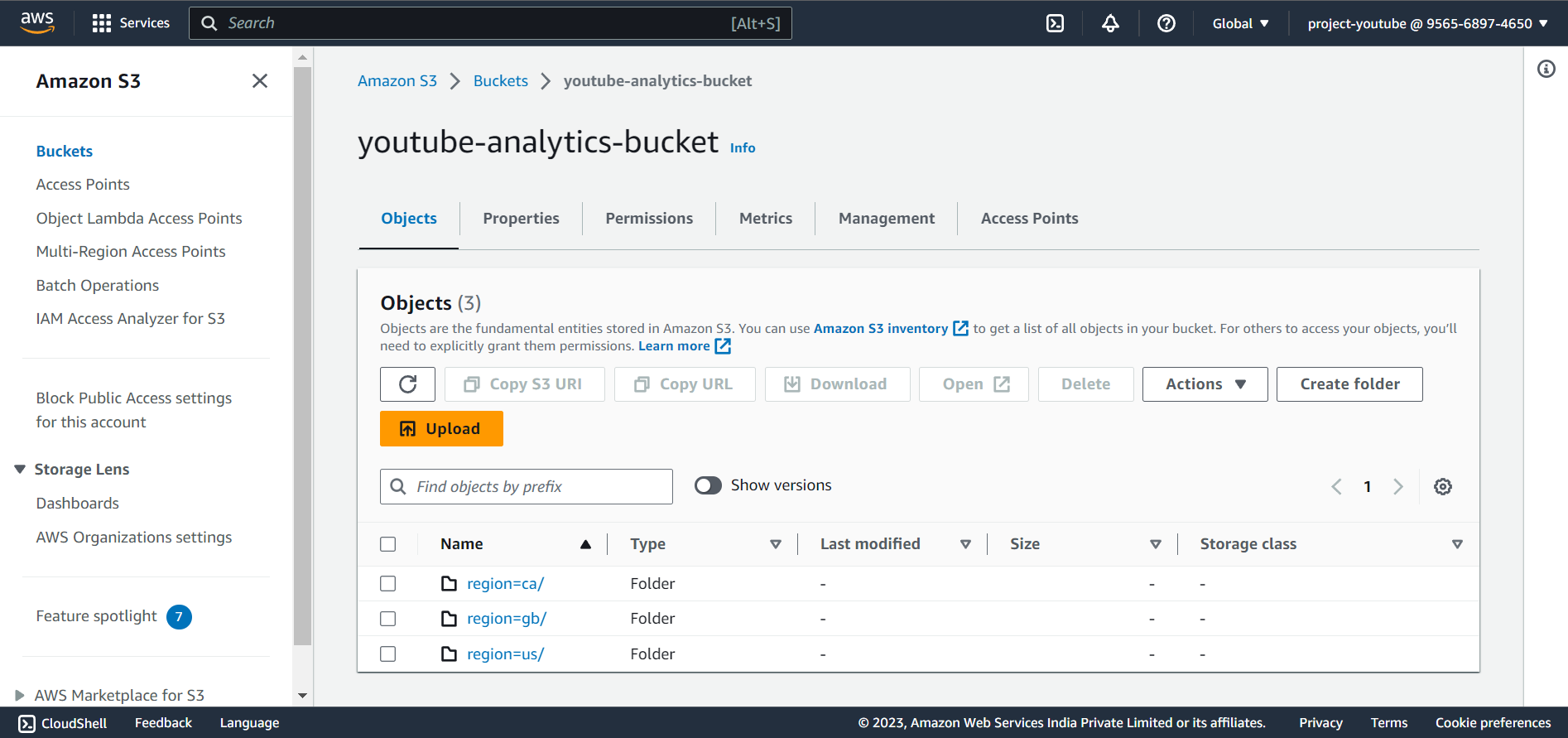


Fig. 11- S3 Bucket

# 

# Parquet File format

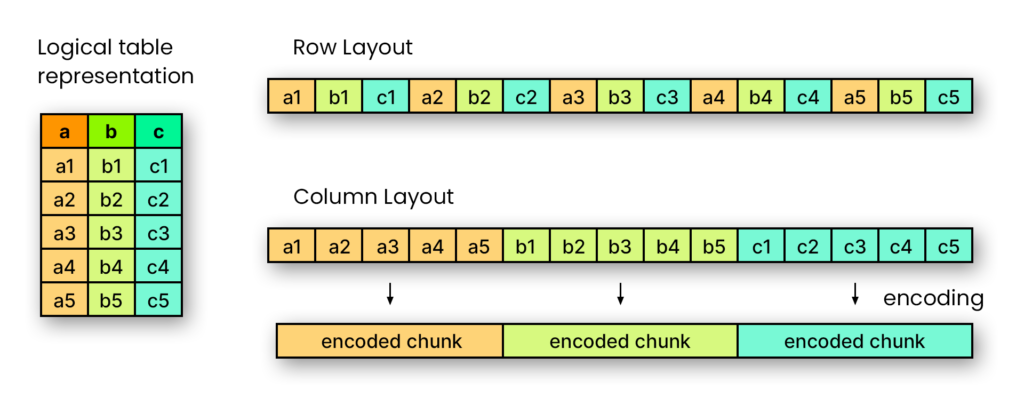


Fig. 12- Parquet file format

Parquet is an open source, column-oriented data file format designed for efficient data storage and retrieval. It provides efficient data compression and encoding schemes with enhanced performance to handle complex data in bulk. Parquet files are smaller than CSV files, and they can be read and written much faster. Parquet files also support nested data structures, which makes them ideal for storing complex data query services like AWS EMR ( Apache Hive ) or Amazon Athena charge you by the amount of data scanned per query. Google and Amazon charge you for the amount of data stored on GS/S3.

# AWS Quicksight

QuickSight is a fast business analytics service to build visualizations, perform ad hoc analysis. QuickSight seamlessly discovers AWS data sources, enables organizations to scale to hundreds of thousands of users, and delivers fast and responsive query performance. Amazon QuickSight Super-fast, Parallel, In-Memory, Calculation Engine (SPICE). We used QuickSight to perform analysis from the data store from the analytic bucket.

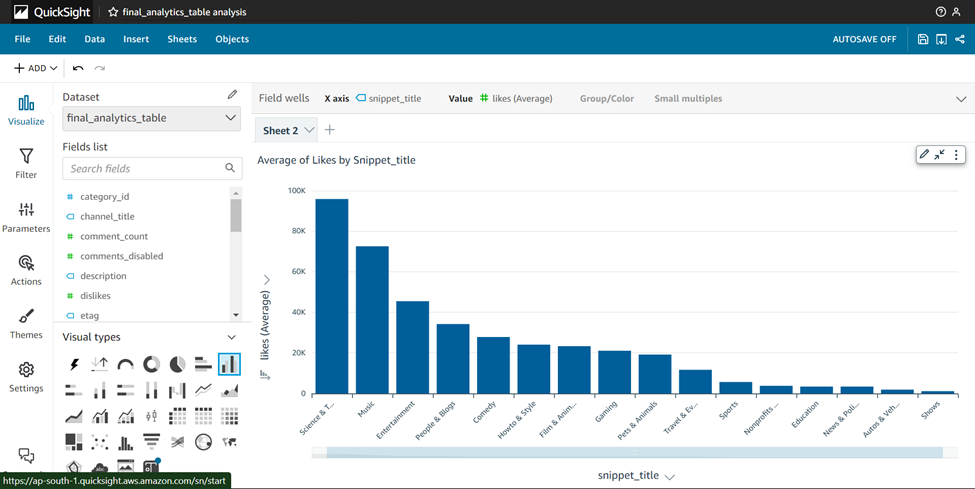


Fig. 13- Bar Plot for Average of likes by Snippet\_title

# Further Analysis:

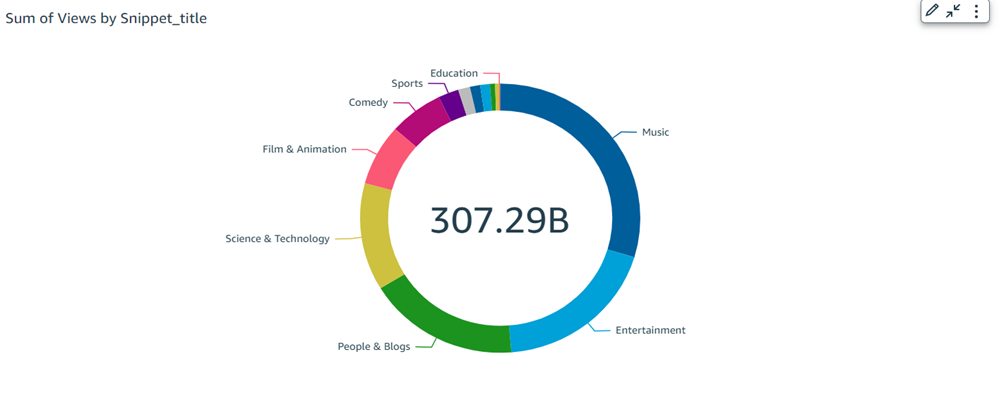


Fig. 14- Donut Chart of Sum if views by title

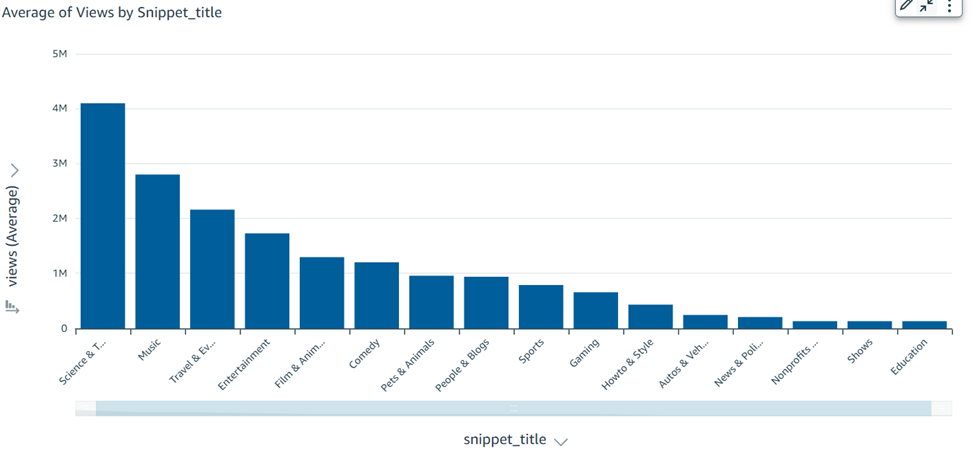


Fig. 15- Bar Chart of average of views by title

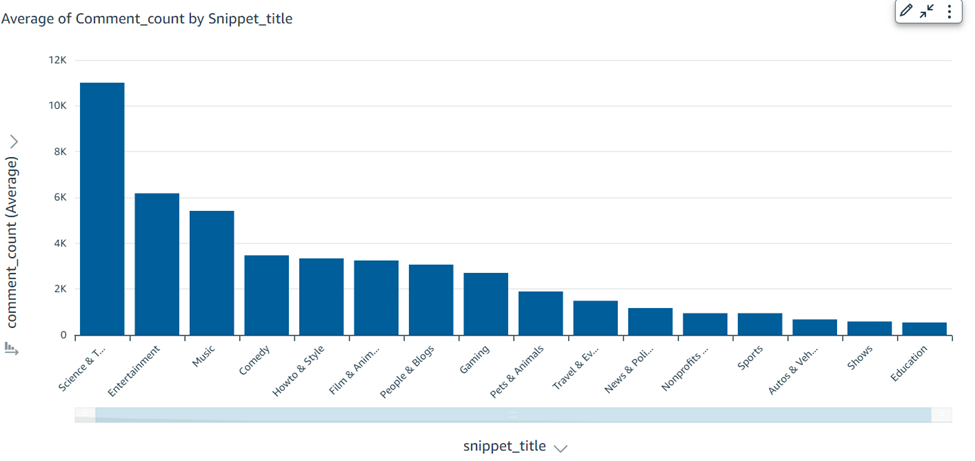


Fig. 16. Bar Chart of average of comment\_count by title

# Natural Language Processing

Natural language processing (NLP) refers to the branch of computer science—and more specifically, the branch of [artificial intelligence or AI](https://www.ibm.com/topics/artificial-intelligence)—concerned with giving computers the ability to understand text and spoken words in much the same way human beings can.

NLP combines computational linguistics—rule-based modeling of human language—with statistical, machine learning, and deep learning models. Together, these technologies enable computers to process human language in the form of text or voice data and to ‘understand’ its full meaning, complete with the speaker or writer’s intent and sentiment.

NLP drives computer programs that translate text from one language to another, respond to spoken commands, and summarize large volumes of text rapidly—even in real time. There’s a good chance you’ve interacted with NLP in the form of voice-operated GPS systems, digital assistants, speech-to-text dictation software, customer service chatbots, and other consumer conveniences. But NLP also plays a growing role in enterprise solutions that help streamline business operations, increase employee productivity, and simplify mission-critical business processes.

The Natural Language Processing (NLP) is also used for Text Classification. It is a process of automatically categorizing text data into predefined categories. It is used to classify and organize large amounts of unstructured data into meaningful categories. Text Classification can be used in various use cases such as sentiment analysis, spam detection, topic classification, document categorization, and more.

With the help of advanced algorithms, it can be used to analyze large amounts of text data quickly and accurately.

This makes it a powerful tool for NLP applications, such as machine translation, question answering systems, and automated customer service bots.

## Recurrent Neural Network

The model consists of an embedding layer that encodes words into continuous vector representations. Subsequently, a bidirectional Gated Recurrent Unit (GRU) layer captures intricate temporal dependencies in the word sequences. To exploit hierarchical relationships between words and phrases, a Global Average Pooling layer is introduced, reducing the variable-length sequences into fixed-size representations.

## Keras

Keras is an open-source deep learning framework that provides a high-level interface for building and training neural networks. It was developed by François Chollet and initially released in 2015. Keras aims to simplify the process of creating complex neural network models by offering a user-friendly and modular API that works seamlessly with popular backends like TensorFlow, Theano, and Microsoft Cognitive Toolkit (CNTK).

Key features of Keras:

1. User-Friendly API: Keras offers a simple and intuitive API that allows you to define and customize neural network models easily. It abstracts away much of the complexity of low-level operations, making it suitable for both beginners and experienced deep learning practitioners.

2. Modular Architecture: Keras is designed with a modular architecture. You can create neural network models by stacking and connecting layers in a straightforward manner. The framework provides a wide range of layer types, such as dense (fully connected), convolutional, recurrent, and more.

3. Flexibility: Keras allows you to quickly prototype and experiment with various architectures. You can create both sequential models (where layers are stacked sequentially) and functional models (where layers can have multiple inputs and outputs).

4. Backends: Keras is designed to be backend-agnostic, which means it can use different deep learning libraries as computation backends. TensorFlow is the default backend, but you can switch to Theano or CNTK as well.

5. Easy Model Visualization: Keras provides tools for visualizing models' architectures and summaries, making it easier to understand the structure of your neural networks.

6. Built-in Preprocessing: Keras includes utilities for data preprocessing and augmentation, which are essential for preparing data for training.

7. Wide Range of Applications: Keras supports a wide variety of tasks, including image classification, object detection, natural language processing, and more.

8. Integration with TensorFlow: Keras has been integrated as part of TensorFlow's official API since TensorFlow version 2.0, allowing developers to use TensorFlow as both a backend and a comprehensive machine learning framework.

Overall, Keras is a popular choice for rapid development and prototyping of neural network models due to its ease of use, flexibility, and strong integration with TensorFlow.

# 

## Transfer Learning

Transfer learning is a machine learning technique in which a model trained on one task is leveraged to perform well on a related task. Instead of training a model from scratch for a specific task, transfer learning allows you to start with a pre-trained model that has learned features from a different but related task. This approach can save time, computational resources, and data, while often leading to better performance on the target task.

Here's how transfer learning generally works:

1. Pre-trained Model: A pre-trained model is a neural network that has been trained on a large dataset for a different task. This dataset is usually extensive and diverse, allowing the model to learn general features and representations from the data.

2. Feature Extraction: The pre-trained model contains layers that have already learned to recognize useful features from the original task's data. These features can be considered as generalized representations of the data's underlying structure. These layers are often referred to as the "base" or "feature extraction" layers.

3. Fine-tuning: After importing the pre-trained model, you can fine-tune it on your target task. This typically involves modifying the top layers of the model or adding new layers to adapt it to the specific requirements of the target task. Fine-tuning helps the model learn task-specific details while preserving the valuable features learned from the original task.

Transfer learning can be categorized into a few common scenarios:

1. Feature Extraction: In this scenario, you use the pre-trained model's feature extraction layers and add a new classifier on top. You freeze the pre-trained layers and only train the new classifier on your dataset. This approach is useful when your new task has a small dataset or when the lower-level features learned by the pre-trained model are relevant to your task.

2. Fine-tuning the Entire Model: Here, you not only add a new classifier but also unfreeze some or all of the layers of the pre-trained model. This allows the entire model to be fine-tuned on your task's dataset. This approach is useful when your target task's dataset is larger and you want to adapt the pre-trained model to the specifics of your data.

3. Domain Adaptation: This involves transferring knowledge from a related but different domain. For example, you might have a model trained on images of animals and want to adapt it to images of cars. This involves aligning the features between domains to ensure effective transfer.

Transfer learning has proven to be highly effective, especially in scenarios where labeled data for the target task is limited. By leveraging the knowledge gained from one task, transfer learning enables models to achieve better performance, faster convergence, and improved generalization on new tasks. Popular pre-trained models for transfer learning include ImageNet-trained models (for image tasks) and models like BERT (for natural language processing tasks)

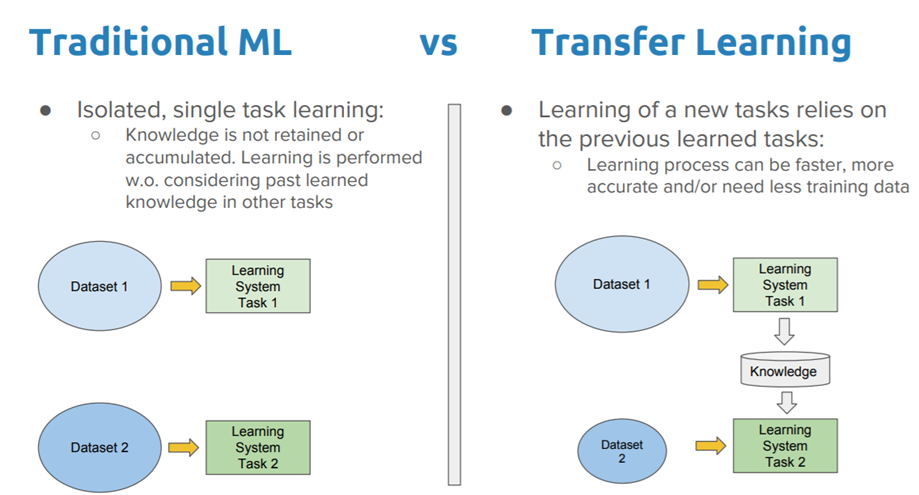


Fig. 17- Traditional ML vs Transfer Learning

## Softmax classifier

A softmax classifier, also known as a softmax regression or a multinomial logistic regression, is a type of mathematical function used in machine learning and deep learning for classification tasks. It's particularly useful when you need to classify data into multiple classes. The softmax classifier computes the probability distribution over the classes for a given input and assigns the input to the class with the highest probability.

Here's how the softmax classifier works:

1. Input and Weights: Given an input vector (usually the output of a neural network's last layer) and a set of weights associated with each class, the softmax classifier computes a score for each class.

2. Exponentiation: The scores are exponentiated, which transforms them into positive values. Exponentiation is important as it amplifies the differences between the scores, making the classifier more confident about the prediction.

3. Normalization: The exponentiated scores are then normalized by dividing them by the sum of all exponentiated scores. This normalization step ensures that the resulting values form a valid probability distribution, where each value represents the probability of the input belonging to a specific class.

4. Output Probabilities: The resulting normalized values are the predicted probabilities for each class. The class with the highest probability is selected as the final predicted class.

softmax function is defined as:

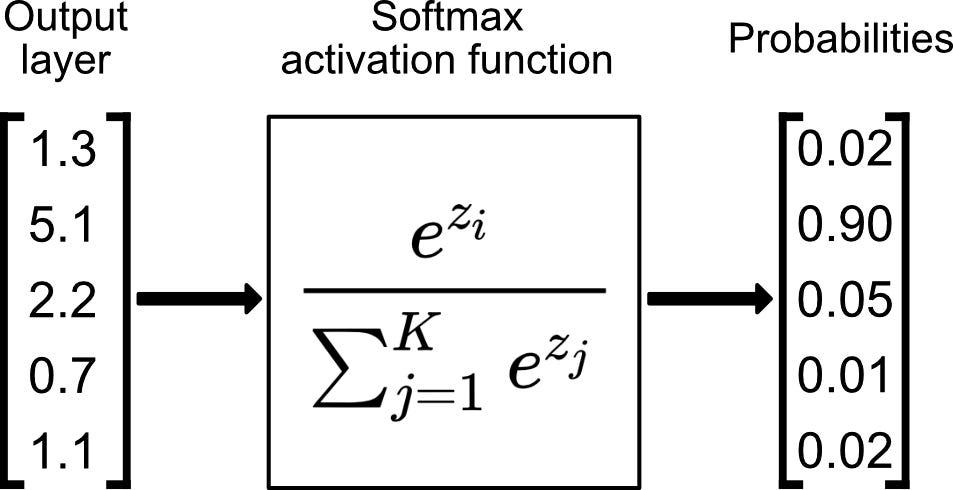


Fig. 18- Softmax activation function

The softmax classifier is commonly used as the output layer of neural networks for multi-class classification problems. It's often paired with a loss function called the "cross-entropy loss" or "log loss," which measures the difference between the predicted probabilities and the true labels.

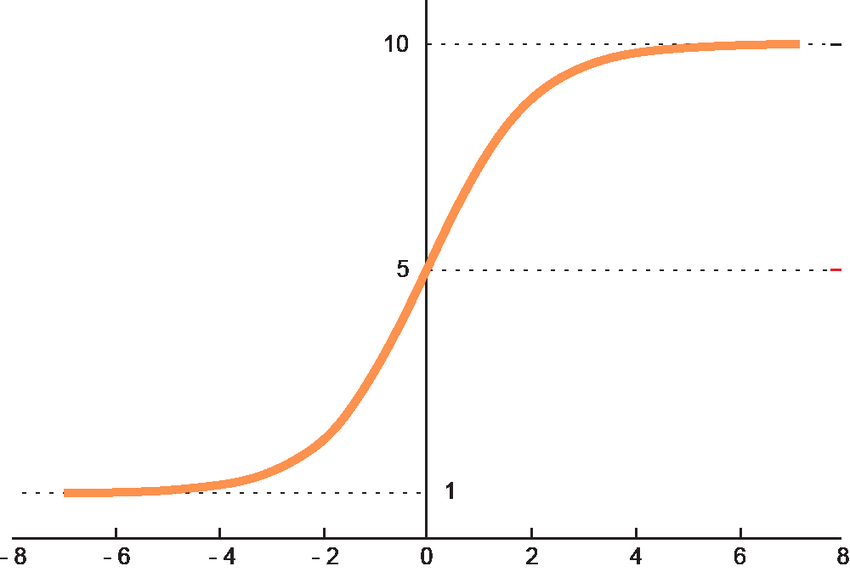


Fig. 19- Graph of softmax function

In summary, the softmax classifier transforms input scores into a probability distribution over multiple classes, making it a fundamental component for solving classification tasks in machine learning and deep learning.

## Pickle and HDF5

**Pickle:**

Pickle is a Python-specific serialization module that allows you to convert complex Python objects (such as data structures and classes) into a byte stream. This byte stream can be saved to a file or transmitted over a network, and then unpickled to reconstruct the original Python objects. Here are some key points about Pickle:

**Python-Specific:** Pickle is specific to the Python programming language. While it's great for serializing Python objects, it might not be compatible with other programming languages.

**Flexibility:** Pickle can serialize a wide range of Python objects, including custom classes and complex data structures. It's particularly useful for preserving the internal state of objects.

**Limitations:** Pickle might not be the best choice for long-term storage or cross-language compatibility, as its serialization format is tied to Python's internals and might change between Python versions.

**HDF5 (Hierarchical Data Format 5):**

HDF5 is a versatile data storage format that is not specific to any programming language. It's designed to store large amounts of complex data, providing support for various data types, compression, and efficient querying. Here are some key points about HDF5:

**Cross-Language Compatibility:** HDF5 is not tied to any specific programming language, making it suitable for data exchange between different languages and platforms.

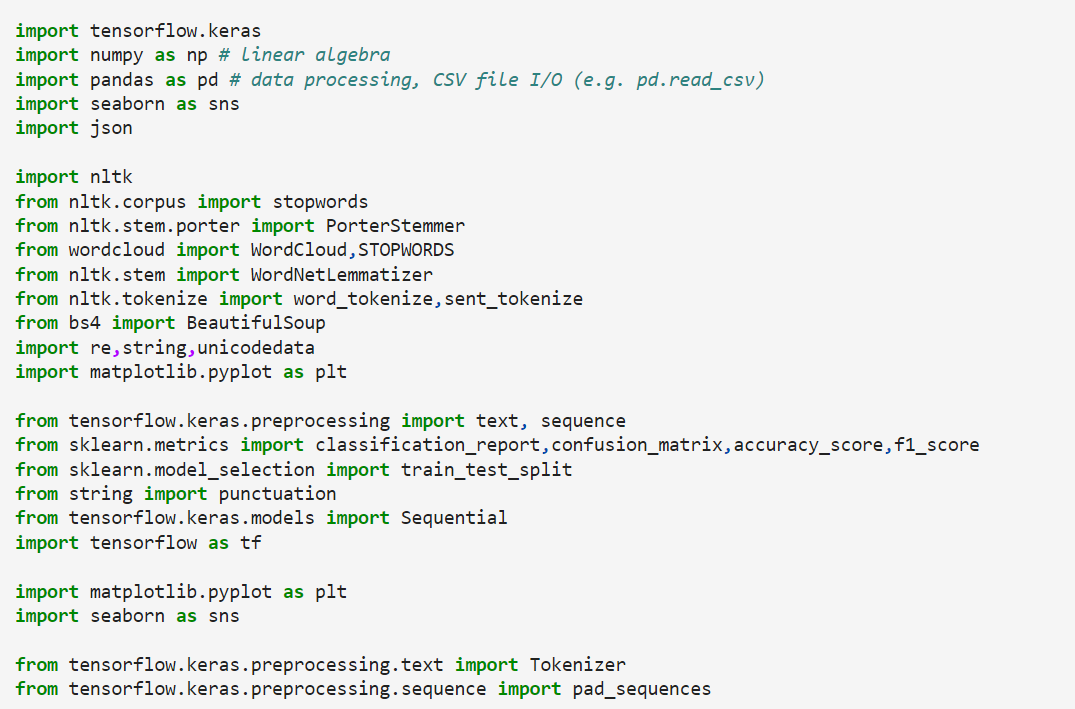
**Structured Data:** HDF5 allows you to organize data hierarchically into groups and datasets. Each dataset can have attributes and can be multidimensional, making it great for scientific and numerical data.

**Performance:** HDF5 supports data compression and chunking, which can lead to efficient storage and retrieval, especially for large datasets.

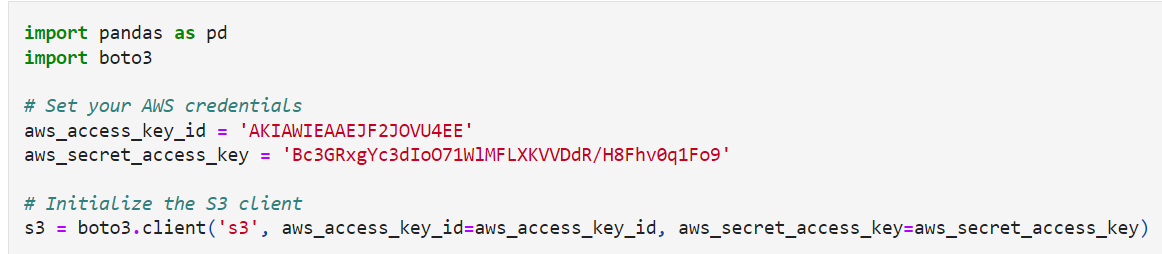
**Community Support:** HDF5 is widely used in scientific computing and data analysis, so it has a strong community and various libraries and tools for working with HDF5 files.

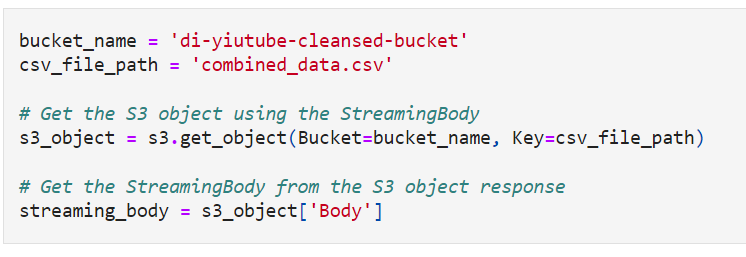
# Code Snippet

Library Imported

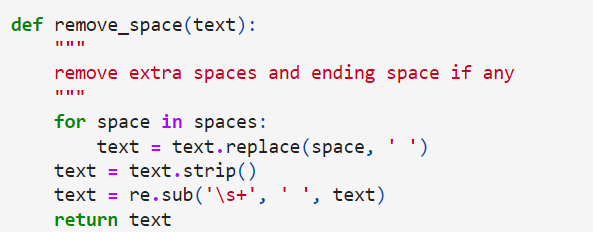


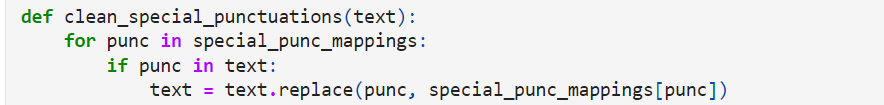
boto3 library used to connect to s3 bucket and retrieve data

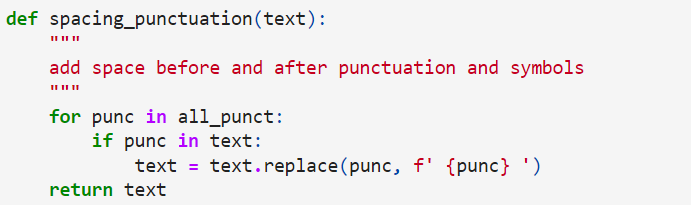


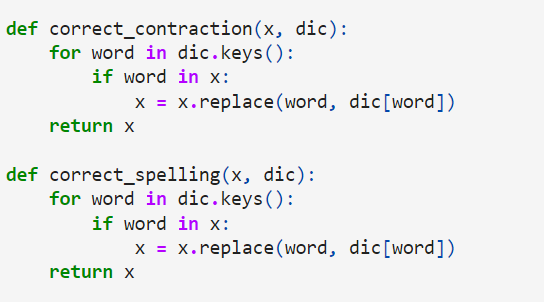


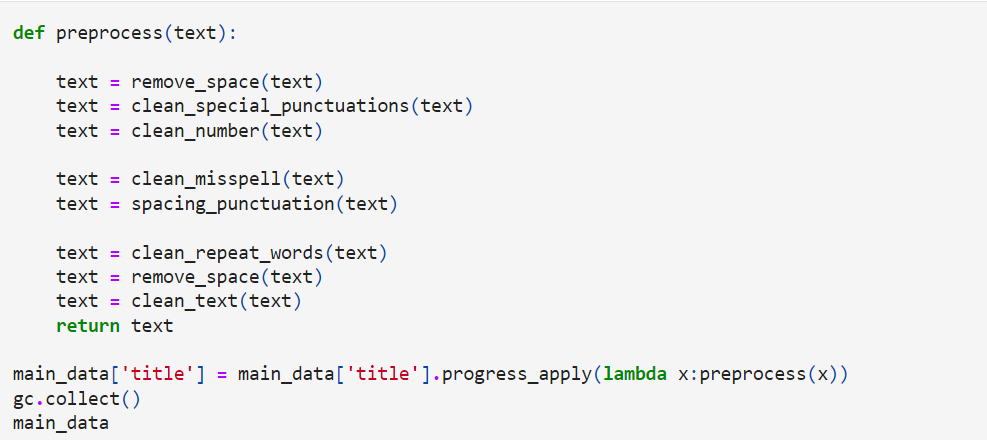
User defined functions for cleaning the sentences in title column



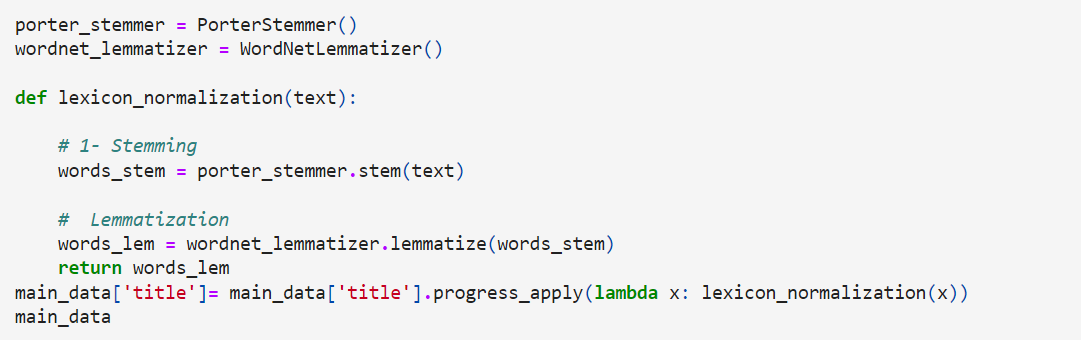




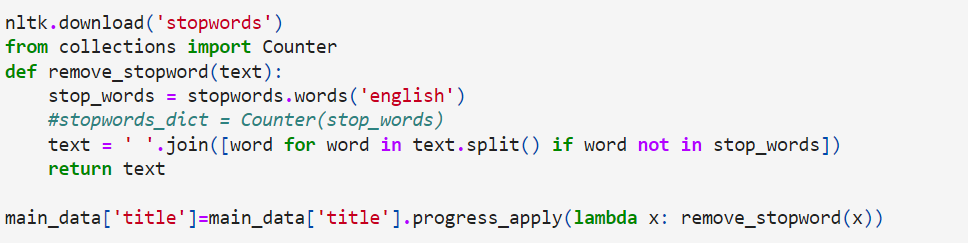




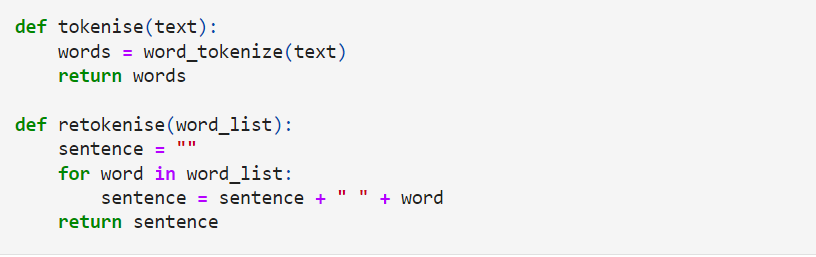
The stemming and lematize algorithm reduces dimensions to the text to improve model performance and increase the speed



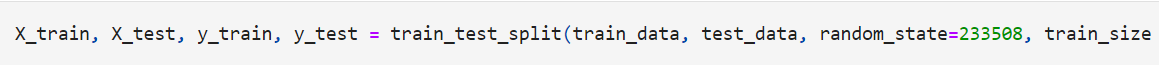
To remove all the stop words in the stopwords library in the nltk package.It removes common words which have no effect in prediction



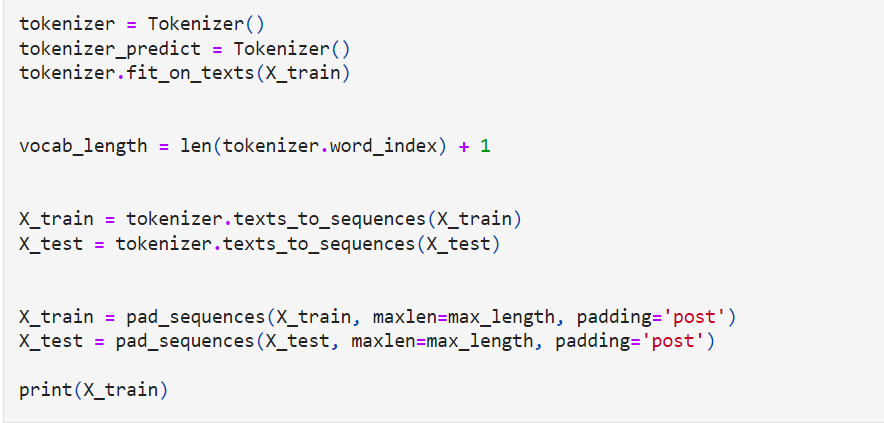
Then Tokenize the text which is an identification of basic units to be processed and retokensise by joining back to the sentence



Then train,test split the data



Before training the model the text is converted to sequences of integers and padded so that all integers are of same length



The model we used is a Recurrent Neural Network

Embedding Layer:

The first layer is an embedding layer. This layer is responsible for converting integer-encoded tokens (words or other units) into dense vectors of fixed size

Bidirectional GRU Layer:

The next layer is a bidirectional GRU (Gated Recurrent Unit) layer with 256 units. The bidirectional GRU processes the input sequences in both forward and backward directions, capturing information from both past and future contexts

Global Average Pooling Layer:

The Global Average Pooling layer calculates the average value for each feature across the time dimension of the sequences. This reduces the sequence dimension to a fixed-size representation.

Dense Layers:

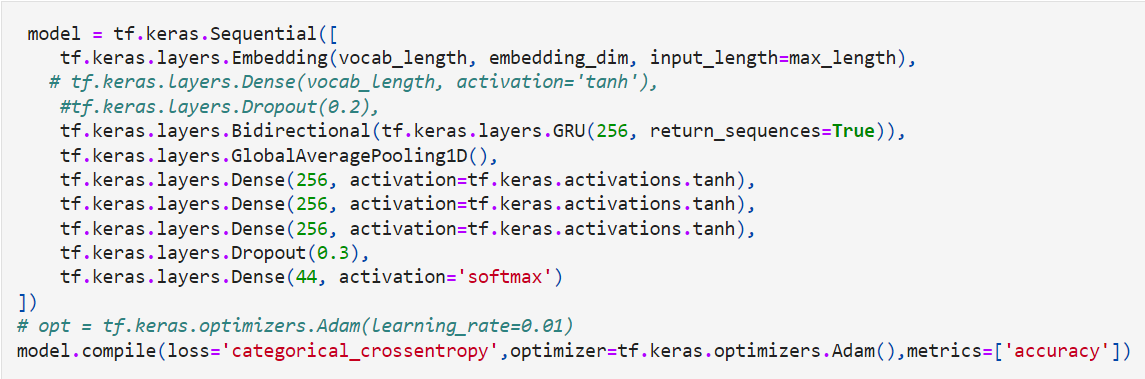
There are three consecutive dense layers, each with 256 units and tanh activation functions. These dense layers are likely intended to capture complex patterns and relationships in the data.

Dropout Layer:

A dropout layer is added after the dense layers with a dropout rate of 0.3

Final Dense Layer:

The last dense layer has 44 units and uses the softmax activation function to convert the model's raw outputs into probabilities for each class.



The model is trained



**Classification Report**

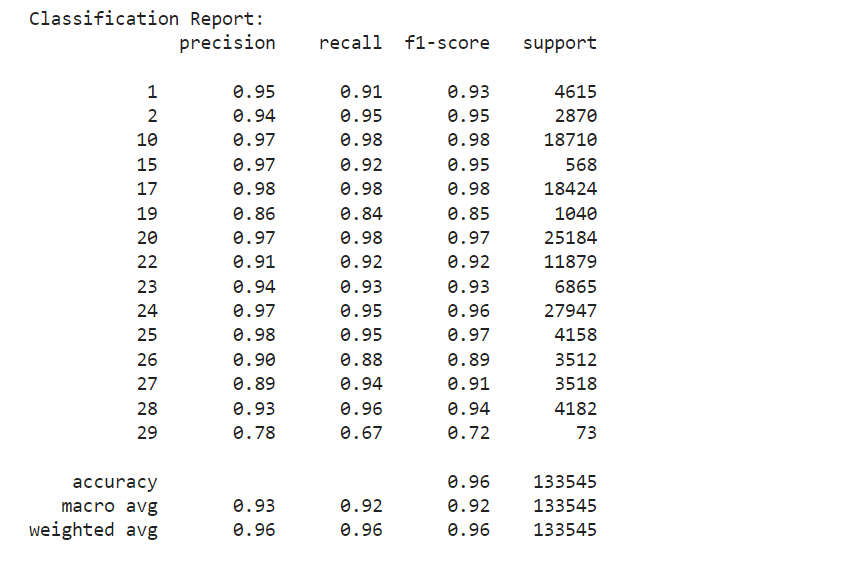


Fig. 20- Classification Report

The model has F1 score of 95% and accuracy of 97%.

**Training and Validation accuracy curve**

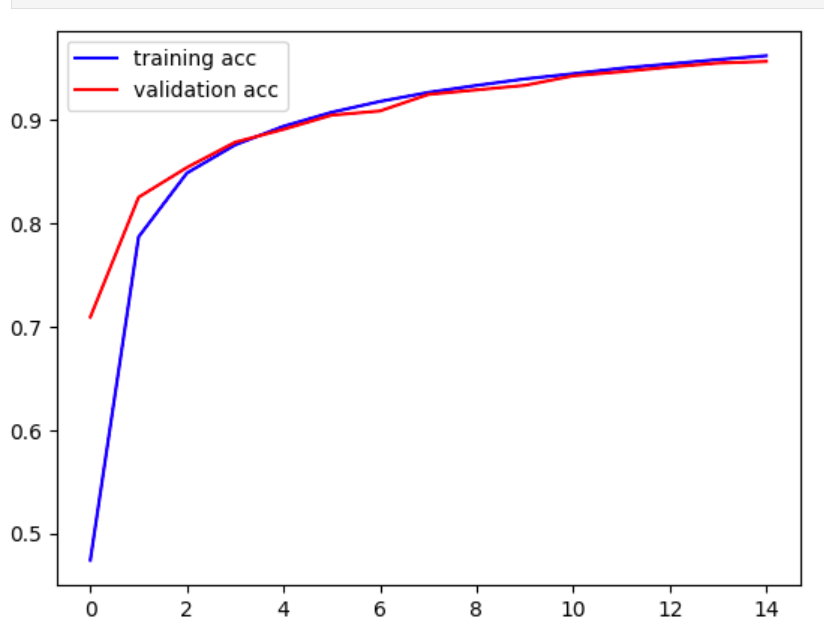
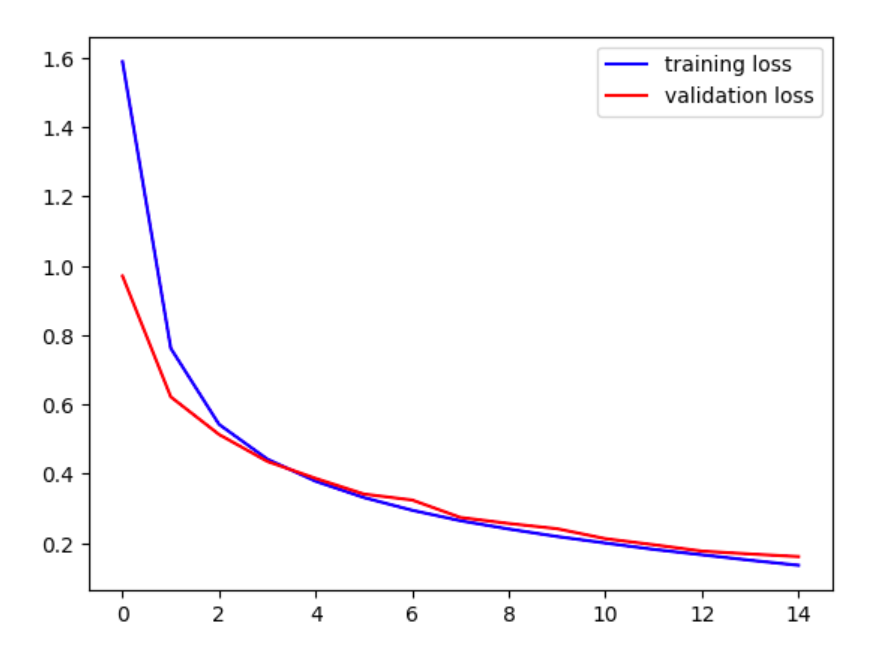


Fig. 21- Dual Line Chart of training and validation accuracy curve

Training and validation loss curve

Validation loss begins to increase slightly after 13th epoch.



# References

1. https://www.ibm.com/topics/natural-language-processing
2. <https://en.wikipedia.org/wiki/Web_crawler>
3. <https://machinelearningmastery.com/>
4. <https://towardsdatascience.com/>
5. <https://www.tensorflow.org/guide/keras>
6. <https://keras.io/api/>
7. <https://medium.com/data-science-365/how-to-apply-l1-and-l2-regularization-techniques-to-keras-models-da6249d8a469>
8. https://www.mygreatlearning.com/