|  |
| --- |
| Flat lay top view of robot deviating from group |
| Sentiment Analysis  Trainer Feedbacks |
| |  |  |  | | --- | --- | --- | | Xaltius Pte. Ltd. | 11/30/21 | Smart LMS System | |

Table of Contents

[USE CASE TITLE 2](#_Toc83901001)

[USE CASE DESCRIPTION 2](#_Toc83901002)

[HOW IS IT USEFUL? 2](#_Toc83901003)

[FLOW DIAGRAM 3](#_Toc83901004)

[FLOW DIAGRAM – EXPLANATION 3](#_Toc83901005)

[TECHNICAL ARCHITECTURE 4](#_Toc83901006)

[TECHNICAL ARCHITECTURE – EXPLANATION 4](#_Toc83901007)

[DEPLOYMENT PROCESS 5](#_Toc83901008)

[TRAINING PROCESS 5](#_Toc83901009)

[INFERENCE PROCESS 5](#_Toc83901010)

[AWS COSTING 5](#_Toc83901011)

[HOW TO USE THE DEPLOYED SYSTEM? 6](#_Toc83901012)

[REFERENCES 7](#_Toc83901013)

# USE CASE TITLE

Sentiment Analysis on Trainer Feedbacks

# USE CASE DESCRIPTION

Post-process trainer feedbacks made by students to obtain overall sentiment polarity and key adjectives describing the feedback text.

## HOW IS IT USEFUL?

This use case contributes to the improvement of the trainers by given them automated feedback on their teaching. The adjectives will inform them on what they are doing correctly and if negative, what they are doing wrongly so they know what they can improve on in their teaching.

# FLOW DIAGRAM

Diagram

Description automatically generated

## FLOW DIAGRAM – EXPLANATION

Docker image encapsuling the execution script is built and pushed into AWS ECR for serverless deployment via AWS Lambda.

AWS Lambda executes an AWS Sagemaker Processing Job instance using inbuilt boto3 python library.

Sagemaker Processing Job will query data from RDS and update/insert data into the respective tables in RDS upon job completion.

Function is scheduled via AWS EventBridge as a Cron job of biweekly frequency.

# TECHNICAL ARCHITECTURE

Diagram

Description automatically generated

## TECHNICAL ARCHITECTURE – EXPLANATION

Pipeline comprises of 2 major components, polarity and phrase model.

Prior to inference, raw text is preprocessed by removing non-alphanumeric characters and lowercased.

The polarity model utilizes a pretrained model (distilroberta-base) as its feature extractor and uses a Sequence Classification framework to categorise the sentiment polarity of the input text.

The sentiment polarity is then ingested into the phrase model which uses a pretrained model (bert-base-uncased) with a Question&Answering framework to infer the related text that motivated the sentiment polarity.

# DEPLOYMENT PROCESS

Both training and inference pipelines were deployed using serverless pipeline utilizing customized docker images hosted via AWS Elastic Container Registry (ECR).

Docker images were built with GPU capabilities by building on top of NVIDIA’s TensorRT docker images that includes all NVIDIA Drivers and its prerequisities.

The docker images saved in ECR are then called through AWS built-in library Boto3 using AWS Lambda functions. Outputs from the called functions queries and updates tables in AWS RDS from within the docker image itself.

## TRAINING PROCESS

The models were trained using a subset of Sentiment140 dataset. The polarity model was trained first to categorize the sentiment polarity of the text whilst the phrase model was trained after to infer the sentiment text contributing to the polarity.

|  |  |  |
| --- | --- | --- |
| Parameter | Polarity | Phrase |
| Pretrained weights | bert-base-uncased | distilroberta-base |
| Optimizer | Adam | Adam |
| Learning Schedule | Constant | Cosine Decay with Restarts |
| Initial learning rate | 1e-5 | 1e-5 |
| Loss metric | Categorical Cross-entropy | Categorical Cross-entropy |
| Evaluation metric | Micro F1-score | Jaccard Similarity (MeanIOU) |
| Epochs | 5 | 20 |
| Batch Size | 32 | 32 |
| Early Stopping | Yes | Yes |
| Mixed precision | Yes | Yes |
| Final scores | F1: 0.8196, Val-F1: 0.7910 | Jac: 0.7981, Val-Jac: 0.8023 |

## INFERENCE PROCESS

The inference pipeline was built as an end-to-end structure comprising of both the polarity and phrase models. Output from the polarity model is fed directly into the phrase model to infer the sentiment phrases.

## AWS COSTING

Assuming inference API is called biweekly and training API annually.

|  |  |  |  |
| --- | --- | --- | --- |
| Function | Instance type | Hours | Costing ($ USD) |
| Sagemaker Processing Job | ml.g4dn.xlarge | 0.25 | 0.52 |
| ECR | - | - | Free tier |
| Lambda | - | - | Free tier |
| RDS | - | - | Free tier |
| Total |  |  | **0.52 per month** |

Inference API

|  |  |  |  |
| --- | --- | --- | --- |
| Function | Instance type | Hours | Costing ($ USD) |
| Sagemaker Processing Job | ml.g4dn.xlarge | 4.0 | 4.12 |
| ECR | - | - | Free tier |
| Lambda | - | - | Free tier |
| S3 | - | - | Free tier |
| Total |  |  | **4.12 per year** |

Training API

# HOW TO USE THE DEPLOYED SYSTEM?

As this function is self-regulated, there will be no instructions for usage.

# REFERENCES

<https://huggingface.co/bert-base-uncased>

<https://huggingface.co/distilroberta-base>

<https://arxiv.org/abs/2101.02157>

<https://huggingface.co/transformers/model_doc/bert.html>

<https://www.kaggle.com/kazanova/sentiment140>

<https://www.kaggle.com/c/tweet-sentiment-extraction/data>