CMPE 297 - Special Topics

ALZHEIMER'S - MEDICAL IMAGE ANALYSIS USING SELF SUPERVISED AND SEMI SUPERVISED LEARNING APPROACHES

Team Equality

Abhishek Bais

Haley Feng

Princy Joy

Shannon Phu

Table of contents

Abstract

Alzheimer's disease is a neurodegenerative disease primarily affecting middle-aged and elderly people. Medical imaging techniques are vastly used for primary diagnosis and a lot of computer aided algorithms assist physicians in analysing the brain images. Deep learning based techniques are becoming increasingly adopted in the medical field for detection and study of diseases. The challenging part of creating deep learning models is the lack of high quality labelled dataset. Such conditions are ideally suited for self-supervised or semi-supervised deep learning. In this paper we will discuss our study on classifying the Alzheimer's brain MRI images using self-supervised and semi-supervised techniques.

Introduction

Alzheimer's disease is a common form of dementia that affects many elderly people and causes cognitive decline and memory loss. The National Institute on Aging reports that over 6 million Americans, most of them aged 65 or older, may suffer from dementia caused by Alzheimer's, making it the sixth leading cause of death of elderly Americans [1]. Alzheimer's is a progressive disease, in which the symptoms of dementia gradually worsen over time. On average, a person with Alzheimer's survives approximately 4 to 8 years after being diagnosed. In spite of the fact that Alzheimer's is incurable, early diagnosis and treatment strategies can greatly improve treatment options and quality of life for patients.

Alzheimer's causes structural variations in the brain and the early stage can be diagnosed by analyzing these structural variations. Magnetic resonance imaging (MRI), a method of evaluating the brain's anatomical structure, is a standard way to diagnose Alzheimer's disease [2]. This imaging technique assesses the hippocampus part of the brain and determines the amount and level of atrophy. In recent years, artificial intelligence methods based on deep learning are gaining considerable attention in evaluating early dementia effectively, and have become a focus of dementia research. Methods that rely on deep learning are excellent at extracting image features automatically that would otherwise be very tricky to be captured over traditional machine learning or statistical models.

Deep learning models rely heavily on a huge amount of training data to create powerful and accurate models which have a lot of parameters. One of the challenges in preparing such a huge dataset is the process of annotating it with labels. It requires a considerable amount of time, money and expertise to produce high quality annotations. Therefore, despite the large availability of data in the medical field, it is unusable for lack of good labels. This is where self-supervised and semi-supervised learning methods can be of great practical value.

Self-supervised learning is a form of unsupervised learning where data provides supervision for the training task. As a first step the neural network learns useful representations of data from a large pool of unlabelled data. This is called a pretext task and does not involve any labels. The second step is finetuning the learned neural network with fewer labels for downstream tasks like image classification.

Semi-supervised learning is a combination of supervised and unsupervised learning which uses a small portion of labeled data and a large number of unlabeled data from which the model must learn and make predictions on new examples.

In this study, we used the self-supervised learning, SimCLR (A Simple Framework for Contrastive Learning of Visual Representations) [3] as well as the semi-supervised learning, SimCLRv2 [4] to classify Alzheimer's MRI images into four classes based on severity.

Related work

The artificial intelligence based diagnosis methods are widely adopted and researched in the medical field. These methods have been found to be really helpful for clinicians to analyze and make decisions on important medical procedures. There are statistical methods, traditional machine learning models and deep learning models used for various medical tasks. We will go over some of the history and relevant studies in Alzheimer's medical image analysis.

Support Vector Machines (SVM) is a supervised machine learning model. It is a very stable and widely used machine learning algorithm for classification tasks [5]. However the features need to be handcrafted by domain experts and this is a tedious aspect of traditional models. The deep neural network can identify the underlying patterns and correlations and efficiently extract the feature vectors without an expert needing to craft it. Convolutional neural networks (CNNs) have shown remarkable performance in image recognition, and were also successfully applied to diagnosing Alzheimer's disease (AD) using multimodal neuroimaging data [6].

However, these methods are supervised learning algorithms that require a large number of samples and high-quality annotations in order to adequately represent features. It is possible to use self-supervised, semi-supervised, and unsupervised learning techniques in situations of limited labelled data. A weakly supervised learning technique is discussed in [7]. It forms the backbone network with an attention mechanism. This network can simultaneously perform image classification and image reconstruction on Alzheimer's images using limited annotations.
In the study [8], the self-supervised learning approach was demonstrated to be effective in various medical image analysis tasks including detection and classification. [9] is a semi-supervised learning applied on breast cancer histology classification and the results show that despite having fewer labels, effective models could still be created for various tasks.

Our experiments are guided by the landmark papers on SimCLR [3] and SimCLRv2 [4] as well as insights from the mentioned papers.

Dataset

The Alzheimer dataset [10] is sourced from Kaggle, provided by user Sarvesh Dubey, and consists of MRI images for four different stages of dementia (none, very mild, mild, moderate) Fig. 1 shows an example MRI image for each stage. The training dataset has 5,121 images and the testing dataset has 1,279 images. This dataset is highly imbalanced, as shown in Fig 2, with only 52 moderate demented images in the training dataset and 12 modeate demented images in the testing dataset. During the data preprocessing stage, every image is normalized and resized into a Numpy array with shape (1, 244, 244, 3).

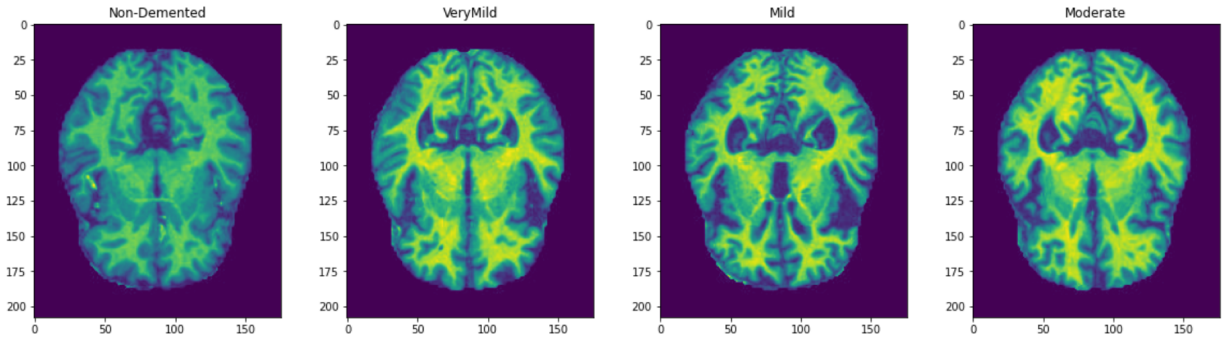


Fig 1: Sample images from each of the four classes of MRI images

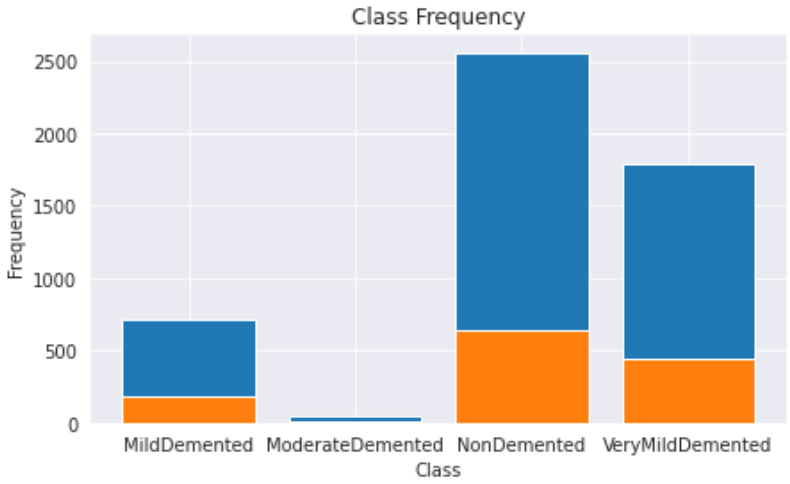


Fig 2: Class distribution of the Alzheimer Dataset

Methodologies overview

SimCLR

SimCLR [3] is a self-supervised learning technique. It learns representations by maximizing agreement between differently augmented views of the same data example via a contrastive loss in the latent space. It has four components.The first one is the data augmentation module,

which produces two correlated views of the same example. The second one is a neural network base encoder for extracting representation vectors from data. The third one is the projection head which is a multi layer perceptron with one hidden layer that maps the representations from the base encoder to 128-dimensional latent space where contrastive loss is applied. The fourth part is the Contrastive Loss Function, objective is to treat correlated images, positive pairs as similar and other pairs to be dissimilar.

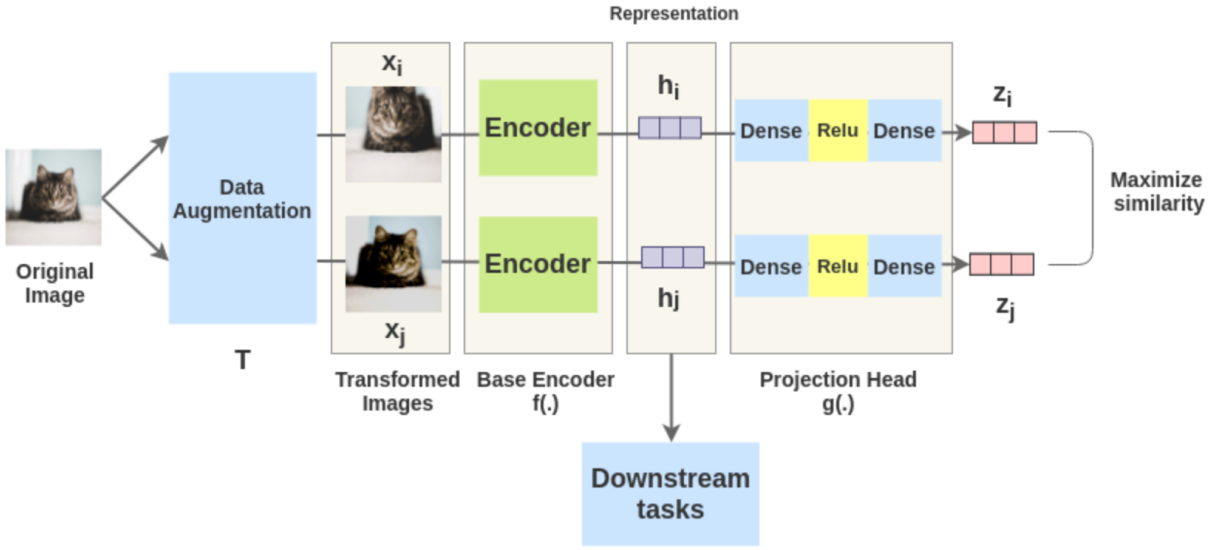


Fig 3: SimCLR Framework

The specific loss function used here is the Normalized Temperature-scaled Cross Entropy (NT-Xent).

SimCLRv2

SimCLRv2 [4] is the semi-supervised version of SimCLR. SimCLRv2 can be summarized in three steps. First step is the task agnostic unsupervised pre-training for learning useful representations. The second step is supervised fine-tuning on labeled samples for a specific task. The final step is the self-training or distillation with task-specific unlabeled examples. In the pre-training step, SimCLRv2 learns representations by maximizing the contrastive learning loss function. This loss function is a distance-based loss calculated on pairs of samples with the goal that augmented views of the same sample should be closer together in the embedding space while the rest should be far apart. The main architectural difference with SimCLR is that SimCLR uses 2 layer projection head and SimCLRv2 uses 3 layer projection head.

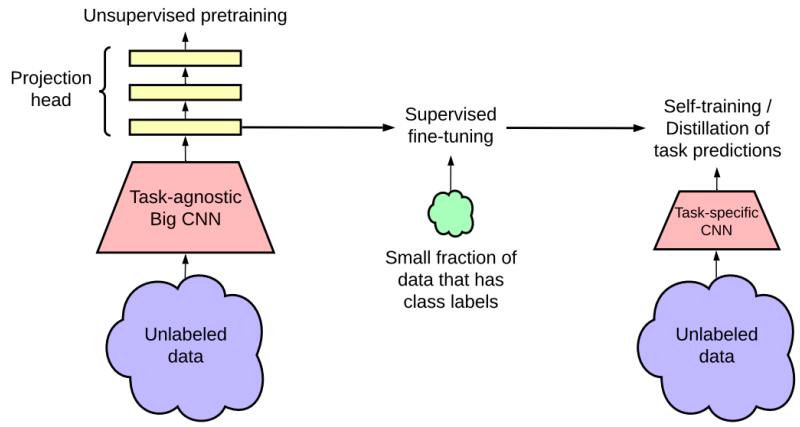


Fig 4: SimCLRv2 Framework

Teacher-Student Knowledge Distillation

Knowledge distillation [4] is an intuitive way of improving the performance of deep learning models on thin computational devices like mobiles. First, a teacher network is trained on huge amounts of data to learn the features. Then a student network is trained to make comparable predictions as the teacher.

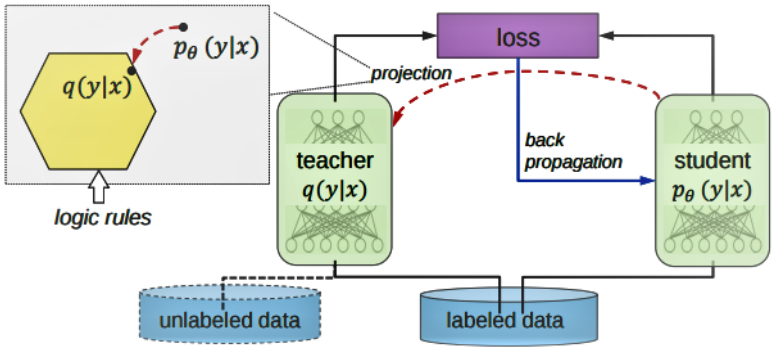


Fig 5: Knowledge distillation architecture

The student is trained to mimic teachers output instead of training on the raw data directly. Also the backpropagation takes place only on the student model as the teacher is already pre-trained and that knowledge is just transferred.

Model Development

Configuration

1. SimCLRv1 - Pre-trained on imagenet with Alzheimers images
    - Extract image features using Resnet50 using pretrained weights and bias from Imagenet.

- Create a one layer linear model with 128 features matching the last layer of the Resnet50 model.
- Set the activation to be softmax with loss as cross-entropy loss.
- Use a linear model for training and prediction with a 64 batch size and early stopping integrated.

2. SimCLRv2 - Fine-tune a pre-trained SimCLRv2 model on Alzheimer's images
    - Load a pre-trained SimCLR model on 100% of labels from the hub module (gs://simclr-checkpoints/simclrv2/finetuned_100pct).
    - Attach a trainable linear layer with LARS optimization (weight decay = 1e-3, momentum = 0.8, learning rate = cosine decay) and a batch size of 32.
3. SimCLRv2 - Perform Student Teacher knowledge distillation using fine-tuned simClrv2
    - Load a pre-trained SimCLR model on 100% of labels from the hub module (gs://simclr-checkpoints/simclrv2/finetuned_100pct).
    - Fine-tune the pre-trained SimCLR model on Alzheimer images to create a teacher model.
    - Build a student model with 3 convolutional layers, batch normalization, relu activation layer, max pooling layers, and 2 dense layers for prediction.
    - Train only the student model with the teacher's label with LARS optimization (weight decay = 1e-3, momentum = 0.8, learning rate = cosine decay) and softmax cross-entropy with temperature with a batch size of 16.

Evaluation Metrics

We evaluated our models' classification results primarily using Area Under the Curve (AUC) metric. We visualized our metrics using the ROC-AUC curve, Precision-Recall curve, and the Confusion Matrix. While accuracy is commonly used to check how many predictions matched the ground truth, it is not the best measure for assessing our models, since we have an imbalanced dataset. In AUC, the higher the value, the better the classifier. An AUC value of 0.50 means the model is performing similar to a random guess, and values below 0.50 mean the model is performing worse than a random guess.

Hyper-Parameter Tuning

To get improved accuracy we tuned the hyper-parameters for our models. In doing so, we experimented with different optimizers (Adam, RMSprop) and learning rate scheduling methods such as (step decay, cosine decay, and exponential decay). In our SimCLRv1 experiment, we found that the best hyper-parameter combination is a setting with the optimizer set to Adam and exponential decay as it produced the best accuracy, AUC, and F1-score across both on the training and testing dataset. The results of our hyper-parameter tuning experiments are captured in Fig 6. below.

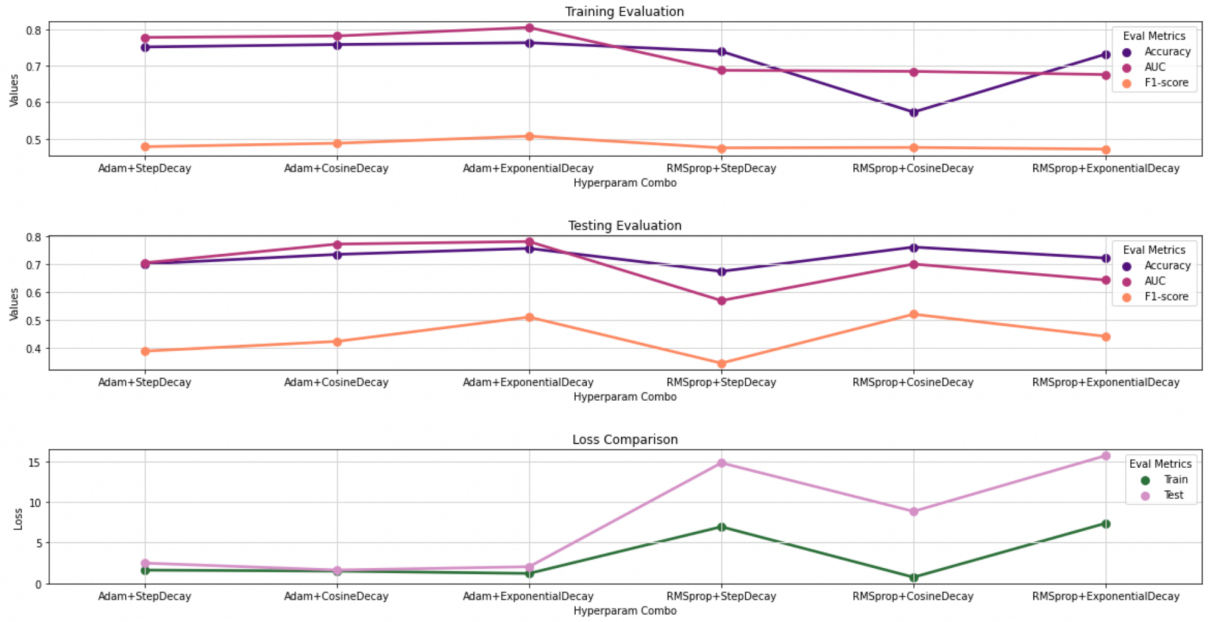


Fig 6: Hyper-parameter comparison on SimCLRv1. The first two plots show accuracy, AUC, and F1-score performance. The last plot shows the loss of each combination at training and testing.

Prediction results

Table 1: Each methods output at inference

Models comparison

From the ROC-AUC curves comparison, we observed that SimCLRv1 performed the best with the highest micro-average (0.82) and macro-average (0.68) numbers. SimCLRv1 also had the best micro-average (0.60) on the Precision-Recall curve. Across all the Confusion Matrices, we also observed that the incorrectly classified images for each model were skewed towards non demented or very mild demented. This aligned well with our initial concern of working with a highly imbalanced dataset where there are more images for these two classes than the rest of the classes. A comparison of the evaluation results for the three models (SimCLRv1, SimCLRv2

- fine tuned on Alzheimer images, and SimCLRv2 - with teacher, student distillation) is captured in Table 2. below.

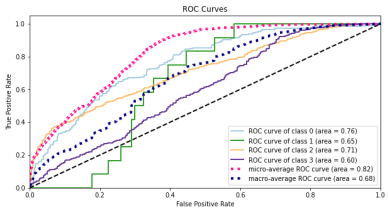
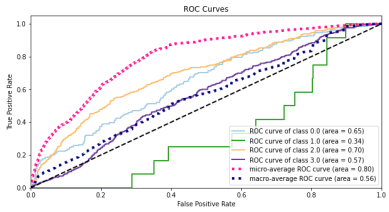
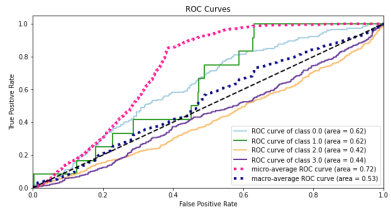
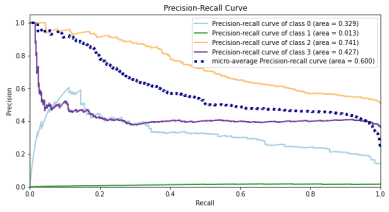
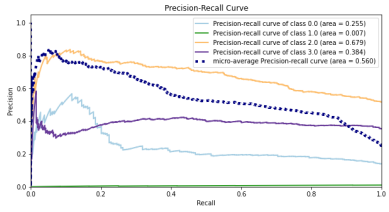
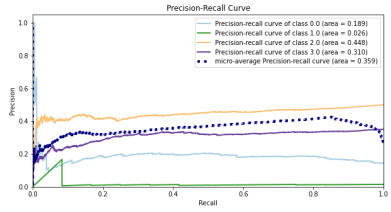
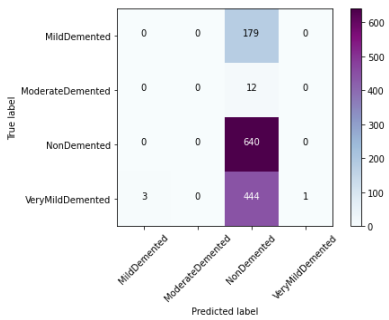
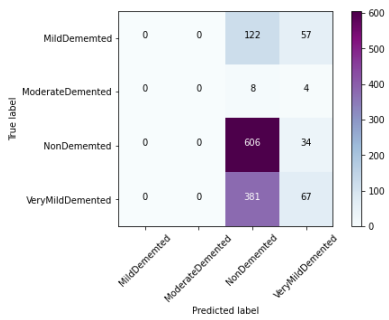
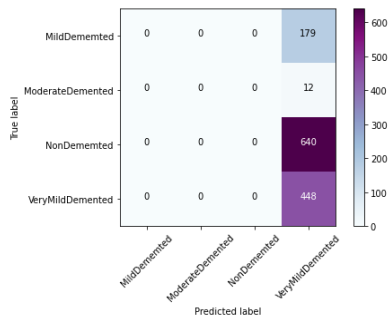
SimCLRv1	SimCLRv2 (finetune on alzheimer)	SimCLRv2 (teacher-student distillation)
		
		
		

Table 2: Method comparison using ROC-AUC curve, precision-recall curve, confusion matrix

Model deployment

We successfully implemented an end-to-end Machine Learning Pipeline using TensorFlow Extended (TFX) and Kubeflow for our project. We deployed our models on the google AI platform using which and together with GitHub we performed several ML-OP2 tasks such as CI/CD, Model versioning, Model Gating, Pipeline Monitoring, Skew, Drift and Anomaly Monitoring. We used TensorFlow serving and Rest APIs to use the model to make live predictions. These technologies enabled us to automate the model development, training, inference and management of endpoints.

TFX pipeline

TFX is a production-scale machine learning platform based on TensorFlow. It has a sequence of components that implement a ML pipeline specifically designed for scalable, high-performance machine learning tasks. These components are organized into Directed Acyclic Graphs (DAG)

of tasks which are executed in a particular sequence. TFX pipeline can be orchestrated using Apache Airflow or Kubeflow. The main advantage of TFX pipelines is that they avoid code duplication and eliminate possible training/serving skew. The pipeline ensures that both the training and inference data consistently go through the same set of code.

TFX pipelines typically have the following components: ExampleGen, StatisticsGen, SchemaGen, ExampleValidator, Transform, Trainer and Pusher. These are discussed in the following sections.

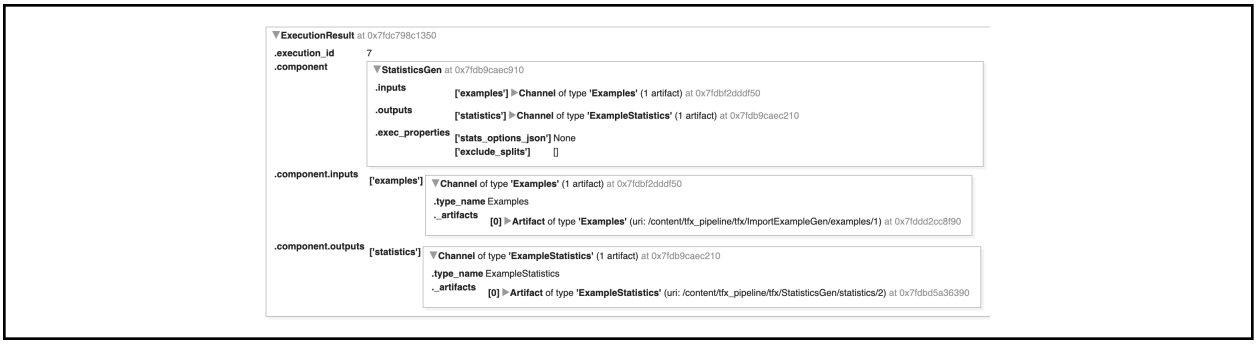
ExampleGen

Here, the train and test datasets are preprocessed and transformed into TFRecord which is Tensorflow's standard file format. The ExampleGen is the entry point of the pipeline that ingests these TFRecords and splits them into train-test datasets. ExampleGen emits tf.Example records which are consumed in the subsequent stages.

Table 3: ExampleGen output

StatisticsGen

StatisticsGen computes the general statistics of the dataset and also provides visualization of the results. This stage consumes the tf.Examples from ExampleGen stage and emits dataset statistics.



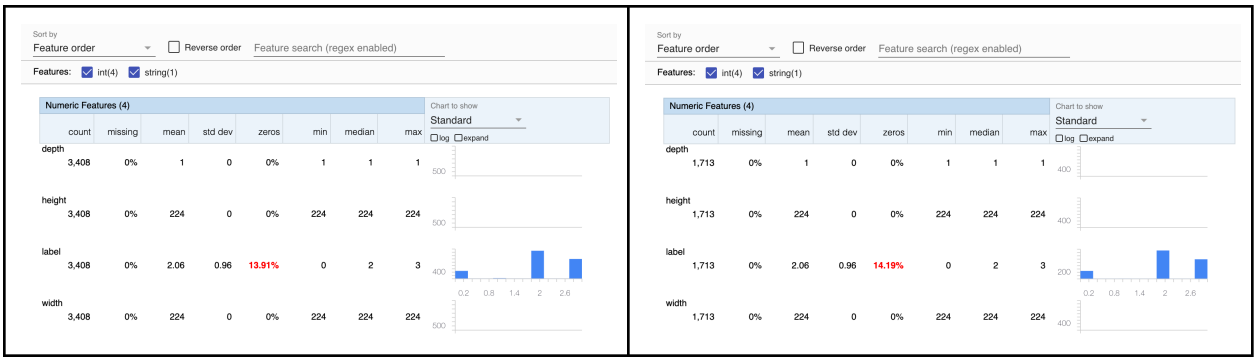


Table 4: StatisticsGen output

SchemaGen

The SchemaGen pipeline component automatically generates a schema by inferring types, categories, and ranges from the ingested data. This stage consumes statistics from the StatisticsGen component and emits data schema proto.

Table 5: SchemaGen output

ExampleValidator

ExampleValidator checks for missing values and anomalies in the data and reports them if detected. This stage also detects training/serving skew and data drift by looking at a series of data. It consumes schema from a SchemaGen component, and statistics from a StatisticsGen component and emits validation results.

Table 6: ExampleValidator output

Transform

The Transform component performs feature engineering on the dataset. It takes in the output artifacts from the ExampleGen and SchemaGen stages and emits a SavedModel for the Trainer component.

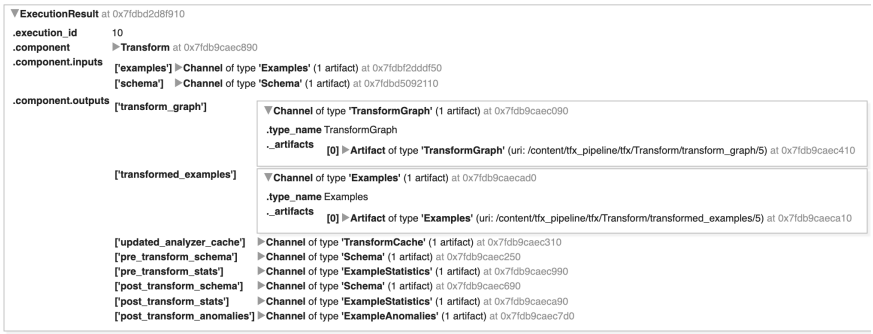
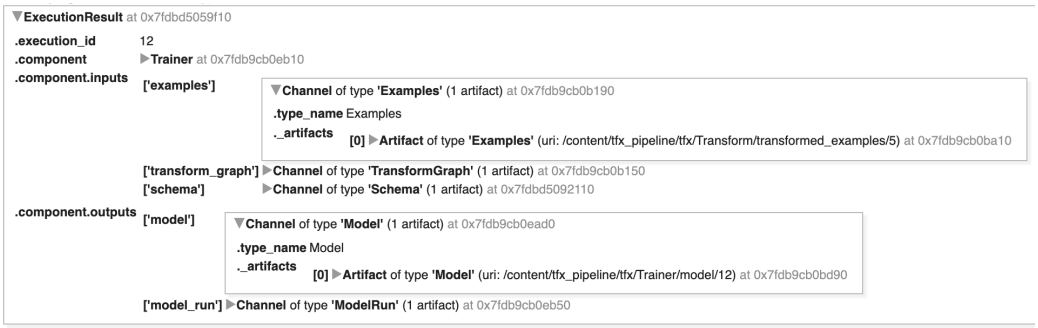


Table 7: Transform output

Trainer

Model training is performed by this component. The trainer consumes artifacts from previous stages and emits at least one model for inference/serving in the SavedModelFormat. The training epoch logs can be viewed. A tensorboard visualization also can be launched from the monitoring logs.



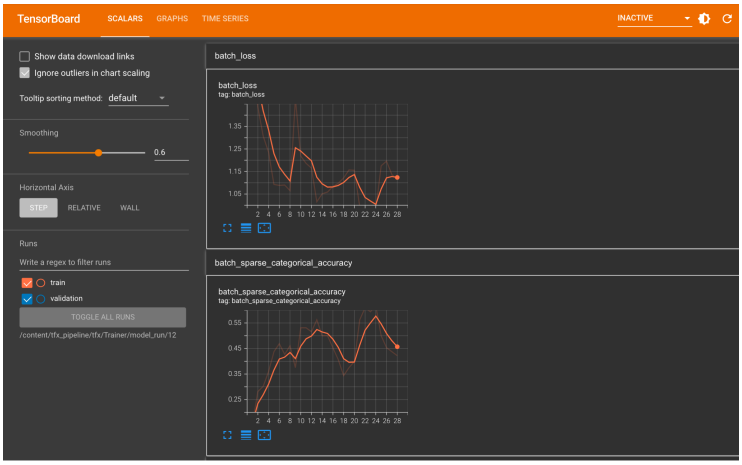
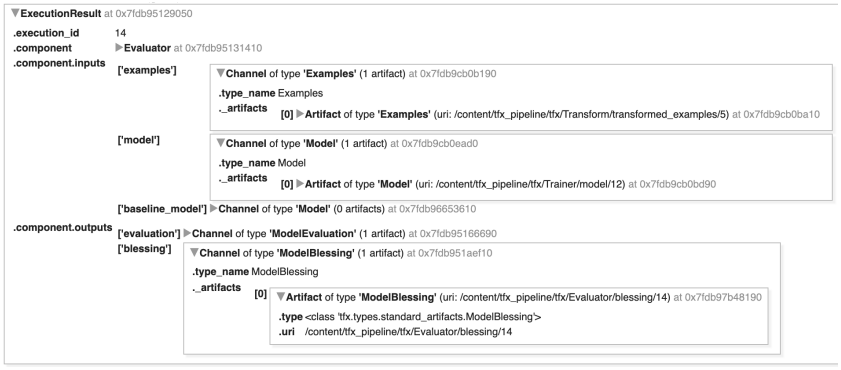


Table 8: Trainer output and tensorboard visualization

Evaluator

The Evaluator component performs analysis on the training results and the models created from the previous stage. A new model is compared to a baseline, such as the current model, in order to determine if it is "good enough" to push to production. Both models are evaluated on an evaluation dataset and their performance evaluated using set metrics. Whenever a model meets the developer's criteria, it is "blessed" (marked good), indicating to the Pusher that it can be pushed to production.



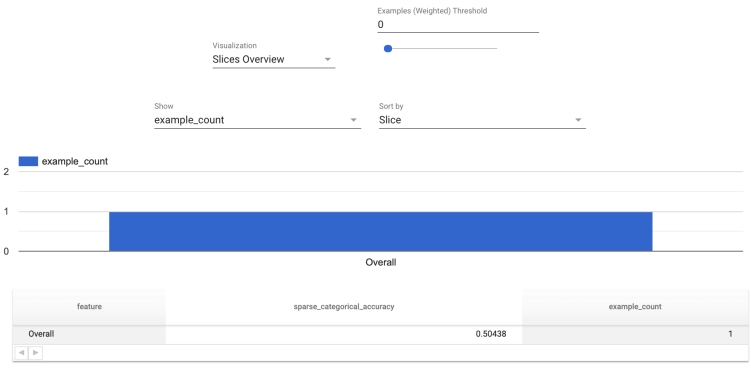


Table 9: Evaluator output

Pusher

The Pusher deploys the blessed model on the serving endpoint.

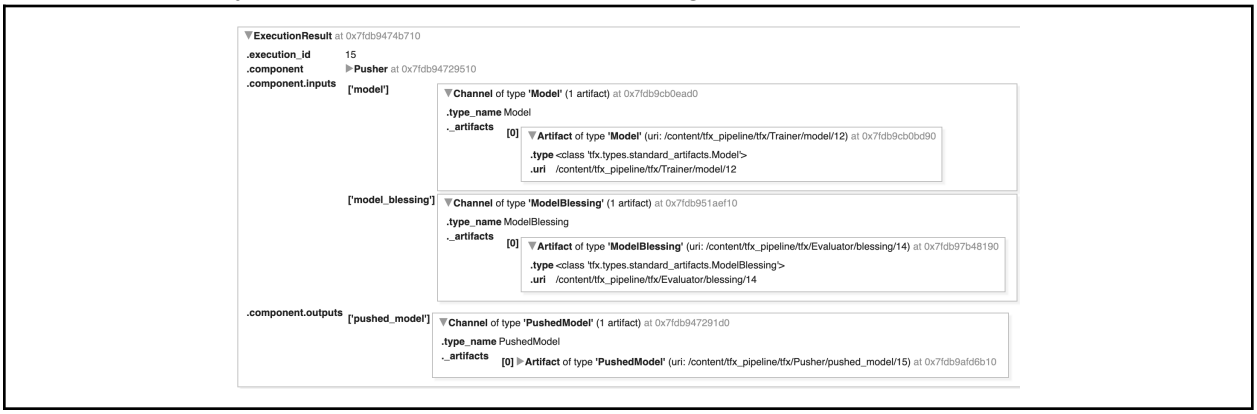
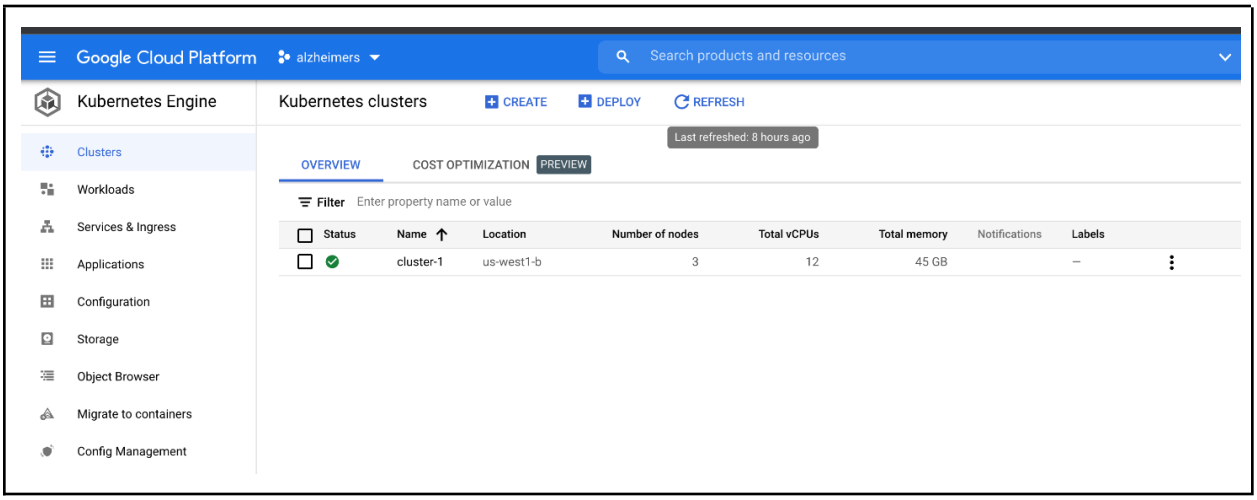


Table 10: Pusher output

The TFX pipeline is orchestrated using Kubeflow pipelines and is discussed in the subsequent section.

Kubeflow

Kubeflow incorporates many MLOps features which allows us to track and monitor our model pipeline runs. In addition, Kubeflow supports resource management to create Kubernetes clusters and cloud notebooks.



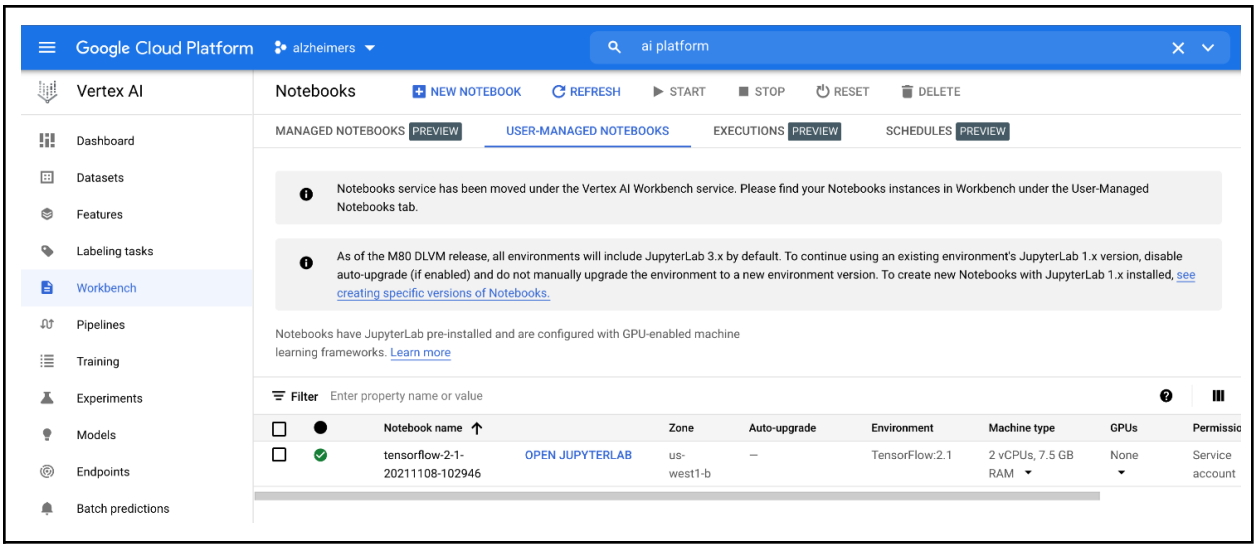
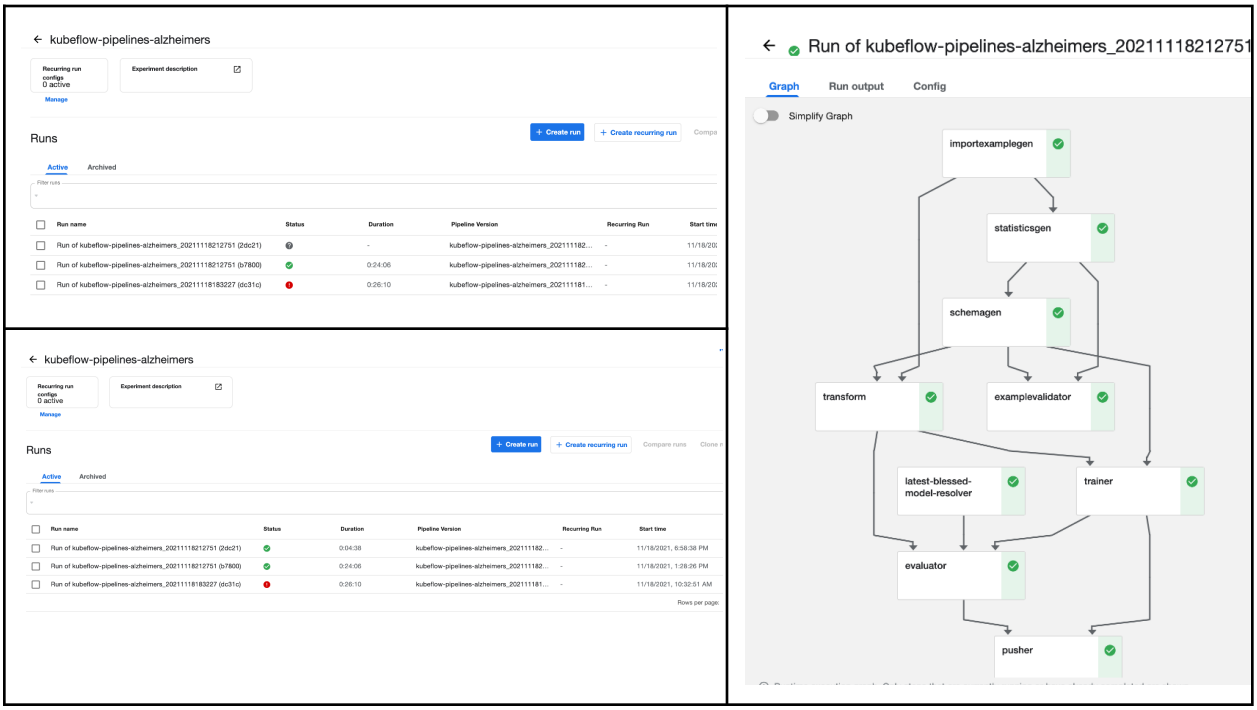


Table 11: Kubeflow resource management

Kubeflow also supports tracking of training jobs. By tracking jobs we can investigate the logs for any errors or issues which affect the training job. This also provides insights into the amount of cloud resources the job utilizes.



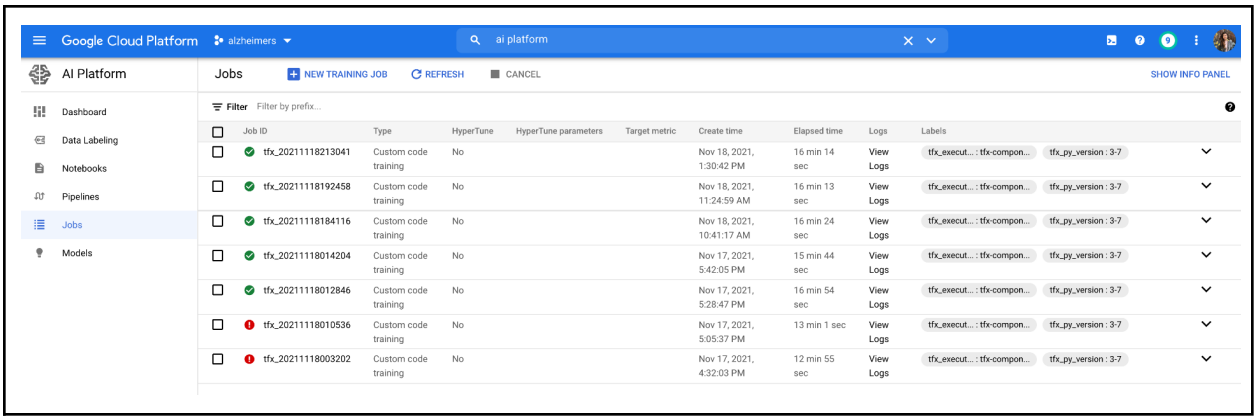


Table 12: Kubeflow job pending, job succeeded, Kubeflow pipeline graph, and AI Platform training job tracking

Kubeflow supports CI/CD with Git integration. This allows us to pull and push the latest changes from and to our master branch and deploy the most up to date model version to GCP AI Platform.

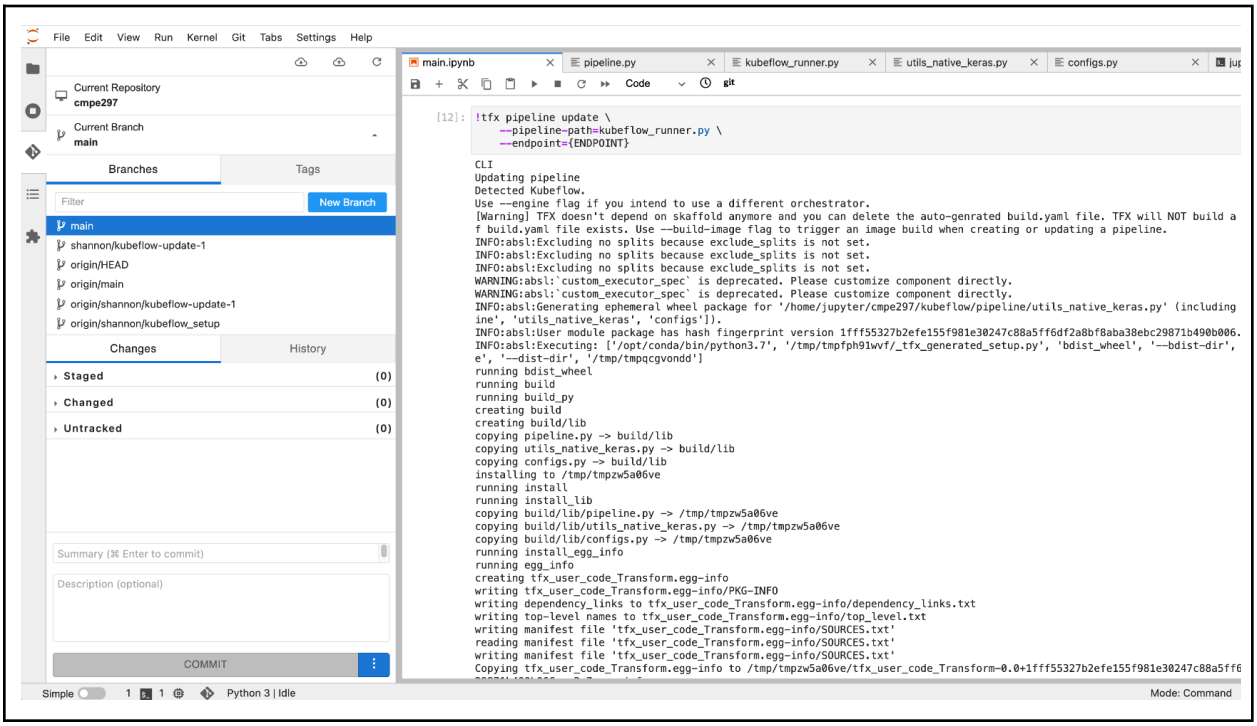


Table 13: Git integration with AI Platform to push to Kubeflow

Kubeflow and Google AI Platform support model version and model storage. This let's our deployed web application access the most up to date model pushed.

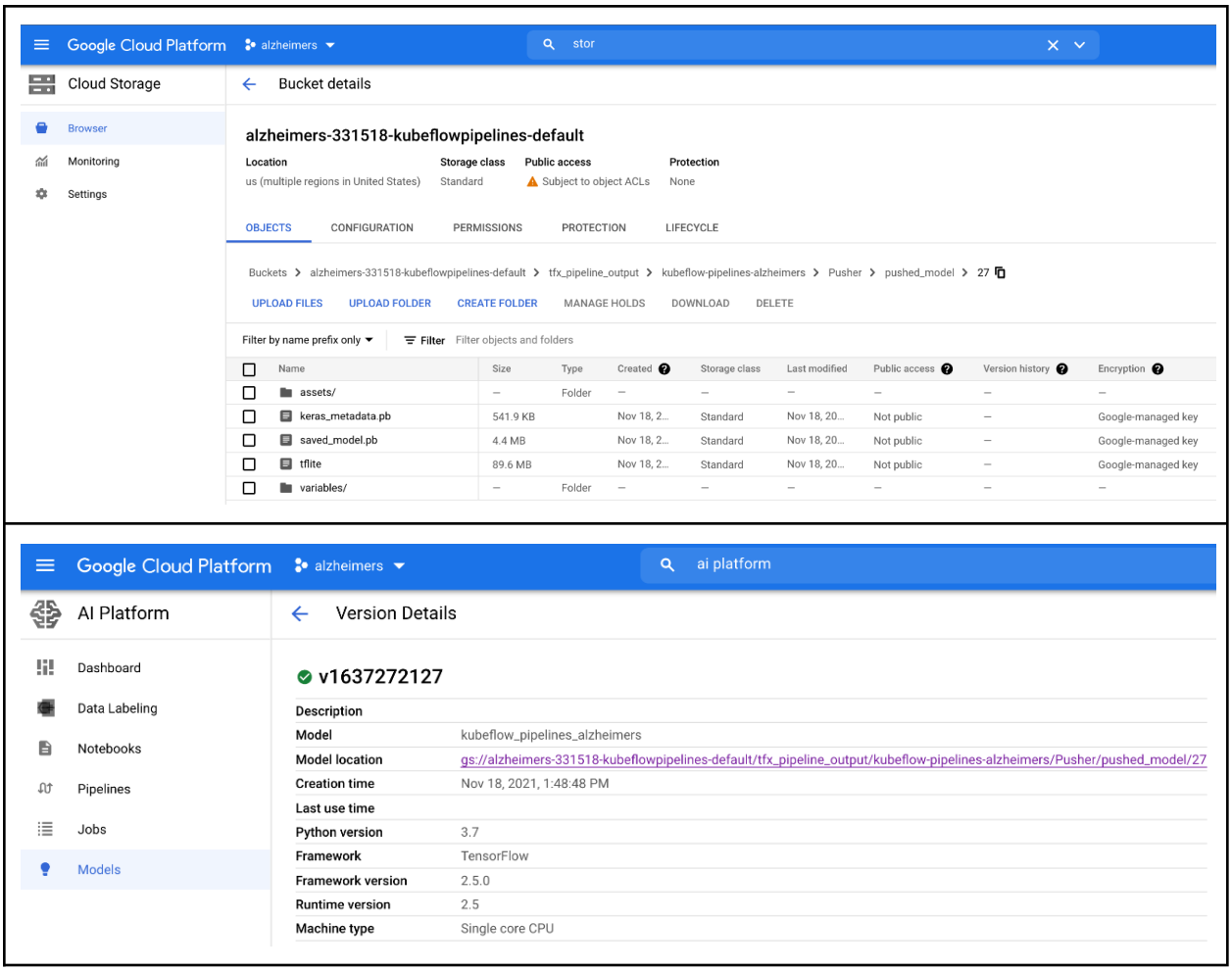


Table 14: Model storage and model versioning

Kubeflow additionally supports model gating through accuracy thresholds. We set our model thresholding low enough to allow any model updates to be pushed to the model store.

Table 15: Evaluator stage model gating view

Kubeflow also supports model drift detection, skew detection, and anomaly detection. Our input dataset did not have any skew or drift therefore no anomalies were detected.

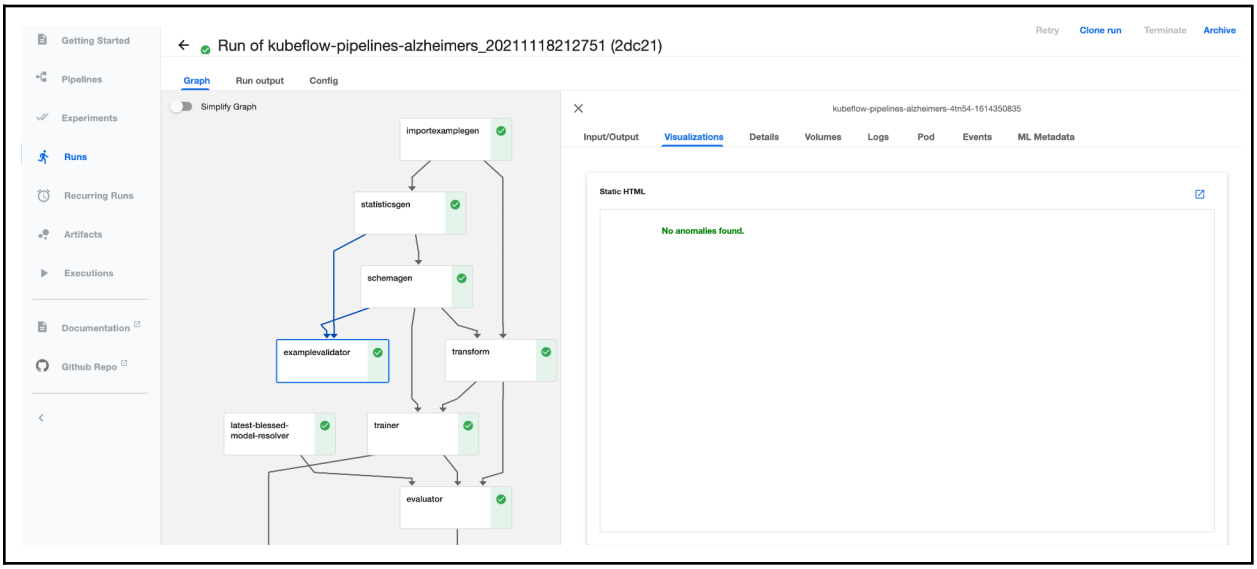


Table 16: Kubeflow drift, skew, anomaly detection. No anomalies were detected.

Flask App

We used Flask, Bootstrap and HTML to build the inference end point. Flask is a lightweight micro web framework which helps in rapid development of web applications. Bootstrap is a styling template library for HTML pages. The user can upload a brain MRI image through this interface and get a prediction value. An example classification result is captured in Table 17 below.

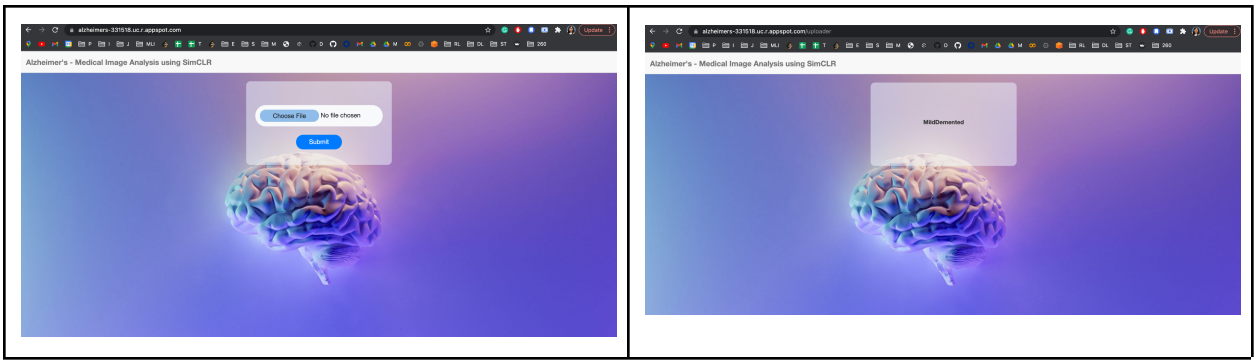


Table 17: 1) Interface to upload image. 2) Inference on the image

The front-end website is deployed in the Google Cloud App engine. App Engine is a fully managed, serverless platform for developing and hosting web applications at scale. The backend of the application is the latest model created and served by the Kubeflow pipeline. It is accessed using Google Client API Discovery library.

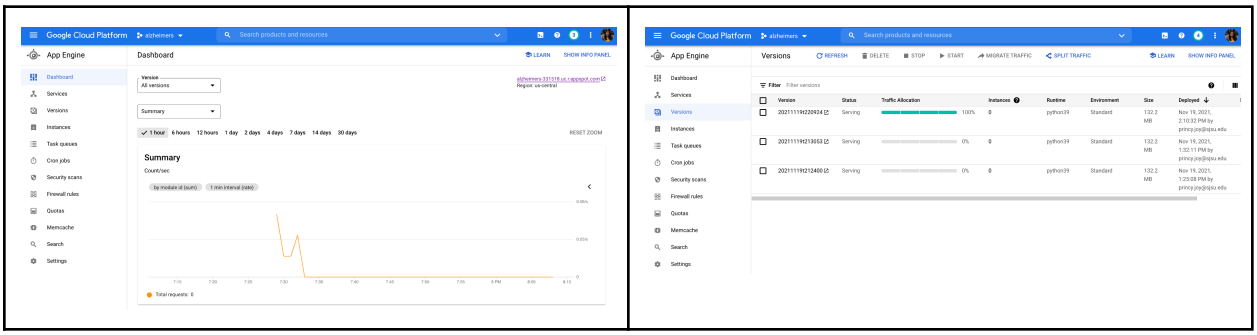


Table 18: 1) App engine dashboard. 2) Traffic serving endpoints

The traffic and the logs can be monitored using the App Engine dashboard. The deployed endpoints and the versions can also be viewed through the platform.

Conclusion

In this study, we successfully implemented and evaluated models for self-supervised and semi-supervised learning. We compared SimCLRv1 against SimCLRv2 when it was fine-tuned on our dataset as well as after performing the student-teacher distillation process. We found that SimCLRv1 produced the best result with a micro-AUC of 0.82 across the 4 classes.This was an interesting finding and could be because the Alzeimer's image dataset.had few samples. For future research, we would like to explore and perform additional comparative studies to evaluate the performance of SimCLRv1 and SIMCLRv2 against Supervised Learning methods.

References

[1] National Institute on Aging, "Alzheimer's Disease Fact Sheet," *National Institute on Aging*, May 22, 2019. https://www.nia.nih.gov/health/alzheimers-disease-fact-sheet.

[2] M. J. de Leon and S. DeSanti, “MRI and CSF studies in the early diagnosis of Alzheimer’s disease,” *Journal of Internal Medicine*, vol. 256, no. 3, pp. 205–223, Sep. 2004, doi: 10.1111/j.1365-2796.2004.01381.x.

[3] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, “A Simple Framework for Contrastive Learning of Visual Representations.” [Online]. Available: https://arxiv.org/pdf/2002.05709.pdf.

[4] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. Hinton, “Big Self-Supervised Models are Strong Semi-Supervised Learners.” [Online]. Available: https://arxiv.org/pdf/2006.10029.pdf.

[5] J. Shi, X. Zheng, Y. Li, Q. Zhang, and S. Ying, “Multimodal Neuroimaging Feature Learning With Multimodal Stacked Deep Polynomial Networks for Diagnosis of Alzheimer’s Disease,” *IEEE Journal of Biomedical and Health Informatics*, vol. 22, no. 1, pp. 173–183, Jan. 2018, doi: 10.1109/JBHI.2017.2655720.

[6] F. Li, D. Cheng, and M. Liu, “Alzheimer’s disease classification based on combination of multi-model convolutional networks,” *2017 IEEE International Conference on Imaging Systems and Techniques (IST)*, Oct. 2017, doi: 10.1109/ist.2017.8261566.

[7] S. Liang and Y. Gu, “Computer-Aided Diagnosis of Alzheimer’s Disease through Weak Supervision Deep Learning Framework with Attention Mechanism,” Sensors, vol. 21, no. 1, p. 220, Dec. 2020, doi: 10.3390/s21010220.

[8] L. Chen, P. Bentley, K. Mori, K. Misawa, M. Fujiwara, and D. Rueckert, “Self-supervised learning for medical image analysis using image context restoration,” Medical Image Analysis, p. 101539, Jul. 2019, doi: 10.1016/j.media.2019.101539.

[9] M. Y. Lu, R. J. Chen, and F. Mahmood, “Semi-supervised breast cancer histology classification using deep multiple instance learning and contrast predictive coding (Conference Presentation),” Medical Imaging 2020: Digital Pathology, Mar. 2020, doi: 10.1117/12.2549627.

[10] “Alzheimer’s Dataset (4 class of Images),” kaggle.com. https://www.kaggle.com/tourist55/alzheimers-dataset-4-class-of-images.