1.Approach

1. Tools and Framework

- **FastAPI** - developing the RESTful APIs with python web framework
- **MongoDB** - storing the training details and testing metrics.NoSQL nature
- **Streamlit** - designing a informative frontend for visualisation using python streamlit library
- **Docker** - containerize the application services to easier deployment

2. System Design

- Microservice architecture for application
- CNN Model for image classification to CIFAR-10 dataset
- CNN Model train and evaluate using CIFAR-10 dataset
- Store the training details and testing metrics store in the MongoDB database
- RESTful APIs develop using FastAPI for store and retrieve the metrics
- Metrics visualisation using Streamlit.

2. Design Decisions

- Selected CIFAR-10 10 classes available dataset to Model training and testing. Convolution Neural Network architecture using to model development.
- Tracked the useful details and metrics which are storing and retrieve from MongoDB. Data Store as a JSON.

- Developed meaningful API Endpoints for model training,evaluating,predicting and retrieving. More details in the Functional Document.
- Show model training, evaluating metrics variation overtime using line charts and graphs in Streamlit python.
- Build Docker images and containers to run 3 instances(Backend,frontend,mongoDB) in the same network.

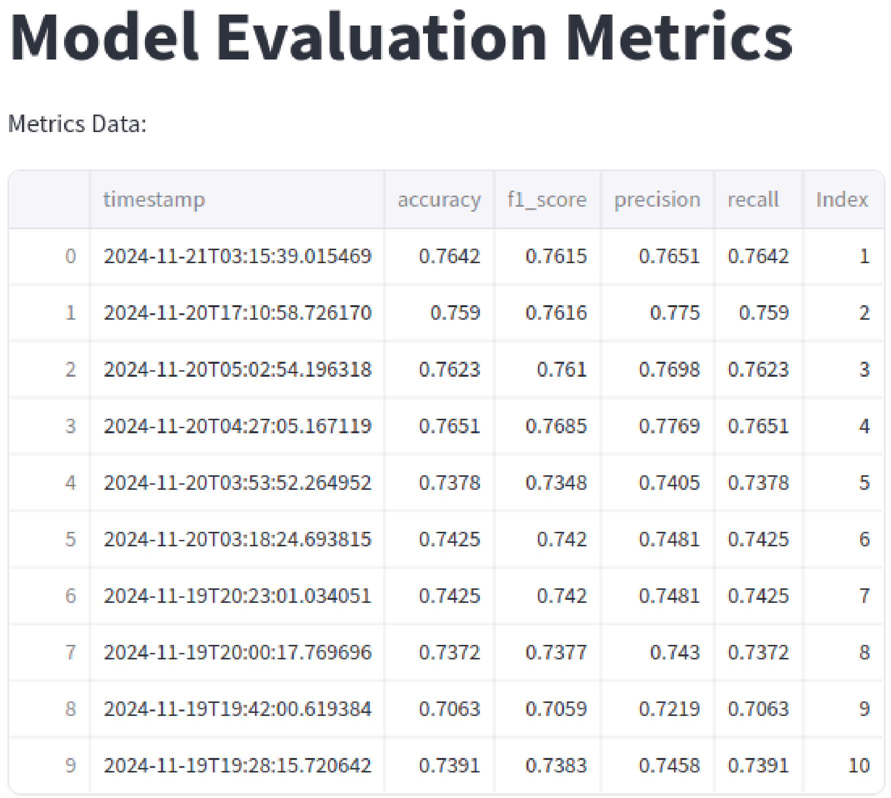
3.Challenges Faced

- **Dataset Selection -** There are so many datasets available publicly. CIFAR-10 dataset we can import from python tensorflow.keras.datasets library. CIFAR-10 dataset is preprocessed. Easy to apply for model training.
- **Web Framework Selection** - For microservice developed in python language. Easy to develop ML applications. FastAPI web framework selected to develop RESTful APIs.
- **Model Architecture Selection** - CIFAR-10 has 10 classes. CNN model selected for image classification.
- **Model improvement** - over fitting issues during the training. So added the dropout layers, kernel regularizer and Batch Normalization techniques to reduce the overfitting. Selected suitable 3x3 kernel for convolution. Used 3 convolution layers and dense layers to improve the accuracy and precision of the model. Epochs,batch size,validation split,learning rate are the parameters apply through the training API endpoint. According to the time constraint improved the model around 0.76 accuracy and precision.
- **Database Selection -** First selected SQlite database. But, it is SQL database. NoSQL databases are more suitable for data storage. We can add new Model metrics later. Not affect to store data in the NoSQL database. Stored the data as a JSON. Selected the MongoDB database for it.
- **Frontend** - According to the time constraint selected the streamlit python framework to develop the frontend. It was more easy to develop graphs and charts.

4.Potential Improved

- We can improve the model accuracy,precision and other metrics more.
- We can develop new API endpoints for authentication for ML application.
- We can improve predict inference API endpoint to upload the image.
- We can improve exception handling more.
- We can develop frontend more advanced to interactive with UI to users.

5.Metrics Graphs Model Training

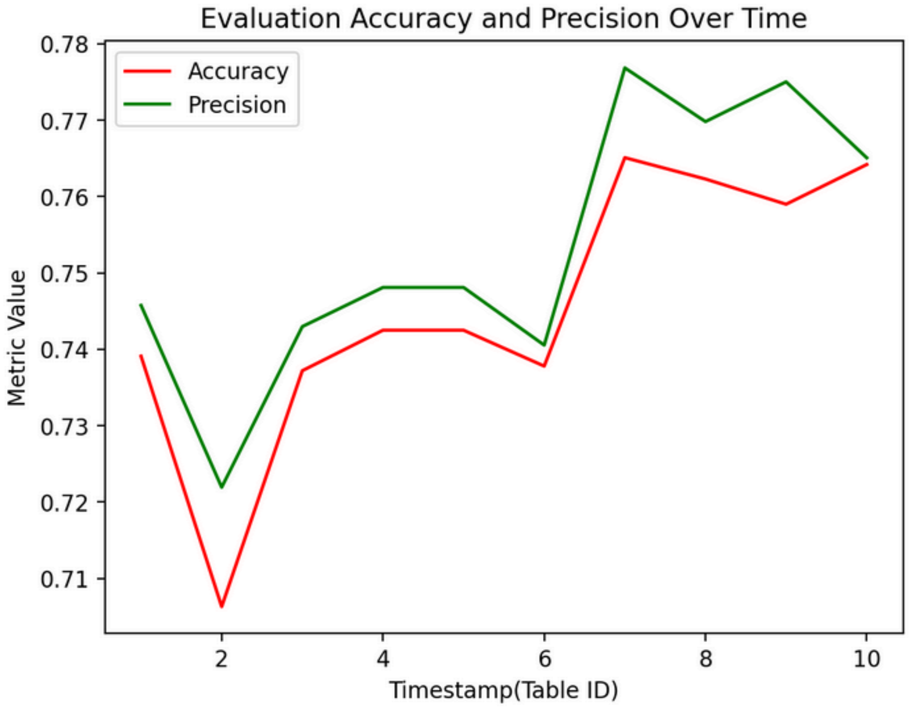


Model Evaluation Metrics

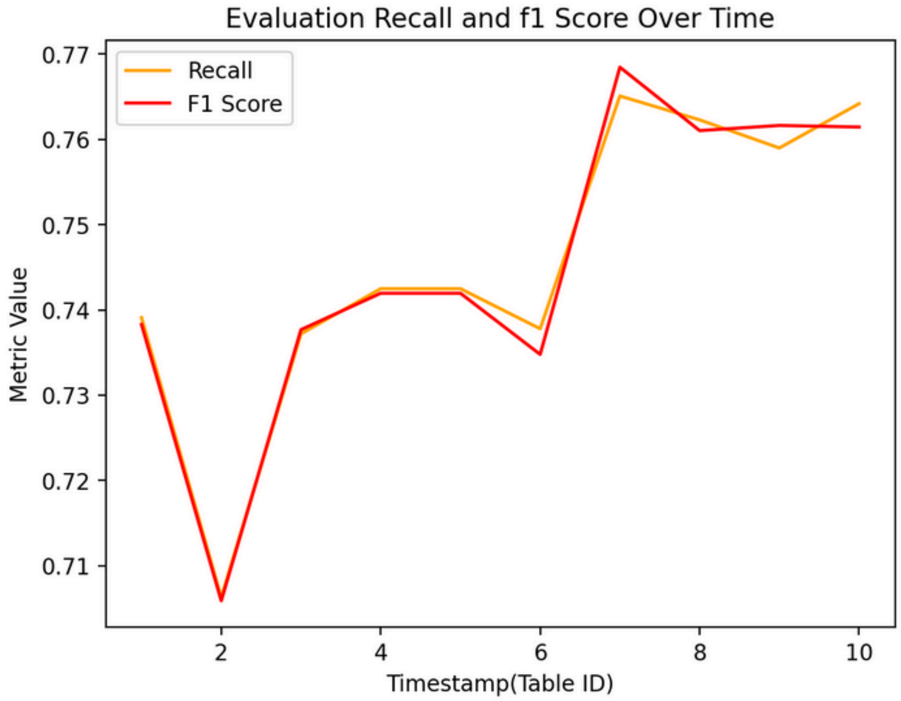
Metrics Data:

	timestamp	accuracy	f1_score	precision	recall	Index
0	2024-11-21T03:15:39.015469	0.7642	0.7615	0.7651	0.7642	1
1	2024-11-20T17:10:58.726170	0.759	0.7616	0.775	0.759	2
2	2024-11-20T05:02:54.196318	0.7623	0.761	0.7698	0.7623	3
3	2024-11-20T04:27:05.167119	0.7651	0.7685	0.7769	0.7651	4
4	2024-11-20T03:53:52.264952	0.7378	0.7348	0.7405	0.7378	5
5	2024-11-20T03:18:24.693815	0.7425	0.742	0.7481	0.7425	6
6	2024-11-19T20:23:01.034051	0.7425	0.742	0.7481	0.7425	7
7	2024-11-19T20:00:17.769696	0.7372	0.7377	0.743	0.7372	8
8	2024-11-19T19:42:00.619384	0.7063	0.7059	0.7219	0.7063	9
9	2024-11-19T19:28:15.720642	0.7391	0.7383	0.7458	0.7391	10

Accuracy and Precision



Recall and F1 Score



6.Links

- Github link -https://github.com/Lakshan-Jayaweera285/UCD_ML_Application_Project
- Dataset cifar10 - https://keras.io/api/datasets/cifar10/
- FastAPI - https://fastapi.tiangolo.com/
- Streamlit - https://streamlit.io/