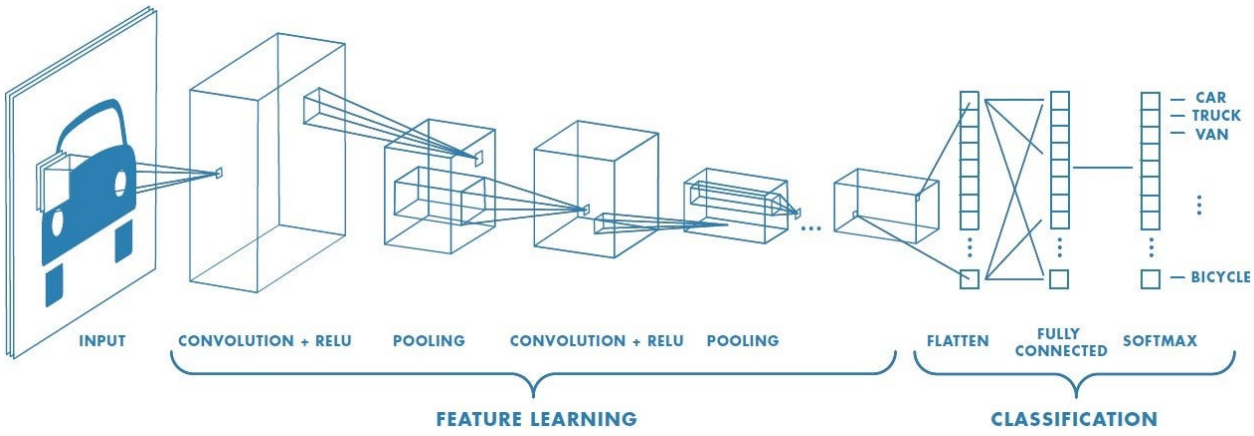
ML4Sci Task Report

I am delighted to share with you that I have successfully completed all three tasks assigned to me as part of the ML4Sci evaluation test. Enclosed in this document is the report.pdf file that contains a comprehensive overview of my work.

Table of contents:

Common Task 1. Electron/photon classification



source

In this task, the objective was to classify Electron and Photon based on the input data consisting of 32x32 matrices with 2 channels for hit energy and time. For this purpose, I opted for a convolutional neural network (CNN) since it has several advantages over other models. The main advantage of CNNs is that they are spatially invariant, meaning that they can detect the same patterns regardless of their position in the input image. Moreover, CNNs take care of the time component by sliding over the input data one-by-one, which makes them particularly suitable for processing time-series data. Another advantage of CNNs is that they are generally lightweight, which makes them fast and efficient for large-scale applications.

Note:
I trained the CNN on only 0.032% of the entire dataset, which amounts to 16,000 samples with 800 photon examples and 800 electron examples due to limited computational resources.

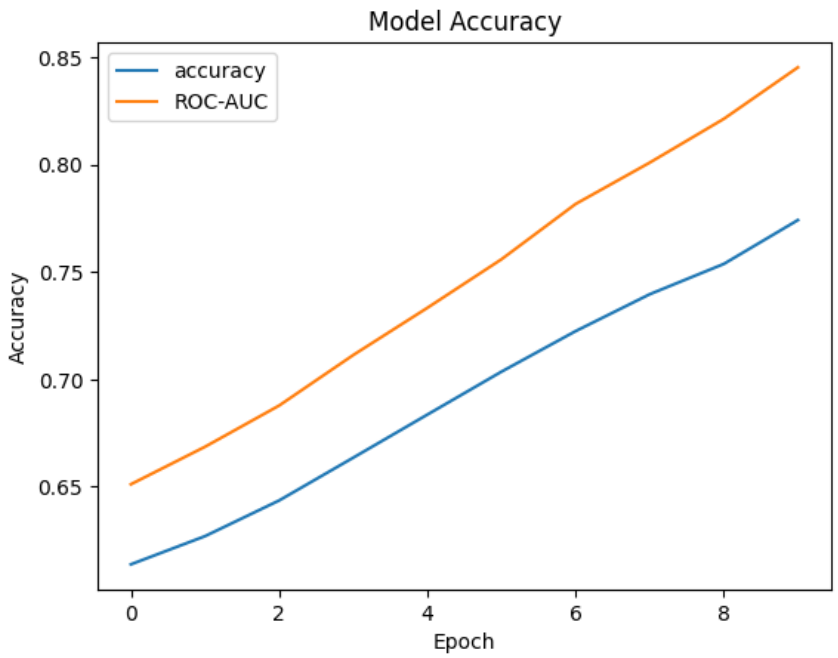
The Model Architecture

```
model = tf.keras.Sequential([
    tf.keras.layers.Conv2D(filters=32, kernel_size=(3,3), activation='relu'),
    tf.keras.layers.Conv2D(filters=64, kernel_size=(3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D((2,2)),
    tf.keras.layers.Conv2D(filters=64, kernel_size=(3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D((2,2)),
    tf.keras.layers.Conv2D(filters=64, kernel_size=(3,3), activation='relu'),
    tf.keras.layers.Conv2D(filters=64, kernel_size=(3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D((2,2)),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(units=64, activation='relu'),
    tf.keras.layers.Dense(units=1, activation='sigmoid')
])
```

The proposed architecture is composed of multiple convolutional neural network (CNN) layers, followed by MaxPooling and densely connected layers. Rectified Linear Unit (ReLU) activation function has been utilized throughout the network.

Result

```
Epoch 1/10
250/250 [==============================] - 79s 317ms/step - loss: 0.6541 - accuracy: 0.6138 - auc: 0.6511
Epoch 2/10
250/250 [==============================] - 79s 314ms/step - loss: 0.6446 - accuracy: 0.6269 - auc: 0.6686
Epoch 3/10
250/250 [==============================] - 79s 316ms/step - loss: 0.6326 - accuracy: 0.6436 - auc: 0.6878
Epoch 4/10
250/250 [==============================] - 79s 316ms/step - loss: 0.6168 - accuracy: 0.6635 - auc: 0.7113
Epoch 5/10
250/250 [==============================] - 79s 316ms/step - loss: 0.6000 - accuracy: 0.6836 - auc: 0.7333
Epoch 6/10
250/250 [==============================] - 79s 315ms/step - loss: 0.5811 - accuracy: 0.7036 - auc: 0.7559
Epoch 7/10
250/250 [==============================] - 79s 317ms/step - loss: 0.5572 - accuracy: 0.7224 - auc: 0.7817
Epoch 8/10
250/250 [==============================] - 79s 314ms/step - loss: 0.5370 - accuracy: 0.7396 - auc: 0.8009
Epoch 9/10
250/250 [==============================] - 79s 317ms/step - loss: 0.5141 - accuracy: 0.7538 - auc: 0.8213
Epoch 10/10
250/250 [==============================] - 79s 317ms/step - loss: 0.4837 - accuracy: 0.7742 - auc: 0.8453
```



The model was trained for 10 epochs, and it achieved a **ROC AUC score of 0.84**, which exceeds the required ROC AUC score of 0.80.

Common Task 2. Deep Learning based Quark-Gluon Classification

In this task, the objective was to classify Quark and Gluon based on the input data consisting of 125x125 matrices with 3 channels.

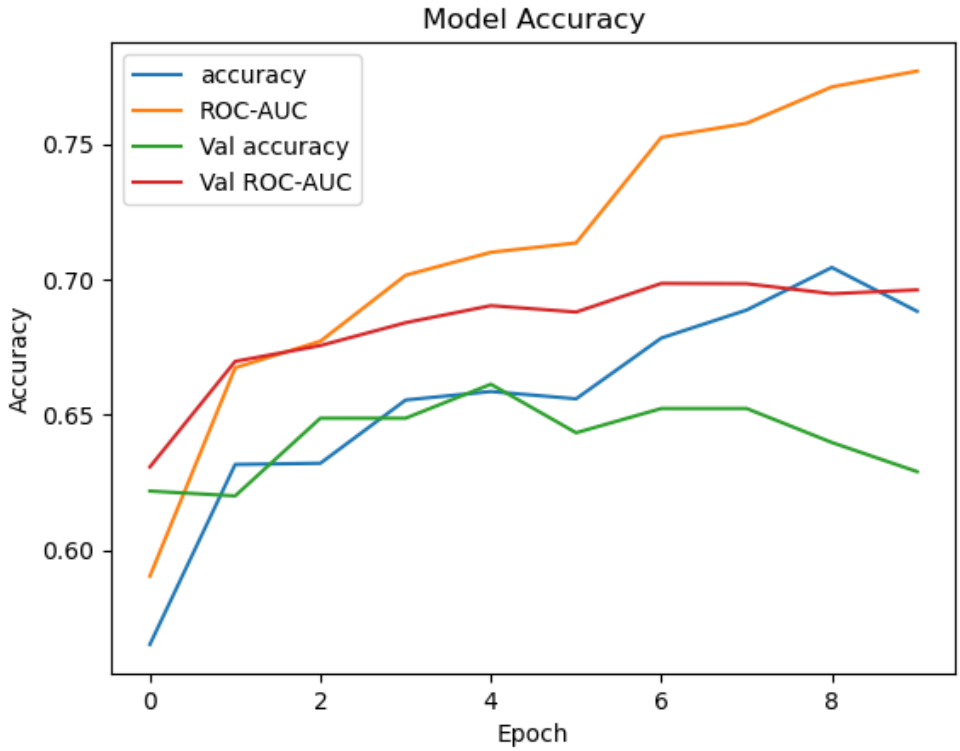
Initial Experiment

In the beginning, I attempted to train the data using the same model as presented in Task 1. However, the model was significantly overfitting the data.

Final Solution

Eventually, I arrived at the final solution by iterating through several models and hyperparameters. The chosen model consisted of a simple CNN with a single filter and a dropout rate of 0.5. This model achieved the best performance on the validation set with minimal overfitting.

Result

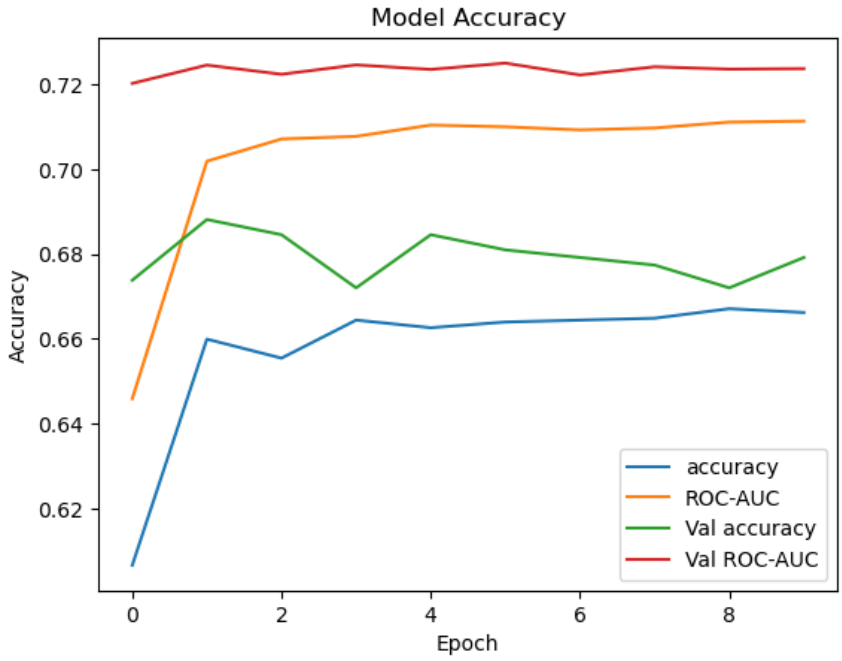


Training ROC AUC score: **0.77**
Validation ROC AUC score: **0.70**

The model was still significantly overfitting.

Attempt 2

For the second attempt, I utilized 'pt' and 'm0' to predict whether a given particle was a quark or gluon. However, unlike the previous method, this approach did not involve the use of a convolutional neural network. Despite this, the results obtained were significantly better than those from the first attempt.



Training ROC AUC Score: **0.71**
Validation ROC AUC Score: **0.72**

This method does not overfit.

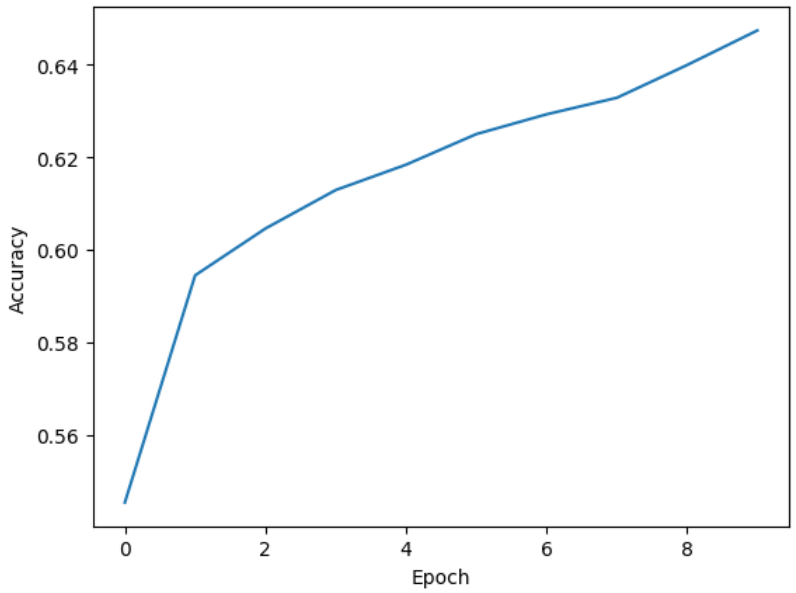
Task 3. Vision Transformers for End-to-End Particle Identification with the CMS Experiment

For Task 3, we aimed to perform the same classification as in Task 1, but using a vision transformer. While I had experience working with vision transformers before, this was my first time using one on scientific data. As a result, the outcome was completely unpredictable to me.

The Model Architecture

To process the input data using the Vision Transformer, I first created a patch layer to extract patches from the input data. This was followed by a PatchEncoder layer, which added positional embeddings to the patches. The resulting tensor was then passed through a transformer encoder. Finally, the encoded tensor was fed into a dense layer for classification.

Result



I stopped the training after 10 epochs at 65% accuracy. But from the accuracy curve above, it is apparent that the model's performance continues to improve with more epochs of training. This suggests that if we continue training the model for more epochs, we can expect further improvement in its performance.

Additional Information & Final thoughts

I am extremely passionate about machine learning and have been working on various projects for the past few years. I love exploring new algorithms, models, and techniques to solve problems with the help of machine learning.

I am proud to say that I am a **TensorFlow Developer certified**, which demonstrates my strong grasp on TensorFlow. This certification has helped me understand various advanced concepts of machine learning, which have been invaluable in my work.

In addition to my work, I also have a YouTube channel with over 1.9K subscribers and more than 220K total views. On my channel, I share my knowledge and insights about machine learning with the wider community. I enjoy creating content on complex topics in a simple and easy-to-understand way.

I have already created various open source projects, some of which have been quite popular in the community (this, this and this). These projects have not only helped me improve my skills but have also enabled me to contribute to the larger machine learning community.

I am eager to learn anything new and I understand that my solution might not be perfect. Therefore, I request my mentors to provide constructive feedback and forgive me for any mistakes that I might have made. I believe that there is always room for improvement, and I am committed to learning and growing as a machine learning practitioner.

Sincerely,
Pritish Mishra