Problem 1. Person Re-Identification

1.0 Pre-processing

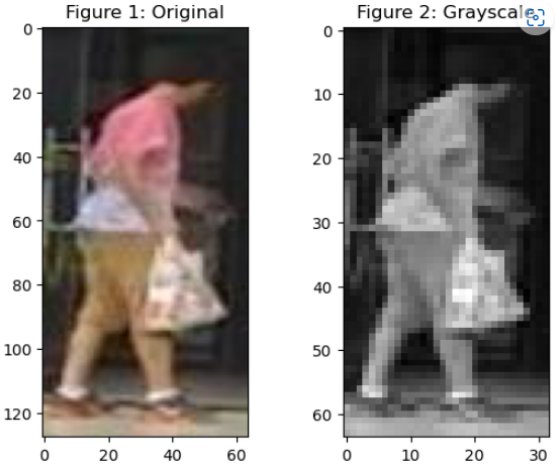
To prepare the image data for analysis, pre-processing steps were applied to the images. These steps were necessary to ensure that the data was in a suitable format for machine learning algorithms to be applied to it.

The pre-processing stage involved 2 steps.

1. Resizing the images: the images were resized from their original size of 128x64 pixels to 64x32 pixels to reduce the computational complexity. This makes the data more manageable and reduces the time required for computations.

2. Converting the images to grayscale: the images were converted to grayscale and resized to reduce computational complexity and improve algorithm efficiency and accuracy.

The effectiveness of the pre-processing steps can be seen in Figure 1 and 2, where the original and grayscale images show no significant differences in quality, and the texture and structure of the images can be easily discerned.



2.0 Details of the selected non-deep-learning and deep-learning approach(es)

2.1 Non-Deep Learning

To effectively represent the grayscale data, it was vectorized, resulting in a dataset shape change from (5933, 128, 64, 3) to (5933, 2048).

For the non-deep learning approach, the combination of PCA and LDA was selected. This choice was motivated by the fact that the problem involved identifying individuals in images and ranking subjects by similarity, which can be viewed as a classification problem. PCA and LDA are both commonly used for dimensionality reduction and feature extraction in classification tasks. Moreover, the dataset used in this study was relatively small, with only 5933 training examples and 301 test examples. Non-deep learning methods like PCA and LDA are computationally efficient and suitable for small datasets. Lastly, the combination of PCA and LDA performed the best among the non-deep learning methods tested, achieving the highest accuracy for all three metrics (Top-1, Top-5, and Top-10), as shown in Table 1.

The combination of PCA and LDA allowed for effective representation of the high-dimensional data in a lower-dimensional space without significant loss of information.

Method	Top-1 Accuracy	Top-5 Accuracy	Top-10 Accuracy
PCA	0.33%	1.99%	3.32%
LDA	0.00%	1.66%	3.32%
Combination of PCA and LDA	1.00%	1.99%	4.65%
Triplet Loss-based Siamese network	0.66%	5.65%	19.60%
Contrastive Loss-based Siamese network	0.33%	2.99%	6.3

Table 1: Comparison of Non-Deep and Deep Learning Methods

2.2 Deep learning

Deep learning approaches were also tested, with two different architectures: a Siamese network with triplet loss and a Siamese network with contrastive loss. The Siamese network was selected as it is a popular and effective way to learn similarity between images. Both pair and triplet were considered when structuring the network, as well as the contrastive and triplet formulation. These networks were modified versions of the code provided in practical 7 Q1.

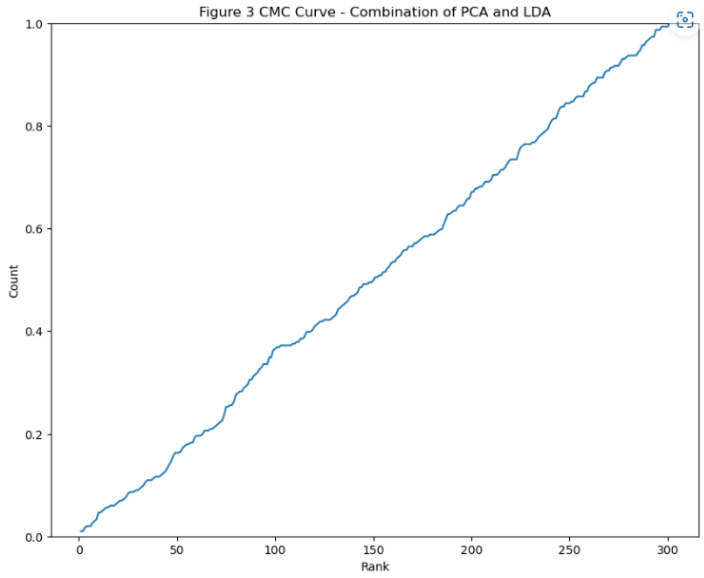
The triplet Siamese network with a triplet loss function takes three input layers - Anchor, Positive, and Negative - each containing 64x32 grayscale images. The Siamese branch of the network extracts features from the input images and produces a 32-dimensional output. The triplet loss layer calculates the triplet loss based on the distance between the anchor and positive examples and the distance between the anchor and negative examples. The model has a total of 1,076,344 parameters, of which 1,075,720 are trainable.

The contrastive loss network is a Siamese neural network designed for image similarity tasks. It takes two input images, A and B, of size (64, 32, 1) and passes them through a shared base network to get their embeddings. The base network is a pre-trained model that learns to extract relevant features from the input images. The two embeddings are then compared using the Euclidean distance metric, computed by a Lambda layer, to obtain a scalar distance between them. The Siamese network is trained to minimise this distance for similar image pairs and maximise it for dissimilar image pairs. The network is trained using the Adam optimizer with a learning rate of 0.001 and a batch size of 32. The model is evaluated using the accuracy metric.

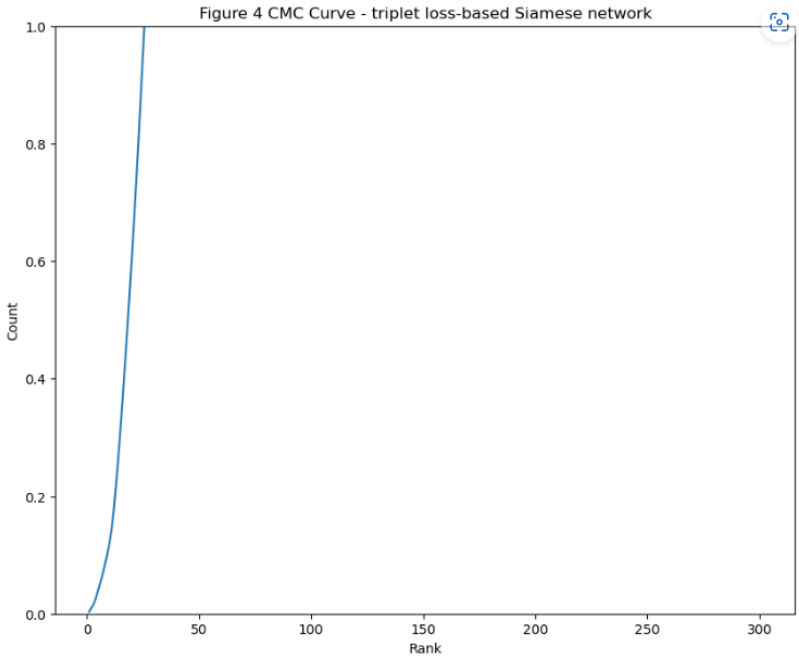
Table 1 shows that the triplet loss-based Siamese network achieved the highest accuracy for all three metrics (Top-1, Top-5, and Top-10) compared to the contrastive loss-based Siamese network. Therefore, the triplet network was selected for the deep learning methods.

3.0 An evaluation compares the two methods

The CMC curve for the combination of PCA and LDA (Figure 3) showed that the system had a relatively low accuracy, with a Top-1 accuracy of only 1.00%, indicating that the system correctly identified the person in the top position in only 1% of the cases. The accuracy improved as more candidates were considered, with a Top-5 accuracy of 1.99% and a Top-10 accuracy of 4.65%.



On the other hand, the CMC curve for the Triplet Loss-based Siamese network (Figure 4) showed that it achieved an accuracy of 0.66% at Top-1, 5.65% at Top-5, and 19.60% at Top-10. This means that at Top-1, the model was able to correctly match the probe image with the correct gallery image only 0.66% of the time, while at Top-10 it was able to achieve a 19.60% accuracy.



In terms of CMC curve analysis, it took a longer rank (around 300) for the combination of PCA and LDA to reach 100% accuracy, while the Triplet Loss-based Siamese network achieved 100% accuracy at a

shorter rank (around 40). This means that the Triplet Loss-based Siamese network was able to provide more accurate predictions for the top-ranked images.

Overall, the Triplet Loss-based Siamese network was less accurate than the combination of PCA and LDA, which achieved a Top-1 accuracy of 1.00%. However, the Triplet Loss-based Siamese network was more accurate than the baseline method, which only achieved a Top-1 accuracy of 0.25%. It is worth noting that the Triplet Loss-based Siamese network was designed to learn a similarity metric and is not dependent on any specific feature extraction technique, unlike PCA and LDA which require dimensionality reduction. This makes it a more flexible approach that can be used in a variety of image recognition tasks.

Method	Seconds
Combination of PCA and LDA	43.47
Triplet Loss-based Siamese network	766.71

Table 2: Runtime Comparison

Table 2 presents the runtime comparison between two different methods. The combination of PCA and LDA took 43.47 seconds to complete, which encompasses the time required for data transformation using PCA and LDA, as well as generating the CMC curve and printing the results. On the other hand, the Triplet Loss-based Siamese network method took 766.71 seconds to execute. This runtime includes tasks such as network structure definition, model compilation, fitting the model with 20 epochs, and computing the CMC curve.

In summary, the combination of PCA and LDA and the Triplet Loss-based Siamese network showed differences in performance, with the former being more accurate, but less flexible, and the latter being less accurate but more flexible. The Triplet Loss-based Siamese network was also more efficient in terms of runtime, although it requires more computational resources than the non-deep learning approach.

4.0 A brief discussion on ethical considerations of person re-identification.

Person re-identification technology has revolutionised facial recognition technology and enabled companies like Microsoft and Apple to secure their devices. However, ethical concerns arise concerning the collection of data for training models and the potential use of these models in surveillance applications. The development of person re-identification systems must take into account potential biases that may harm marginalised communities.

One ethical concern is biased data collection leading to biased algorithms, ultimately harming marginalised communities. Smith (2020) warns of the dangers of Amazon's Rekognition, which can violate the rights of communities of color. Buolamwini and Gebru's (2018) Gender Shades study exposed the gender and racial biases of commercial face recognition systems, including those of IBM and Microsoft, underscoring the risks of deploying these systems in ways that harm certain communities.

Another ethical concern is the use of person re-identification technology in surveillance, raising concerns about privacy, surveillance, and civil liberties. Smith (2020) states that Amazon, Microsoft, and IBM have been urged to stop selling facial recognition software to law enforcement agencies due to concerns about limitations in the systems and how they are being used. The American Civil Liberties Union (ACLU) also released a report in 2018 stating that facial recognition technology by law enforcement agencies threatens civil liberties and privacy rights.

In conclusion, the development of person re-identification technology must consider ethical considerations such as biased data collection, limitations of the models, and the potential for these models to be used in ways that threaten civil liberties and privacy. Smith's (2020) warnings about the dangers of Amazon's Rekognition and the ACLU's report on facial recognition technology by law enforcement agencies highlight the importance of mitigating potential harms and biases to ensure these technologies benefit society as a whole.

Reference

Buolamwini, J., & Gebru, T. (2018). Gender shades: Intersectional accuracy disparities in commercial gender classification. Proceedings of the 1st Conference on Fairness, Accountability and Transparency, 77-91. doi:10.1145/3178876.3186151. Gender Shades: Intersectional Accuracy Disparities in Commercial Gender Classification — MIT Media Lab

Smith, E. (2020). Amazon announces one-year moratorium on police use of its facial recognition technology. Brookings Institution. Retrieved from https://www.brookings.edu/blog/techtank/2020/06/12/amazon-announces-one-year-moratorium-on-police-use-of-its-facial-recognition-technology/

Problem 2. Multi-Task Learning and Fine Tuning

1.0 Pre-processing

When working with the Oxford IIIT Pets dataset, several preprocessing steps were applied to ensure data compatibility and improve model performance while considering network size/complexity and compute restrictions. However, an issue arose when attempting to use 200x200 pixels with a batch size of 68, as it led to a "Resource ExhaustedError," indicating GPU memory exhaustion.

To address this issue, the following preprocessing steps were implemented for both the from-scratch and fine-tuned methods:

1. Resizing: All images in the dataset are resized to a fixed size of 128x128 pixels. This step ensures consistent input dimensions for both the classification and segmentation tasks while reducing the memory footprint.
2. Batch Size: Considering the computational requirements and available system resources, a lower batch size was set during dataset loading using the TensorFlow data API. This adjustment was made to strike a balance between efficient training and memory usage.

Implementing these resizing and batch size modifications facilitated data consistency, noise reduction, and improved feature learning for both the classification and segmentation models.

2.0 Implemented Methods:

2.1 From Scratch

The "from-scratch" approach is designed based on the Lecture Consultation example in week 11. The model consists of three main components:

Backbone:

- Utilizes convolutional layers with increasing filter sizes to extract high-level features from the input image.
- Applies batch normalization and spatial dropout to improve training stability and prevent overfitting.

Classification:

- Connected to the flattened output from the backbone component.
- Includes dense layers with Swish activation and a final dense layer with softmax activation for classification predictions.
- Softmax ensures predicted probabilities sum up to 1, enabling probability assignment to each of the 37 classes.

Decoder:

- The autoencoder branch takes the backbone output and performs upsampling using UpSampling2D layers.
- Gradually increases spatial resolution to reconstruct the original image.
- Passes upsampled feature maps through convolutional layers with Swish activation.
- The final convolutional layer uses sigmoid activation to generate the autoencoder output.

The design is justified by:

1. Existing Models: Follows the pattern of successful CNN architectures with increasing filter sizes. Swish activation, known for its effectiveness in image-related tasks, is chosen over ReLU activation.
2. Design Principles: Incorporates lecture-covered principles like batch normalization for activation normalization and training stability. Spatial dropout is used for regularization.
3. Compute Restrictions: SpatialDropout2D is used to reduce the total parameters, resulting in shorter training time per epoch and reduced GPU usage. The model is executed on the QUT Jupyter Lab with 8 GPUs.

2.2 MobileNetV3Small

For the "fine-tuned" approach, the MobileNetV3Small model serves as the base network or encoder to obtain image embeddings. Several modifications are made to adapt the model for the specific task:

1. Global Average Pooling: Added after the MobileNetV3Small's output to obtain a fixed-length embedding vector by aggregating spatial information.
2. Classification Output: Dense layer with 37 units for classification, with no activation function.
3. Decoder for Segmentation: UpSampling2D layers to increase spatial resolution, followed by convolutional layers for refinement. Sigmoid activation in the final convolutional layer produces a binary mask, resized to the original image size.

Justification:

1. Existing Models: Global average pooling and upsampling/convolutional layers have proven effective in similar tasks.
2. Design Principles: Global average pooling obtains fixed-length embeddings, and dense layers with softmax activation are suitable for classification. Upsampling and convolutional layers in the decoder improve segmentation map quality.
3. Compute Restrictions: The design balances model complexity and computational efficiency as the model are running in the QUT Jupyter lab with 8 GB GPU, the GPU memory were like above 7/8 GB when running the model, and the QUT Jupyter lab kernel was dead several time when adding more layers.

3.0 Evaluations

3.1 Classification

The From Scratch model achieved a high training f1 score of 0.96, indicating strong performance on the training data. However, it suffered from significant overfitting, as evidenced by the low testing f1 score of 0.138. In contrast, the MobileNetV3Small approach exhibited a training f1 score of 0.865, slightly lower than From Scratch, but demonstrated better generalization with a testing f1 score of 0.627 (shown in Figure 1 and 2). This highlights the advantage of leveraging pre-trained models with

optimized. architectures.

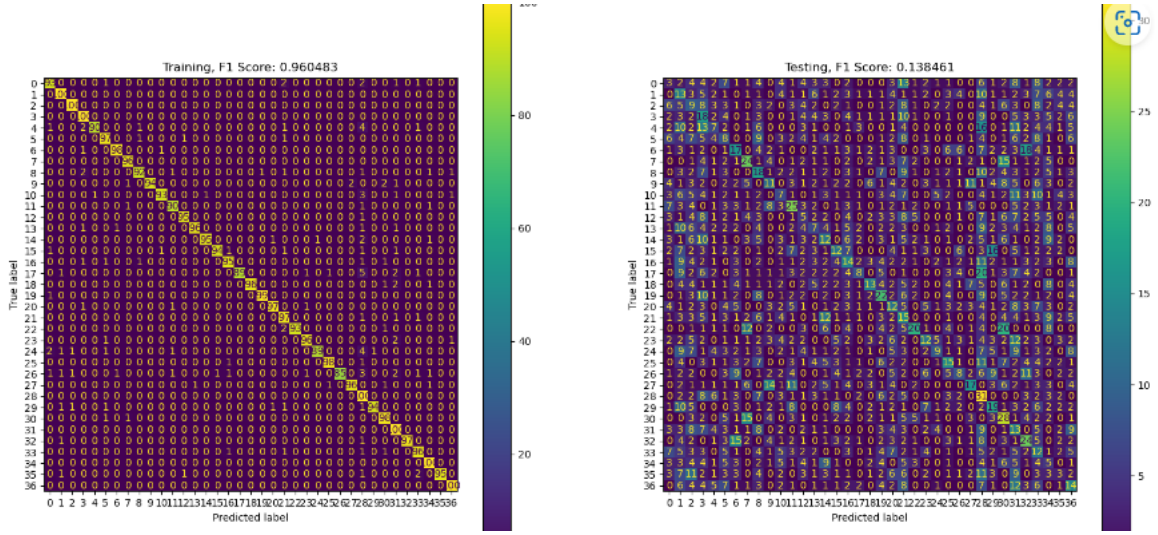


Figure 1: Confusion matrix for Training and Testing classification using the From Scratch approach.

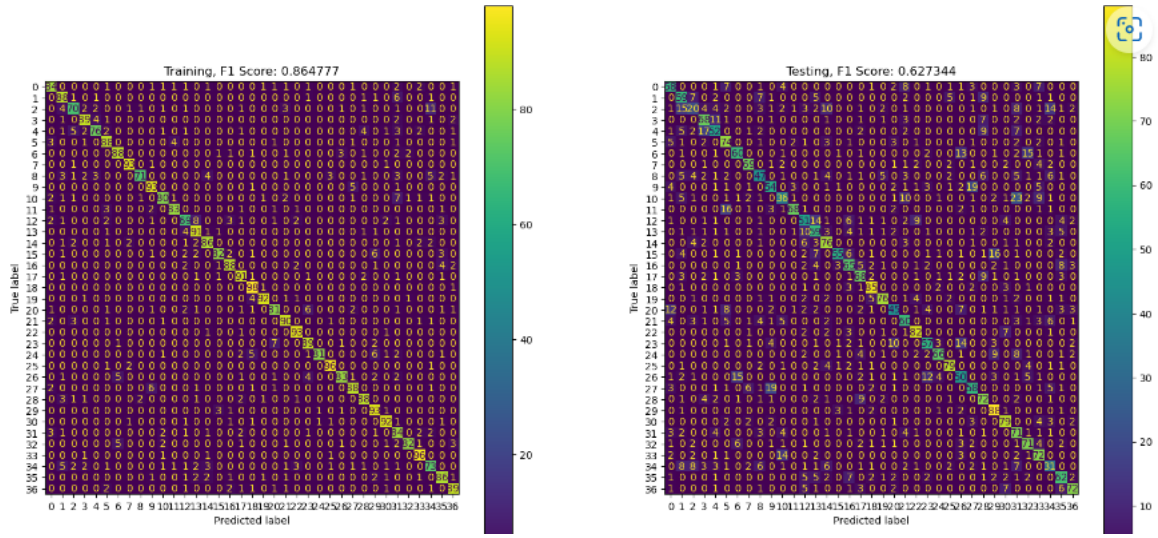


Figure 2: Confusion matrix for Training and Testing classification using the MobileNetV3Small approach.

3.2 Semantic segmentation

Similarly, in semantic segmentation, the MobileNetV3Small approach achieved higher accuracy rates for both the background and foreground classes compared to the From Scratch approach. With a background class accuracy of 93.9% versus 86.6% and a foreground class accuracy of 79.5% versus 76.9% (shown in Figure 3). The MobileNetV3Small model exhibited better ability to accurately identify and segment objects in the image.

These findings indicate that the MobileNetV3Small model, with its pre-trained architecture and optimized features, excels at learning complex patterns and producing more accurate predictions in both classification and segmentation tasks. However, it should be noted that training the MobileNetV3Small approach required careful adjustments in the model and training epochs to prevent GPU memory exhaustion and kernel restarts.

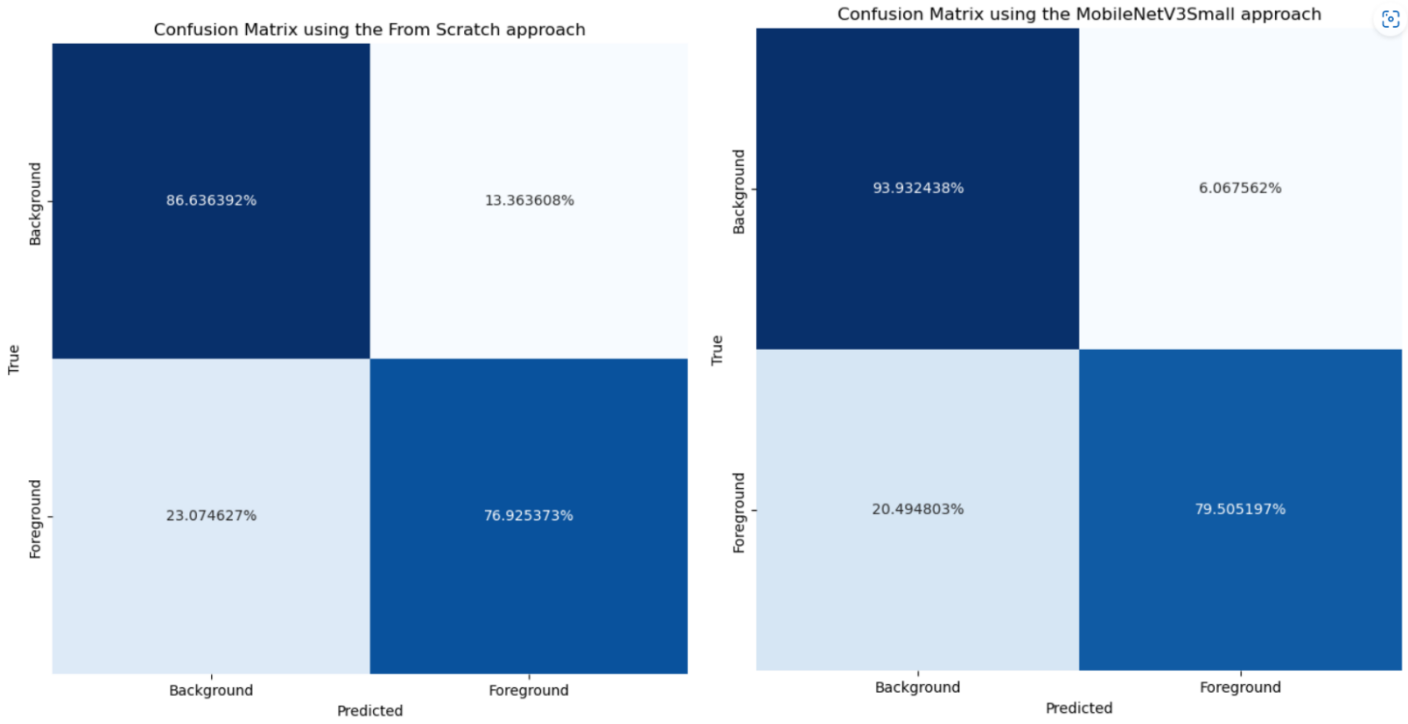


Figure 3: Confusion matrix for Semantic segmentation

4.0 Discussion of Methods Explored:

One strategy employed to mitigate these limitations was data augmentation, which aimed to artificially increase the diversity of the training data. Techniques like rotation were applied to enhance the model's ability to generalize and reduce overfitting. However, the performance gains achieved through basic data augmentation were minimal, as shown in Figure 4 and Table 1 (the QUT Jupyter lab kernel was dead after 10 epochs).

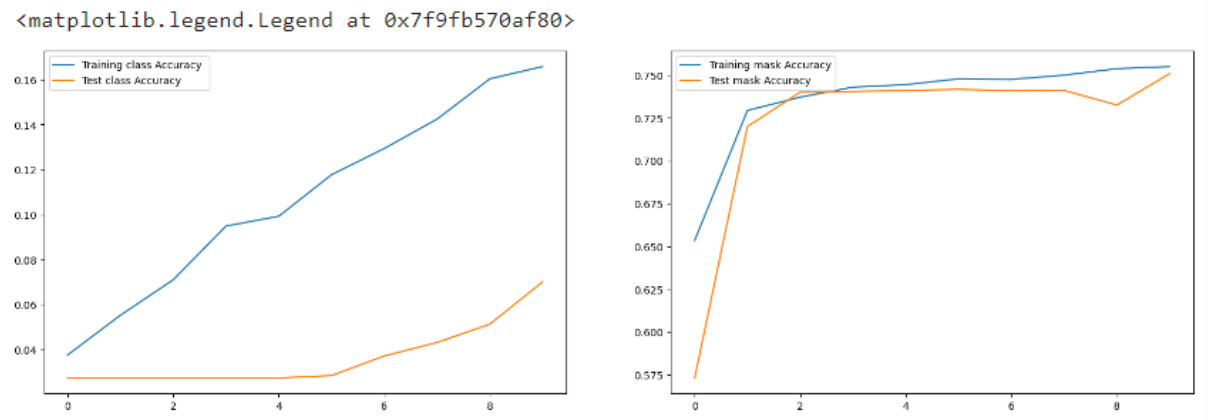


Figure 4: Training Accuracy

	From Scratch	MobileNetV3Small	From Scratch (Data augmentation)	MobileNetV3Small (Data augmentation)
Training F1	0.96	0.138	0.01	0.02
Testing F1	0.138	0.627	0.05	0.62
Background	86.8%	93.9%	77.3%	92.4%
Foreground	76.9%	79.5%	72.9%	81.5%

Table 1: All model's finding

In Table 1, it is observed that applying data augmentation to the From Scratch model resulted in a decrease in the training F1 score to 0.01, indicating that the model struggled to learn meaningful patterns from the augmented data. The testing F1 score also remained low at 0.05, suggesting poor generalization. Similarly, data augmentation had a limited impact on the MobileNetV3Small model, with only marginal increases in the training and testing F1 scores.

Although the accuracy of classifying the background and foreground classes improved slightly when data augmentation was employed, the overall performance improvement was not significant. The MobileNetV3Small model consistently outperformed the From Scratch model in both cases.

These findings highlight the significant challenges posed by limited computational power and training time in achieving improved performance. The models were unable to fully converge and learn complex patterns within the constrained training duration. Moreover, the basic data augmentation techniques, such as rotation, did not provide substantial benefits, indicating the potential need for more advanced augmentation methods.

To address these limitations and enhance model performance, the following strategies are recommended:

1. Allocate more powerful computational resources: Utilize higher-capacity hardware or distributed computing to enable longer training sessions and more extensive exploration of model architectures.
2. Extend training duration: Increase the number of training epochs to allow the models to converge and learn more complex patterns effectively.
3. Explore advanced data augmentation techniques: Investigate more sophisticated augmentation methods, such as random scaling and translation, to provide the models with a wider range of variations in the training data.
4. Consider transfer learning: Instead of training models from scratch, leverage pre-trained models on larger datasets to benefit from their learned features and architectures.

In conclusion, while data augmentation was attempted to mitigate the limitations of the small dataset, the models' performance remained suboptimal. The restricted training epochs, limited exploration of advanced augmentation techniques, and computational constraints hindered the models' ability to fully optimize and achieve significant performance improvements. To overcome these limitations, it is recommended to leverage more powerful computational resources, allocate more time for training, and explore advanced augmentation techniques to enhance model performance.