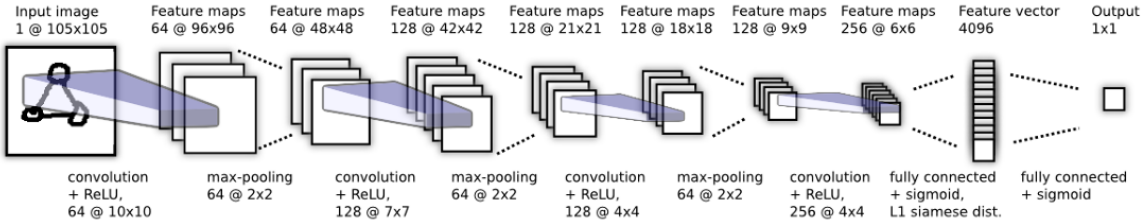
Database

The dataset contains 5,749 individuals, with 3,202 used in this task. It includes over 12,000 images, with some individuals having only one image and others having many. For training, we use 1,100 identical (Class 1) pairs and 1,100 non-identical (Class 0) pairs. The test set includes 500 pairs for each class, and no individuals overlap between the training and test sets. We also set aside 20% of the training pairs (440 pairs) for validation, leaving 1,760 pairs for actual training. All images are resized to 105x105 pixels with one channel to match the input size specified in the referenced paper.

Architecture Description

The Siamese network used for this one-shot face recognition task takes in two 105x105 grayscale images. It processes each image through the same series of layers: four convolutional layers with ReLU activations, each followed by batch normalization, and three of them followed by max-pooling. After these layers, the network flattens the features and passes them through a fully connected layer of 4096 neurons with a sigmoid activation. This creates a 4096-dimensional embedding for each image. The absolute difference between the two embeddings is then computed to measure the similarity between the images. A dropout layer is applied to prevent overfitting, and a final dense layer with a single sigmoid activation outputs the probability that the two images represent the same person.

For optimization, we use the Adam optimizer, applying L2 regularization to the weights. The binary crossentropy loss function is used as the output represents a probability, and accuracy is tracked as the primary evaluation metric. During training, we divide our training set into two parts: 80% is used for actual model training, and the remaining 20% is reserved as a validation set. This split allows us to monitor performance, and ensure the model generalizes well. Early stopping is employed to stop training when the validation performance no longer improves, ensuring an efficient and effective training process



Layer	Input Size	Filters/Units	Kernel Size	Maxpooling	Activation Function	Output Size
1	(105, 105, 1)	64	(10, 10)	(2, 2)	ReLU	(48, 48, 64)
2	(48, 48, 64)	128	(7, 7)	(2, 2)	ReLU	(21, 21, 128)
3	(21, 21, 128)	128	(4, 4)	(2, 2)	ReLU	(9, 9, 128)
4	(9, 9, 128)	256	(4, 4)	None	ReLU	(6, 6, 256)
5	(4096,)	1	-	-	Sigmoid	(1,)

We based the architecture and key parameters of our model on the design described in the "Siamese Neural Networks for One-shot Image Recognition" paper. Since there are so many possible combinations of settings, and we don’t have the resources or expertise to test them all, we followed the setup from the paper, which is already proven to work well. This included decisions like layer sizes, kernel dimensions, and the overall structure.

For aspects not covered in the paper, we used the Adam optimizer and binary cross-entropy as the loss function. For unspecified parameters, we performed a grid search to find the best settings. We also implemented early stopping, halting training if validation accuracy didn’t improve for five epochs. These choices ensured efficient training and good performance.

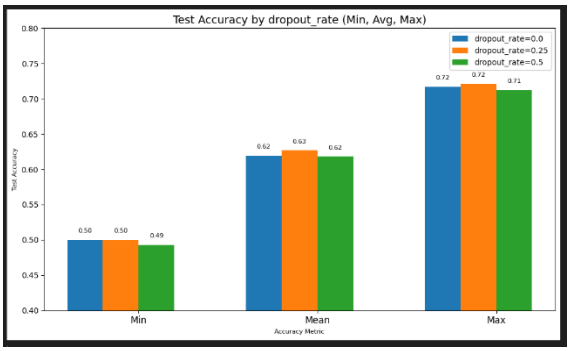
Hyper-Parameters

To determine the best parameters, we conducted experiments testing various combinations of the following hyperparameters:

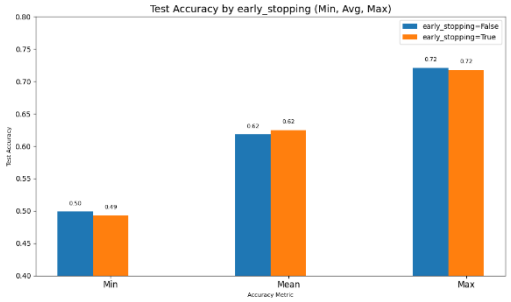
- **Dropout Rate**: 0, 0.25, 0.5
- **Learning Rate**: 0.05, 0.005, 0.0005
- **Batch Size**: 16, 32, 64
- **Early Stopping**: True, False

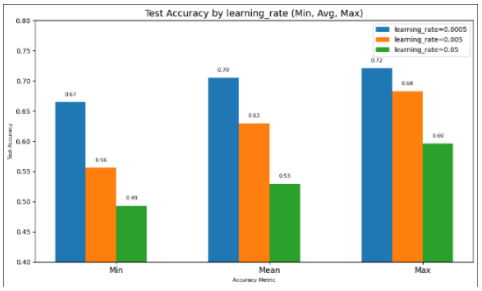
Each experiment was run for up to 20 epochs, with a stopping criterion of 5 epochs without improvement in validation accuracy when early stopping was enabled. For every parameter combination, the model was trained from scratch to avoid biases from previous experiments. Metrics such as training accuracy, validation accuracy, test accuracy, and loss were recorded for all combinations to determine the optimal configuration.

Although the differences we obtained are small, it can be seen that the dropout value that gave the best results is 0.25.

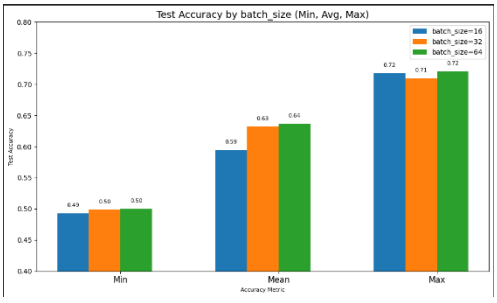


The learning rate showed the most significant differences, with 0.0005 yielding the best results.





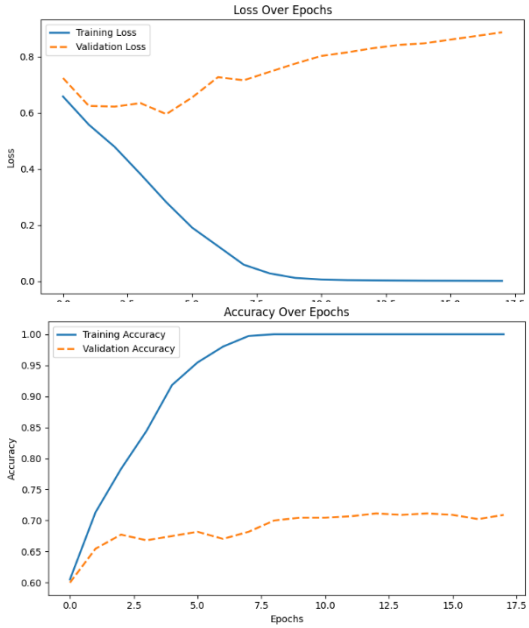
We did not observe significant differences with batch size and early stopping; therefore, we chose a batch size of 64, which gave the highest result (max).



Final result:

```
No pre-existing weights found. Starting training from scratch.
Epoch [1/20] Train Loss: 0.6578, Train Acc: 0.6051 | Val Loss: 0.7234, Val Acc: 0.6000
Epoch [2/20] Train Loss: 0.5575, Train Acc: 0.7131 | Val Loss: 0.6244, Val Acc: 0.6545
Epoch [3/20] Train Loss: 0.4785, Train Acc: 0.7830 | Val Loss: 0.6217, Val Acc: 0.6773
Epoch [4/20] Train Loss: 0.3816, Train Acc: 0.8443 | Val Loss: 0.6341, Val Acc: 0.6682
Epoch [5/20] Train Loss: 0.2814, Train Acc: 0.9182 | Val Loss: 0.5951, Val Acc: 0.6750
Epoch [6/20] Train Loss: 0.1910, Train Acc: 0.9545 | Val Loss: 0.6545, Val Acc: 0.6818
Epoch [7/20] Train Loss: 0.1252, Train Acc: 0.9801 | Val Loss: 0.7267, Val Acc: 0.6705
Epoch [8/20] Train Loss: 0.0590, Train Acc: 0.9972 | Val Loss: 0.7156, Val Acc: 0.6818
Epoch [9/20] Train Loss: 0.0282, Train Acc: 1.0000 | Val Loss: 0.7456, Val Acc: 0.7000
Epoch [10/20] Train Loss: 0.0120, Train Acc: 1.0000 | Val Loss: 0.7751, Val Acc: 0.7045
Epoch [11/20] Train Loss: 0.0064, Train Acc: 1.0000 | Val Loss: 0.8022, Val Acc: 0.7045
Epoch [12/20] Train Loss: 0.0042, Train Acc: 1.0000 | Val Loss: 0.8148, Val Acc: 0.7068
Epoch [13/20] Train Loss: 0.0034, Train Acc: 1.0000 | Val Loss: 0.8298, Val Acc: 0.7114
Epoch [14/20] Train Loss: 0.0027, Train Acc: 1.0000 | Val Loss: 0.8413, Val Acc: 0.7091
Epoch [15/20] Train Loss: 0.0022, Train Acc: 1.0000 | Val Loss: 0.8472, Val Acc: 0.7114
Epoch [16/20] Train Loss: 0.0020, Train Acc: 1.0000 | Val Loss: 0.8603, Val Acc: 0.7091
Epoch [17/20] Train Loss: 0.0017, Train Acc: 1.0000 | Val Loss: 0.8731, Val Acc: 0.7023
Epoch [18/20] Train Loss: 0.0015, Train Acc: 1.0000 | Val Loss: 0.8864, Val Acc: 0.7091
Early stopping triggered.
```

```
Training completed in 28.84 seconds.
Test Loss: 0.8315, Test Accuracy: 0.7220
```

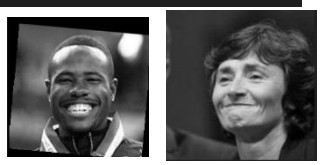



```
True Positive: Pair: ['Lisa_Raymond', '1', '2'], Model Probability: 0.92, True Label: 1
```

The two pictures are relatively similar in style, so it is also relatively easy for the model to understand that it is the same woman.

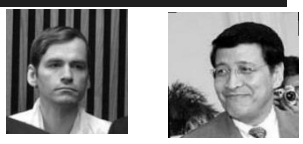
```
Pair: ['Darvis_Patton', '1', 'Estelle_Morris', '1'], Model Probability: 0.01, True Label: 0
```

can see in the pictures that these are two different people, not of the same sex and without shared features that could make it difficult for the model to understand that this is not the same person.



```
Pair: ['Daniel_Montgomery', '1', 'Hassan_Wirajuda', '1'], Model Probability: 0.92, True Label: 0
```

There are similar features between the two images that could confuse the model, such as the same hair style and similar facial expression, which probably made it difficult for the model to make a correct prediction.



```
Pair: ['Tassos_Papadopoulos', '1', '2'], Model Probability: 0.08, True Label: 1
```

In this example, it is difficult to clearly determine why the model was wrong, but it is possible that the different facial expression confused the model. In addition, in one of the images, another person entered the image, which may have created confusion in the model and used the features of the two people from this image to produce the prediction.