CS3631 - Deep Neural Networks
RNNs for Sequence Modeling

Group Members:
1. KEERTHICHANDRA H.M.P.M. - 200304N
2. NANAYAKKARA P.T.D.A. - 200411N
3. NETHMINA K.G.C. - 200418R
4. NILES D.A. - 200421U
5. SUBODHA K.R.A. - 200623P

A. RNN-based Activity Recognition Models

Model 1:	
Architecture	• “simple_rnn” layer: 64 nodes • “dense” layer: 128 nodes • “dense” layer: 12 nodes Optimizer = Adam
Train/Test Accuracy	➔ Train accuracy: 90.19 % ➔ Test accuracy: 84.46 %
Discussion	Number of nodes for RNN and dense layers were chosen by performing hyperparameter tuning. We have performed hyperparameter tuning on Optimizers in this model using Adam(validation Accuracy - 84.46), RMSprop(validation accuracy - 80.02%) and SGD(0.709082) optimizers. And the model performed best on Adam optimizer.

Model 2:	
Architecture	• "simple_rnn" layer: 64 nodes. • "dropout" layer: 64 nodes. • "simple_rnn_1" layer: 64 nodes. • "dropout_1" layer: 64 nodes. • "simple_rnn_2" layer: 64 nodes. • "dropout_2" layer. • "dense" layer: 128 nodes. • "dense_1" layer: 12 nodes Optimizer = Adam.
Train/Test Accuracy	➔ Train accuracy: 91.75 % ➔ Test accuracy: 89.07 %
Discussion	Compared to the previous architecture, this model performed better on Training dataset and Test dataset probably due to the addition of dropout layers before each RNN layer and before the final dense neural network

Model 3:	
Architecture	• "bidirectional" layer: 128 nodes. • "dropout" layer: 128 nodes. • "bidirectional_1" layer: 128 nodes. • "dropout_1" layer: 128 nodes. • "bidirectional_2" layer: 128 nodes. • "dropout_2" layer: 128 nodes. • "dense" layer: 128 nodes. • "dense_1" layer: 12 nodes. With learning_rate = 0.001 Optimizer = Adam
Train/Test Accuracy	➔ Train accuracy: 97.63 % ➔ Test accuracy 94.46 %
Discussion	This RNN architecture was the best performing model due to several reasons. In this model number of nodes were increased hence the model has become less

	biased(more complex) Also we have added bidirectional RNN layers instead of simple RNN layers.Since Bidirectional layers capture both forward and backward dependencies, train accuracy has increased. Also we have tuned the Learning rate hyperparameter. The model performed best at the learning rate of 0.001. Since smaller learning rates lead to less overfitting, test accuracy has increased significantly.

B. LSTM-based Activity Recognition Models

Model 1:	
Architecture	● "lstm" layer: 64 nodes. ● "dropout" layer: 64 nodes. ● "dense" layer: 128 nodes. ● "dropout_1" layer: 28 nodes. ● "dense_1" layer: 12 nodes. With learning_rate = 0.001 Optimizer = Adam
Train/Test Accuracy	➔ Train accuracy: 99.4 % ➔ Test accuracy: 97.62 %
Discussion	Number of nodes for LSTM and dense layers were chosen by performing hyperparameter tuning. We have added dropout layers because the test accuracies were significantly low compared to a model without dropout layers.

Model 2:	
Architecture	● Bidirectional LSTM layer: 64 nodes ● "dropout" layer: 64 nodes. ● "dense" layer: 128 nodes. ● "dropout_1" layer: 28 nodes. ● "dense_1" layer: 12 nodes. With learning_rate = 0.001 Optimizer = Adam
Train/Test Accuracy	➔ Train accuracy 99.45 % ➔ Test accuracy 98.15 %
Discussion	● Bidirectional LSTM outperformed and reduced the difference between Train and test accuracies by reducing overfitting compared to normal LSTM in this classification due to its ability to capture past and future context and dependency understanding.

Model 3:	
Architecture	● "conv1d" layer: 32 nodes. ● "batch_normalization" layer. ● "re_lu" layer (Rectified Linear Unit activation). ● "conv1d_1" layer: 64 nodes. ● "batch_normalization_1" layer. ● "re_lu_1" layer (Rectified Linear Unit activation). ● "max_pooling1d" layer. ● "lstm" layer: 64 nodes. ● "dense" layer: 128 nodes. ● "dense_1" layer: 12 nodes. With learning_rate = 0.01 Optimizer = Adam
Train/Test Accuracy	➔ Train accuracy: 99.58 %

	➔ Test accuracy: 98.64 %
Discussion	LSTM models with convolutional layers, batch normalization, pooling layers outperform bidirectional LSTM and normal LSTM in multiclass classification since LSTM with CNN models learn to detect relevant patterns within the input data, which can be crucial in multiclass classification. Also Pooling layers in a CNN model reduce the sequence length, which can help mitigate vanishing gradient problems.

C. Bidirectional GRU-based Activity Recognition Models

Model 1:	
Architecture	● "conv1d" layer: 32 nodes. ● "batch_normalization" layer. ● "re_lu" layer (Rectified Linear Unit activation). ● "conv1d_1" layer: 64 nodes. ● "batch_normalization_1" layer. ● "re_lu_1" layer (Rectified Linear Unit activation). ● "max_pooling1d" layer. ● "biGRU" layer: 64 nodes. ● "dense" layer: 128 nodes. ● "dense_1" layer: 12 nodes. With learning_rate = 0.001 Optimizer = Adam
Train/Test Accuracy	➔ Train accuracy: 98.75 % ➔ Test accuracy: 97.63 %
Discussion	● By replacing LSTM layers with biGRU layers we created this model. But compared to the previous LSTM model this performed better in training dataset but performed worse in testing data. ● Number of nodes for biGRU and dense layers were chosen by performing hyperparameter tuning. ● Typically LSTM models are less overfitting compared to biGRU based models when it comes to smaller datasets. So this could be the reason for that

Model 2:	
Architecture	● "Bidirectional GRU" layer: 128 nodes. ● "dropout" layer: 128 nodes. ● "dense" layer: 128 nodes. ● "dropout_1" layer: 128 nodes. ● "dense_1" layer: 13 nodes. With learning_rate = 0.001 Optimizer = Adam
Train/Test Accuracy	➔ Train accuracy: 98.83 % ➔ Test accuracy: 98.01 %
Discussion	This simple biGRU model without any CNN layers performed better than the previous. Adding convolutional layers to a recurrent model introduces additional complexity. Also If the dataset is small, adding convolutional layers can lead to overfitting, as CNNs tend to have many parameters. Out of three biGRU based models this model performed the best when it comes to testing accuracy probably due to the addition of dropout layers

Model 3:	
Architecture	● "Bidirectional GRU" layer: 128 nodes. ● "dense" layer: 128 nodes. ● "dense_1" layer: 13 nodes. With learning_rate = 0.001 Optimizer = Adam

Train/Test Accuracy	➔ Train accuracy: 98.9 % ➔ Test accuracy: 94.763 %
Discussion	Although this model performed better in training data, it performed less accurately in test data set probably due to overfitting since dropout layers are not present

D. <u>Transformers-based Activity Recognition Models</u>

Model 1:	
Architecture	● Hidden Dimensions = 32 ● No. of Encoders = 4 ● No. of Decoders = 4 ● No. of heads in the Multi-head Attention = 8
Train/Test Accuracy	➔ Train accuracy: 97.11% ➔ Test accuracy: 89.28%
Discussion	Model 1, with lower hidden dimensions, more encoders and decoders, and a higher number of heads in the multi-head attention mechanism, achieved a test accuracy of 89.28%. The lower hidden dimensions might result in reduced model capacity, making it less capable of capturing patterns in the data. Additionally, the increased number of encoders and decoders might lead to overfitting, which results in lower test accuracy.

Model 2:	
Architecture	● Hidden Dimensions = 64 ● No. of Encoders = 2 ● No. of Decoders = 2 ● No. of heads in the Multi-head Attention = 4
Train/Test Accuracy	➔ Train accuracy: 97.97% ➔ Test accuracy: 94.04%
Discussion	In contrast, Model 2 with larger hidden dimensions of 2 encoders and 2 decoders, along with 4 heads in the multi-head attention mechanism, achieved a higher test accuracy of 94.04%. The increased hidden dimensions enhanced the model's capacity to learn from the data, leading to better generalization. The simplified architecture helps in reducing the risk of overfitting.

Model 3:	
Architecture	● Hidden Dimensions = 24 ● No. of Encoders = 1 ● No. of Decoders = 2 ● No. of heads in the Multi-head Attention = 2
Train/Test Accuracy	➔ Train accuracy: 98.26% ➔ Test accuracy: 94.92%
Discussion	With the lowest hidden dimensions, and fewest number of encoders, decoders, and heads, this model achieved the highest test accuracy. While the hidden dimensions are lower than in Model 2, the simpler architecture with fewer parameters allowed for more efficient training, reducing the risk of overfitting.