Experiment Table

Experiment	Change Description	Hyperparameters/Adjustments	Rationale
1	Baseline (My Current Model)	- Optimizer: Adam, learning rate = 0.0007 - Layers: Dense(128, relu), Dense(64, relu), Dense(32, relu), Dense(16, relu), Dropout(0.3), Dense(2, softmax) - Batch size: 8 - Epochs: 100 - Early Stopping: monitor = 'val_loss', patience = 5, restore_best_weights = True	Establish a reference point to measure improvements.
2	Increase Learning Rate	- Optimizer: Adam, learning rate = 0.001 (changed from 0.0007) - Layers: Same as baseline - Batch size: 8 - Epochs: 100 - Early Stopping: Same as baseline	Faster learning might help escape local minima and improve convergence speed.
3	Decrease Learning Rate	- Optimizer: Adam, learning rate = 0.0003 (changed from 0.0007) - Layers: Same as baseline - Batch size: 8 - Epochs: 100 - Early Stopping: Same as baseline	Slower learning might allow more precise optimization, reducing loss further.
4	Add Learning Rate Scheduler	- Optimizer: Adam, learning rate = 0.001 - Add Callback: ReduceLROnPlateau(monitor='val_loss', factor=0.5, patience=3, min_lr=0.00001) - Layers: Same as baseline - Batch size: 8 - Epochs: 100 - Early Stopping: Same as baseline	Dynamically reduce learning rate when progress stalls, improving loss minimization.
5	Increase Batch Size	- Optimizer: Adam, learning rate = 0.0007 - Layers: Same as baseline - Batch size: 16 (changed from 8) - Epochs: 100 - Early Stopping: Same as baseline	Larger batches might stabilize gradients and improve generalization.
6	Adjust Dropout Rate	- Optimizer: Adam, learning rate = 0.0007 - Layers: Dense(128, relu), Dense(64,	Less dropout might retain useful features,

		relu), Dense(32, relu), Dense(16, relu), Dropout(0.2) (changed from 0.3), Dense(2, softmax) - Batch size: 8 - Epochs: 100 - Early Stopping: Same as baseline	boosting accuracy.
7	Add Dropout Layers	- Optimizer: Adam, learning rate = 0.0007 - Layers: Dense(128, relu), Dropout(0.2), Dense(64, relu), Dropout(0.2), Dense(32, relu), Dense(16, relu), Dropout(0.3), Dense(2, softmax) - Batch size: 8 - Epochs: 100 - Early Stopping: Same as baseline	Extra regularization might reduce overfitting, lowering loss.
8	Increase Model Capacity	- Optimizer: Adam, learning rate = 0.0007 - Layers: Dense(256, relu), Dense(128, relu), Dense(64, relu), Dense(32, relu), Dense(16, relu), Dropout(0.3), Dense(2, softmax) - Batch size: 8 - Epochs: 100 - Early Stopping: Same as baseline	More neurons might capture complex patterns, improving accuracy.
9	Use Batch Normalization	- Optimizer: Adam, learning rate = 0.0007 - Layers: Dense(128, relu), BatchNormalization(), Dense(64, relu), BatchNormalization(), Dense(32, relu), BatchNormalization(), Dense(16, relu), Dropout(0.3), Dense(2, softmax) - Batch size: 8 - Epochs: 100 - Early Stopping: Same as baseline	Normalize activations to stabilize training, potentially reducing loss and boosting accuracy.
10	Change Optimizer to RMSprop	- Optimizer: RMSprop, learning rate = 0.0007 (changed from Adam) - Layers: Same as baseline - Batch size: 8 - Epochs: 100 - Early Stopping: Same as baseline	RMSprop might adapt better to this problem, improving optimization.
11	Increase Patience in Early Stopping	- Optimizer: Adam, learning rate = 0.0007 - Layers: Same as baseline - Batch size: 8 - Epochs: 100 - Early Stopping: monitor = 'val_loss', patience = 10 (changed from 5), restore_best_weights = True	Longer patience might allow the model to find a deeper minimum, reducing loss.

12	Combine Best Adjustments (Hybrid)	- Optimizer: Adam, learning rate = 0.001 - Add Callback: ReduceLROnPlateau(monitor='val_loss', factor=0.5, patience=3, min_lr=0.00001) - Layers: Dense(256, relu), BatchNormalization(), Dense(128, relu), BatchNormalization(), Dense(64, relu), Dense(32, relu), Dense(16, relu), Dropout(0.2), Dense(2, softmax) - Batch size: 16 - Epochs: 100 - Early Stopping: monitor = 'val_loss', patience = 10, restore_best_weights = True	Combines promising changes for optimal performance.