# **Models Trained with their respective parameters and metrics.**

|  |  |  |
| --- | --- | --- |
| Model Name | Parameters and Hyper Parameters | Metrics |
| CNN\_LSTM  (Court Trial Data)  100x64x64x1 | model = Sequential()      # TimeDistributed wrapper to apply CNN across time dimension      model.add(TimeDistributed(Conv2D(32, (3, 3), activation='relu'), input\_shape=input\_shape))      model.add(TimeDistributed(MaxPooling2D((2, 2))))      model.add(TimeDistributed(Conv2D(64, (3, 3), activation='relu')))      model.add(TimeDistributed(MaxPooling2D((2, 2))))      model.add(TimeDistributed(Flatten()))      # LSTM layer for temporal processing      model.add(LSTM(50, return\_sequences=False))      # Fully connected layers      model.add(Dense(100, activation='relu'))      model.add(Dropout(0.5))      model.add(Dense(1, activation='sigmoid'))        # Compile the model      model.compile(optimizer=Adam(),                loss='binary\_crossentropy',                metrics=['accuracy']) | Accuracy: 80% ~ 90%  Loss: 0.4  [Best] |
| CNN\_LSTM  (MU3D Data)  100x64x64x1 | Same as above | Very poor  Accuracy: 30% ~ 60% |
| 3D CNN (Court Trial Data)  100x64x64x1 | model = Sequential()      # TimeDistributed wrapper to apply CNN across time dimension      model.add(TimeDistributed(Conv2D(32, (3, 3), activation='relu'), input\_shape=input\_shape))      model.add(TimeDistributed(MaxPooling2D((2, 2))))      model.add(TimeDistributed(Conv2D(64, (3, 3), activation='relu')))      model.add(TimeDistributed(MaxPooling2D((2, 2))))      model.add(TimeDistributed(Flatten()))      # LSTM layer for temporal processing      model.add(LSTM(50, return\_sequences=False))      # Fully connected layers      model.add(Dense(100, activation='relu'))      model.add(Dropout(0.5))      model.add(Dense(1, activation='sigmoid'))        # Compile the model      model.compile(optimizer=Adam(),                loss='binary\_crossentropy',                metrics=['accuracy']) | Accuracy:  71% ~ 80% |
| 2D CNN (Court Trial Data)  100x64x64x1 | model = Sequential()      # TimeDistributed wrapper to apply CNN across time dimension      model.add(Conv2D(32, (3, 3), activation='relu', input\_shape=input\_shape))      model.add(MaxPooling2D((2, 2)))      model.add(Conv2D(64, (3, 3), activation='relu'))      model.add(MaxPooling2D((2, 2)))      model.add(Flatten())      # Fully connected layers      model.add(Dense(500, activation='relu'))      model.add(Dropout(0.3))      model.add(Dense(100, activation='relu'))      model.add(Dropout(0.3))      model.add(Dense(1, activation='sigmoid'))        # Compile the model      model.compile(optimizer=Adam(),                loss='binary\_crossentropy',                metrics=['accuracy']) | Accuracy:  67% ~ 75% |
|  |  |  |
|  |  |  |
|  |  |  |