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

|  |  |  |
| --- | --- | --- |
| Model Name | Parameters and Hyper Parameters | Metrics |
| CNN\_LSTM  (Court Trial Data)  100x64x64x1  BIASED SPLIT | def build\_CNN\_LSTM(input\_shape):      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))      model.add(Dropout(0.3))      # Fully connected layers      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', Precision(), Recall(), AUC()])      return model | Accuracy: 85.71%  Loss: 0.34 |
| CNN\_LSTM  (MU3D Data)  100x64x64x1 | Same as above | Very poor  Accuracy: 30% ~ 50% |
| 3D CNN (Court Trial Data)  100x64x64x1  BIASED SPLIT | model = Sequential()        # Convolutional layers      model.add(Conv3D(32, kernel\_size=(3, 3, 3), activation='relu'))      model.add(MaxPooling3D(pool\_size=(2, 2, 2)))        model.add(Conv3D(32, kernel\_size=(3, 3, 3), activation='relu'))      model.add(MaxPooling3D(pool\_size=(2, 2, 2)))      #model.add(Dropout(0.3))        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))        # Output layer      model.add(Dense(1, activation='sigmoid'))        # Compile the model      model.compile(optimizer=Adam(),                loss='binary\_crossentropy',                metrics=['accuracy', AUC()])        return model | Accuracy:  80%  Loss: 0.46 |
| 3D CNN (Court Trial Data)  100x64x64x1  NON BIASED SPLIT | Same as before | Accuracy:  83.33%  Loss: 0.35 |
|  |  |  |
|  |  |  |

Note, on real life trial, using other sampling methods like splitting video into smaller videos with different labels gives worse results, also when sampling the whole videos into more than 100 frames or with a bigger frame size, results also seem to be worse.

MU3D didn’t achieve any usable results regardless of the preprocessing combinations and models architectures that were tried.