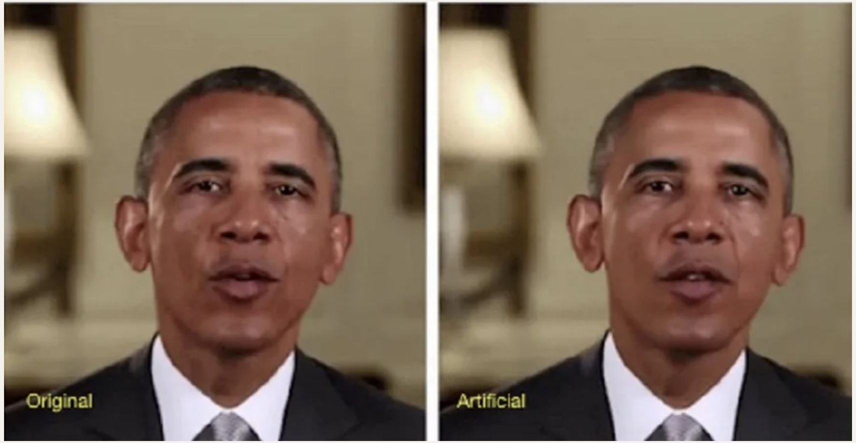
Deepfake Detection

Christina Wang and Isabel Grondin

AIT Fall 2021

What is a Deepfake?

- Modified pieces of media that imitate someone's likeness using digital modification
- Can be used to spread fake news or misinformation



Dataset

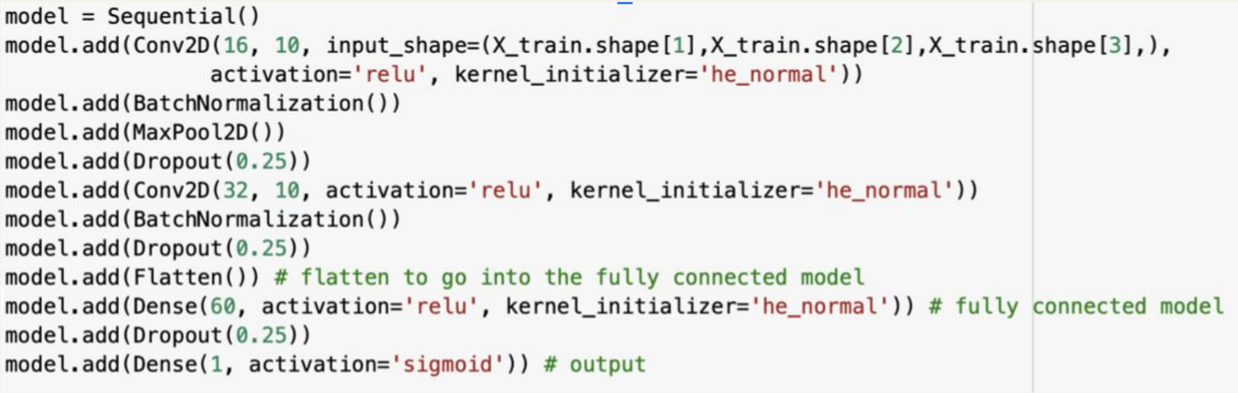
Celebrity Deepfake Dataset:

- Released with the goal of refining the basic deepfake generation algorithm
- All of the original videos were taken from publicly available YouTube videos

Preprocessing The Data:

- Balanced the dataset
- Cropped all the frames to 299 by 299
- Standardized the mode of the videos to RGB.

First Model - CNN

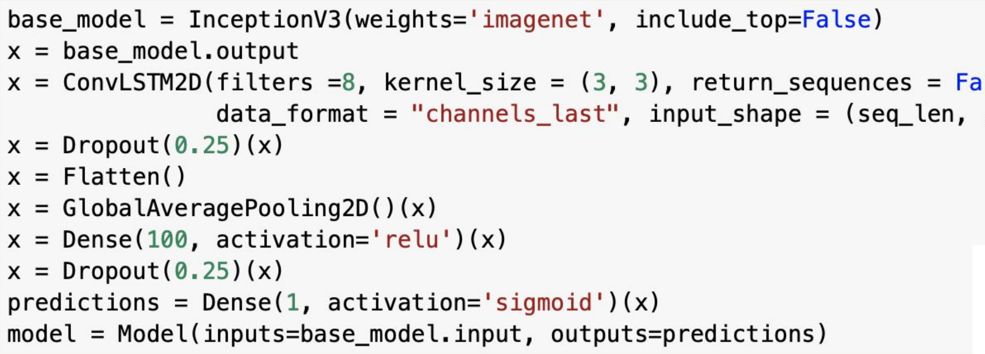


```
model = Sequential()
model.add(Conv2D(16, 10, input_shape=(X_train.shape[1],X_train.shape[2],X_train.shape[3],),
                activation='relu', kernel_initializer='he_normal'))
model.add(BatchNormalization())
model.add(MaxPool2D())
model.add(Dropout(0.25))
model.add(Conv2D(32, 10, activation='relu', kernel_initializer='he_normal'))
model.add(BatchNormalization())
model.add(Dropout(0.25))
model.add(Flatten()) # flatten to go into the fully connected model
model.add(Dense(60, activation='relu', kernel_initializer='he_normal')) # fully connected model
model.add(Dropout(0.25))
model.add(Dense(1, activation='sigmoid')) # output
```

Accuracy: 0.67

This data has a 70-30 split fake-real ratio, meaning that this model was not doing any better than guessing

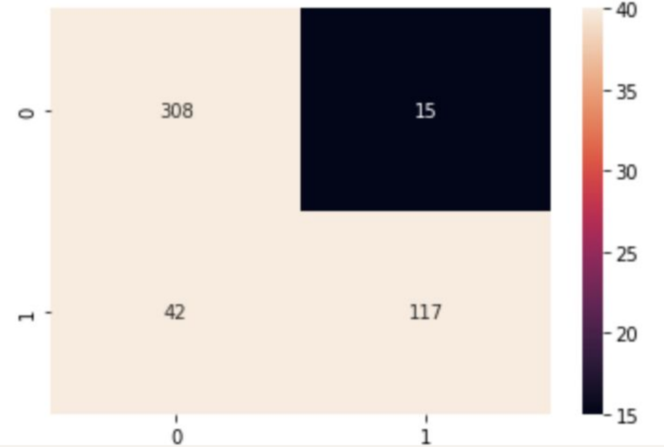
Second Model - InceptionV3



```
base_model = InceptionV3(weights='imagenet', include_top=False)
x = base_model.output
x = ConvLSTM2D(filters =8, kernel_size = (3, 3), return_sequences = False,
               data_format = "channels_last", input_shape = (seq_len, img_height, img_width, 3))
x = Dropout(0.25)(x)
x = Flatten()
x = GlobalAveragePooling2D()(x)
x = Dense(100, activation='relu')(x)
x = Dropout(0.25)(x)
predictions = Dense(1, activation='sigmoid')(x)
model = Model(inputs=base_model.input, outputs=predictions)
```

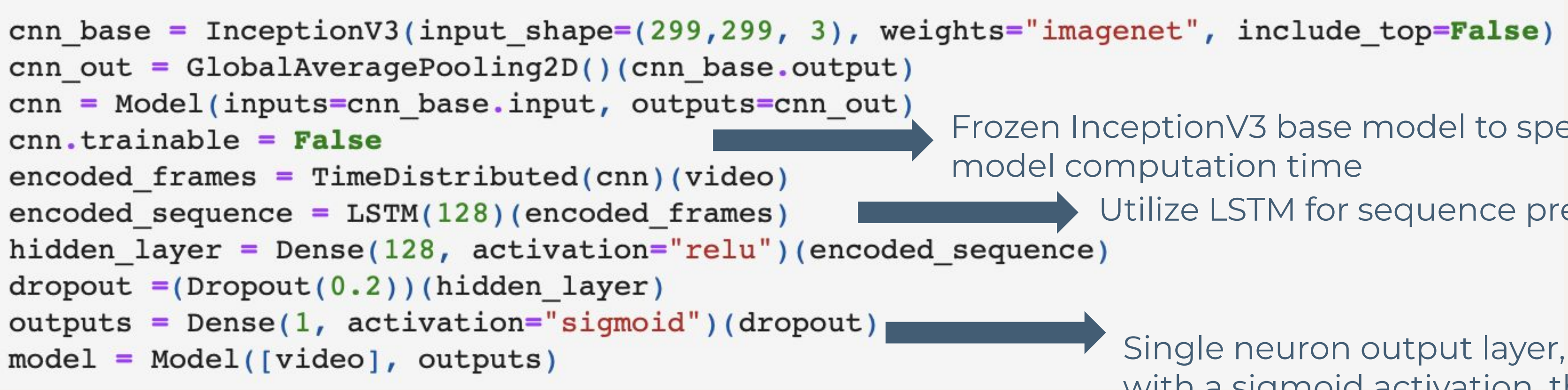
Accuracy: 0.88

We added InceptionV3 as our base
for extracting features in the frames. We also
added augmented data to our training set.



Methods

Final model: CNN LSTM Architecture



```
cnn_base = InceptionV3(input_shape=(299,299, 3), weights="imagenet", include_top=False)
cnn_out = GlobalAveragePooling2D()(cnn_base.output)
cnn = Model(inputs=cnn_base.input, outputs=cnn_out)
cnn.trainable = False
encoded_frames = TimeDistributed(cnn)(video)
encoded_sequence = LSTM(128)(encoded_frames)
hidden_layer = Dense(128, activation="relu")(encoded_sequence)
dropout =(Dropout(0.2))(hidden_layer)
outputs = Dense(1, activation="sigmoid")(dropout)
model = Model([video], outputs)
```

Frozen InceptionV3 base model to speed up model computation time

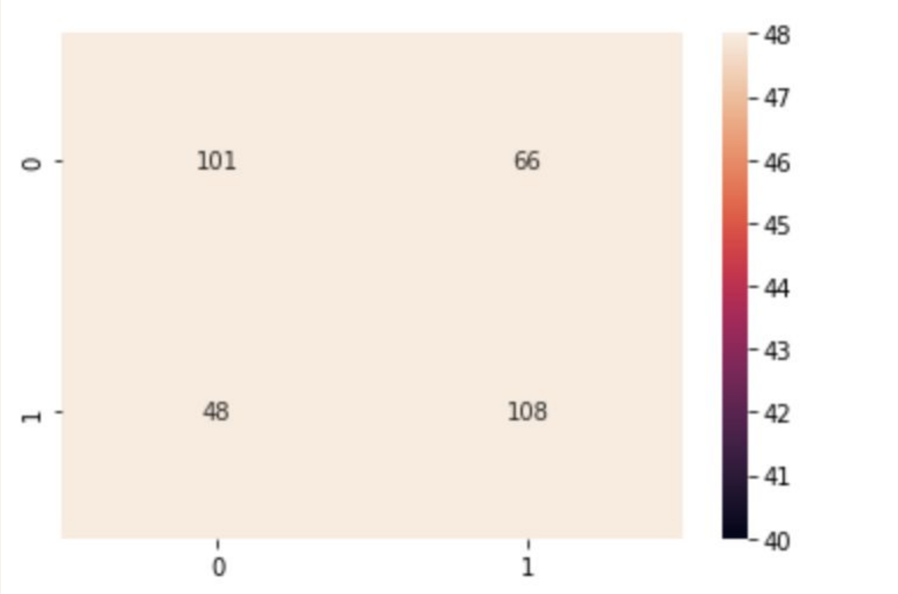
Utilize LSTM for sequence prediction

Single neuron output layer,
with a sigmoid activation, that gives the binary classification, real or fake, for each video

Results

Accuracy: 0.6471

Precision: 0.6207



Confusion Matrix

Next Steps

- Increase the number of frames being used per video
- Adding in augmented data
- Trying more feature extractor base models
  - ex. ResNet50

Thank You!