Import dependencies

In [1]:

```
import os
import cv2
import tensorflow as tf
import numpy as np
from typing import List
from matplotlib import pyplot as plt
import imageio
```

Enable GPU if possible

In [2]:

```
physical_devices = tf.config.list_physical_devices('GPU')
try:
    tf.config.experimental.set_memory_growth(physical_devices[0], True)
    print("GPU enabled")
except:
    print("GPU disabled")
```

```
GPU disabled
```

Build data loading functions

In [3]:

```
def load_video(path:str) -> List[float]:

    cap = cv2.VideoCapture(path)
    frames = []
    for _ in range(int(cap.get(cv2.CAP_PROP_FRAME_COUNT))):
        ret, frame = cap.read()
        frame = tf.image.rgb_to_grayscale(frame)
        frames.append(frame[190:236,80:220,:])
    cap.release()

    mean = tf.math.reduce_mean(frames)
    std = tf.math.reduce_std(tf.cast(frames, tf.float32))
    return tf.cast((frames - mean), tf.float32) / std
```

In [4]:

```
vocab = [x for x in "abcdefghijklmnopqrstuvwxyz'?!123456789 "]

char_to_num = tf.keras.layers.StringLookup(vocabulary=vocab, oov_token="")

num_to_char = tf.keras.layers.StringLookup(vocabulary=char_to_num.get_vocabulary(),
```

In [5]:

```
def load_alignments(path:str) -> List[str]:
    with open(path, 'r') as f:
        lines = f.readlines()
    tokens = []
    for line in lines:
        line = line.split()
        if line[2] != 'sil':
            tokens = [*tokens,' ',line[2]]
    return char_to_num(tf.reshape(tf.strings.unicode_split(tokens, input_encoding='
```

In [6]:
```
def load_data(path: str):
    path = bytes.decode(path.numpy())
    #file_name = path.split('/')[-1].split('.')[0]
    # File name splitting for windows
    file_name = path.split('\\')[-1].split('.')[0]
    video_path = os.path.join('data','s1',f'{file_name}.mpg')
    alignment_path = os.path.join('data','alignments','s1',f'{file_name}.align')
    frames = load_video(video_path)
    alignments = load_alignments(alignment_path)

    return frames, alignments
```

In [7]:
```
def mappable_function(path:str) ->List[str]:
    result = tf.py_function(load_data, [path], (tf.float32, tf.int64))
    return result
```

Test animation.gif and load video alignments (annotations)

In [8]:
```
test_path = '.\\data\\s1\\bbal6n.mpg'
```

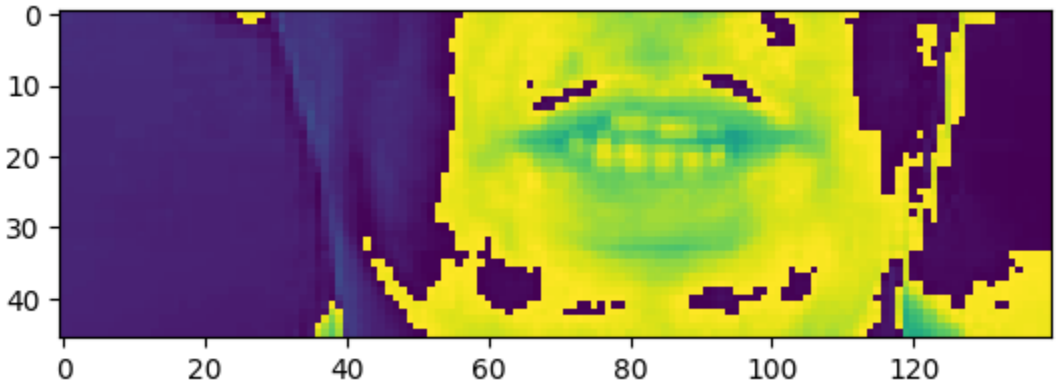
In [9]:
```
tf.convert_to_tensor(test_path).numpy().decode('utf-8').split('\\')[-1].split('.')[
```

Out[9]:
```
'bbal6n'
```

In [10]:
```
frames, alignments = load_data(tf.convert_to_tensor(test_path))

plt.imshow(frames[40])
```

Out[10]:
```
<matplotlib.image.AxesImage at 0x225a129b850>
```



In [11]:
```
alignments
```

Out[11]:
```
<tf.Tensor: shape=(21,), dtype=int64, numpy=
array([ 2,  9, 14, 39,  2, 12, 21,  5, 39,  1, 20, 39, 12, 39, 19,  9, 24,
       39, 14, 15, 23], dtype=int64)>
```

In [12]:
```
tf.strings.reduce_join([bytes.decode(x) for x in num_to_char(alignments.numpy()).nu
```

Out[12]:
```
<tf.Tensor: shape=(), dtype=string, numpy=b'bin blue at l six now'>
```

Creating data pipeline

In [13]:

```
data = tf.data.Dataset.list_files('./data/s1/*.mpg')
data = data.shuffle(500, reshuffle_each_iteration=False)
data = data.map(mappable_function)
data = data.padded_batch(2, padded_shapes=([75,None,None,None],[40]))
data = data.prefetch(tf.data.AUTOTUNE)
# Added for split
train = data.take(450)
test = data.skip(450)
```

In [14]:

```
sample = data.as_numpy_iterator()
```

In [15]:

```
val = sample.next()
```

In [16]:

```
imageio.mimsave('./animation.gif', val[0][0], fps=10)
```

```
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
```

```
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
```

```
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
Lossy conversion from float32 to uint8. Range [0.0, 10.479567527770996]. Convert ima
ge to uint8 prior to saving to suppress this warning.
```

Designing deep neural network

In [17]:

```python
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv3D, LSTM, Dense, Dropout, Bidirectional, Ma
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.callbacks import ModelCheckpoint, LearningRateScheduler
```

In [18]:

```python
model = Sequential()
model.add(Conv3D(128, 3, input_shape=(75,46,140,1), padding='same'))
model.add(Activation('relu'))
model.add(MaxPool3D((1,2,2)))

model.add(Conv3D(256, 3, padding='same'))
model.add(Activation('relu'))
model.add(MaxPool3D((1,2,2)))

model.add(Conv3D(75, 3, padding='same'))
model.add(Activation('relu'))
model.add(MaxPool3D((1,2,2)))
model.add(TimeDistributed(Flatten()))

model.add(Bidirectional(LSTM(128, kernel_initializer='Orthogonal', return_sequences
model.add(Dropout(.5))

model.add(Bidirectional(LSTM(128, kernel_initializer='Orthogonal', return_sequences
model.add(Dropout(.5))

model.add(Dense(char_to_num.vocabulary_size()+1, kernel_initializer='he_normal', ac
```

In [19]:
```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 conv3d (Conv3D)             (None, 75, 46, 140, 128)  3584      
                                                                 
 activation (Activation)     (None, 75, 46, 140, 128)  0         
                                                                 
 max_pooling3d (MaxPooling3D  (None, 75, 23, 70, 128)  0         
 )                                                               
                                                                 
 conv3d_1 (Conv3D)           (None, 75, 23, 70, 256)   884992    
                                                                 
 activation_1 (Activation)   (None, 75, 23, 70, 256)   0         
                                                                 
 max_pooling3d_1 (MaxPooling  (None, 75, 11, 35, 256)  0         
 3D)                                                             
                                                                 
 conv3d_2 (Conv3D)           (None, 75, 11, 35, 75)    518475    
                                                                 
 activation_2 (Activation)   (None, 75, 11, 35, 75)    0         
                                                                 
 max_pooling3d_2 (MaxPooling  (None, 75, 5, 17, 75)    0         
 3D)                                                             
                                                                 
 time_distributed (TimeDistr  (None, 75, 6375)         0         
 ibuted)                                                         
                                                                 
 bidirectional (Bidirectiona  (None, 75, 256)          6660096   
 l)                                                              
                                                                 
 dropout (Dropout)           (None, 75, 256)           0         
                                                                 
 bidirectional_1 (Bidirectio  (None, 75, 256)          394240    
 nal)                                                            
                                                                 
 dropout_1 (Dropout)         (None, 75, 256)           0         
                                                                 
 dense (Dense)               (None, 75, 41)            10537     
                                                                 
=================================================================
Total params: 8,471,924
Trainable params: 8,471,924
Non-trainable params: 0
_________________________________________________________________
```

In [20]:
```
yhat = model.predict(val[0])
```

```
1/1 [==============================] - 3s 3s/step
```

In [21]:
```
tf.strings.reduce_join([num_to_char(x) for x in tf.argmax(yhat[0],axis=1)])
```

Out[21]:
```
<tf.Tensor: shape=(), dtype=string, numpy=b'nkkkkkkkkvvvvvvv1ttttttttttttttttttttt
ttttttttttt11111111111111111111111111'>
```

In [22]:
```
tf.strings.reduce_join([num_to_char(tf.argmax(x)) for x in yhat[0]])
```

```
Out[22]: <tf.Tensor: shape=(), dtype=string, numpy=b'nkkkkkkkkvvvvvvv1tttttttttttttttttttt
         tttttttttt11111111111111111111111111'>
```

```
In [23]: model.input_shape
```

```
Out[23]: (None, 75, 46, 140, 1)
```

```
In [24]: model.output_shape
```

```
Out[24]: (None, 75, 41)
```

Build model training functions

```
In [25]: def scheduler(epoch, lr):
             if epoch < 30:
                 return lr
             else:
                 return lr * tf.math.exp(-0.1)
```

```
In [26]: def CTCLoss(y_true, y_pred):
             batch_len = tf.cast(tf.shape(y_true)[0], dtype="int64")
             input_length = tf.cast(tf.shape(y_pred)[1], dtype="int64")
             label_length = tf.cast(tf.shape(y_true)[1], dtype="int64")

             input_length = input_length * tf.ones(shape=(batch_len, 1), dtype="int64")
             label_length = label_length * tf.ones(shape=(batch_len, 1), dtype="int64")

             loss = tf.keras.backend.ctc_batch_cost(y_true, y_pred, input_length, label_leng
             return loss
```

```
In [27]: class ProduceExample(tf.keras.callbacks.Callback):
             def __init__(self, dataset) -> None:
                 self.dataset = dataset.as_numpy_iterator()

             def on_epoch_end(self, epoch, logs=None) -> None:
                 data = self.dataset.next()
                 yhat = self.model.predict(data[0])
                 decoded = tf.keras.backend.ctc_decode(yhat, [75,75], greedy=False)[0][0].nu
                 for x in range(len(yhat)):
                     print('Original:', tf.strings.reduce_join(num_to_char(data[1][x])).nump
                     print('Prediction:', tf.strings.reduce_join(num_to_char(decoded[x])).nu
                     print('~'*100)
```

Training the model

```
In [28]: model.compile(optimizer=Adam(learning_rate=0.01), loss=CTCLoss)
```

```
In [29]: checkpoint_callback = ModelCheckpoint(os.path.join('models','checkpoint'), monitor=
```

```
In [30]: schedule_callback = LearningRateScheduler(scheduler)
```

In [31]:
```
example_callback = ProduceExample(test)
```

In [32]:
```
model.fit(train, validation_data=test, epochs=100, callbacks=[checkpoint_callback,
```

```
Epoch 1/100
  1/450 [..............................] - ETA: 4:13:52 - loss: 235.5548
```

```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
Cell In[32], line 1
----> 1 model.fit(train, validation_data=test, epochs=100, callbacks=[checkpoint_cal
lback, schedule_callback, example_callback])

File ~\anaconda3\envs\py310\lib\site-packages\keras\utils\traceback_utils.py:65, in
filter_traceback.<locals>.error_handler(*args, **kwargs)
     63 filtered_tb = None
     64 try:
---> 65     return fn(*args, **kwargs)
     66 except Exception as e:
     67     filtered_tb = _process_traceback_frames(e.__traceback__)

File ~\anaconda3\envs\py310\lib\site-packages\keras\engine\training.py:1564, in Mode
l.fit(self, x, y, batch_size, epochs, verbose, callbacks, validation_split, validati
on_data, shuffle, class_weight, sample_weight, initial_epoch, steps_per_epoch, valid
ation_steps, validation_batch_size, validation_freq, max_queue_size, workers, use_mu
ltiprocessing)
   1556 with tf.profiler.experimental.Trace(
   1557     "train",
   1558     epoch_num=epoch,
   (...)
   1561     _r=1,
   1562 ):
   1563     callbacks.on_train_batch_begin(step)
-> 1564     tmp_logs = self.train_function(iterator)
   1565     if data_handler.should_sync:
   1566         context.async_wait()

File ~\anaconda3\envs\py310\lib\site-packages\tensorflow\python\util\traceback_util
s.py:150, in filter_traceback.<locals>.error_handler(*args, **kwargs)
    148 filtered_tb = None
    149 try:
--> 150   return fn(*args, **kwargs)
    151 except Exception as e:
    152   filtered_tb = _process_traceback_frames(e.__traceback__)

File ~\anaconda3\envs\py310\lib\site-packages\tensorflow\python\eager\def_function.p
y:915, in Function.__call__(self, *args, **kwds)
    912 compiler = "xla" if self._jit_compile else "nonXla"
    914 with OptionalXlaContext(self._jit_compile):
--> 915   result = self._call(*args, **kwds)
    917 new_tracing_count = self.experimental_get_tracing_count()
    918 without_tracing = (tracing_count == new_tracing_count)

File ~\anaconda3\envs\py310\lib\site-packages\tensorflow\python\eager\def_function.p
y:947, in Function._call(self, *args, **kwds)
    944   self._lock.release()
    945   # In this case we have created variables on the first call, so we run the
    946   # defunned version which is guaranteed to never create variables.
--> 947   return self._stateless_fn(*args, **kwds)  # pylint: disable=not-callable
    948 elif self._stateful_fn is not None:
    949   # Release the lock early so that multiple threads can perform the call
    950   # in parallel.
    951   self._lock.release()
```

```
File ~\anaconda3\envs\py310\lib\site-packages\tensorflow\python\eager\function.py:24
96, in Function.__call__(self, *args, **kwargs)
   2493 with self._lock:
   2494   (graph_function,
   2495    filtered_flat_args) = self._maybe_define_function(args, kwargs)
-> 2496 return graph_function._call_flat(
   2497     filtered_flat_args, captured_inputs=graph_function.captured_inputs)

File ~\anaconda3\envs\py310\lib\site-packages\tensorflow\python\eager\function.py:18
62, in ConcreteFunction._call_flat(self, args, captured_inputs, cancellation_manage
r)
   1858 possible_gradient_type = gradients_util.PossibleTapeGradientTypes(args)
   1859 if (possible_gradient_type == gradients_util.POSSIBLE_GRADIENT_TYPES_NONE
   1860     and executing_eagerly):
   1861   # No tape is watching; skip to running the function.
-> 1862   return self._build_call_outputs(self._inference_function.call(
   1863       ctx, args, cancellation_manager=cancellation_manager))
   1864 forward_backward = self._select_forward_and_backward_functions(
   1865     args,
   1866     possible_gradient_type,
   1867     executing_eagerly)
   1868 forward_function, args_with_tangents = forward_backward.forward()

File ~\anaconda3\envs\py310\lib\site-packages\tensorflow\python\eager\function.py:49
9, in _EagerDefinedFunction.call(self, ctx, args, cancellation_manager)
    497 with _InterpolateFunctionError(self):
    498   if cancellation_manager is None:
--> 499     outputs = execute.execute(
    500         str(self.signature.name),
    501         num_outputs=self._num_outputs,
    502         inputs=args,
    503         attrs=attrs,
    504         ctx=ctx)
    505   else:
    506     outputs = execute.execute_with_cancellation(
    507         str(self.signature.name),
    508         num_outputs=self._num_outputs,
   (...)
    511         ctx=ctx,
    512         cancellation_manager=cancellation_manager)

File ~\anaconda3\envs\py310\lib\site-packages\tensorflow\python\eager\execute.py:54,
in quick_execute(op_name, num_outputs, inputs, attrs, ctx, name)
     52 try:
     53   ctx.ensure_initialized()
---> 54   tensors = pywrap_tfe.TFE_Py_Execute(ctx._handle, device_name, op_name,
     55                                       inputs, attrs, num_outputs)
     56 except core._NotOkStatusException as e:
     57   if name is not None:

KeyboardInterrupt:
```

Saving the weights of trained model

```
In [33]: #model.save("LipReaderDNN")
```