Spoken Digit Recognition

In this notebook, You will do Spoken Digit Recognition.

Input - speech signal, output - digit number

It contains

1. Reading the dataset. and Preprocess the data set. Detailed instrctions are given below. You have to write the code in the same cell which contains the instrction.
2. Training the LSTM with RAW data
3. Converting to spectrogram and Training the LSTM network
4. Creating the augmented data and doing step 2 and 3 again.

instructions:

    1. Don't change any Grader Functions. Don't manipulate any Grader functions. If you manipulate any, it will be considered as plagiarised.

    2. Please read the instructions on the code cells and markdown cells. We will explain what to write.

    3. please return outputs in the same format what we asked. Eg. Don't return List of we are asking for a numpy array.

    4. Please read the external links that we are given so that you will learn the concept behind the code that you are writing.

    5. We are giving instructions at each section if necessary, please follow them.

Every Grader function has to return True.

In [1]:

```
import numpy as np
```

```
import pandas as pd
import librosa
import os
import tensorflow as tf
##if you need any imports you can do that here.
```

In [2]:

```
%load_ext tensorboard
```

In [3]:

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

We shared recordings.zip, please unzip those.

In [4]:

```
#read the all file names in the recordings folder given by us
#(if you get entire path, it is very useful in future)
#save those files names as list in "all_files"
from tqdm import tqdm
path=[]
all_files=[]
for i in tqdm(os.listdir('/content/drive/MyDrive/recordings')):
    path.append('/content/drive/MyDrive/recordings'+'/'+i)
    all_files.append(i)
```

```
100%|██████████| 2000/2000 [00:00<00:00, 1462449.09it/s]
```

Grader function 1

In [5]:

```
def grader_files():
    temp = len(all_files)==2000
    temp1 = all([x[-3:]=="wav" for x in all_files])
    temp = temp and temp1
    return temp
grader_files()
```

Out[5]:

```
True
```

Create a dataframe(name=df_audio) with two columns(path, label).

Create a dataframe(name=df_audio) with two columns(path, label).
You can get the label from the first letter of name.
Eg: 0_jackson_0 --> 0
0_jackson_43 --> 0

In [6]:

```
allf=[int(list(i[:-4].split('_'))[-3]) for i in all_files]
df_audio=pd.DataFrame(list(zip(path,allf)),columns =['path', 'label'])
df_audio.head(2)
#Create a dataframe(name=df_audio) with two columns(path, label).
#You can get the label from the first letter of name.
#Eg: 0_jackson_0 --> 0
#0_jackson_43 --> 0
```

Out[6]:

	path	label
0	**/content/drive/MyDrive/recordings/2_yweweler_6...**	**2**
1	**/content/drive/MyDrive/recordings/4_yweweler_1...**	**4**

In [7]:

```
#info
df_audio.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2000 entries, 0 to 1999
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   path    2000 non-null   object
 1   label   2000 non-null   int64
dtypes: int64(1), object(1)
memory usage: 31.4+ KB
```

Grader function 2

In [8]:

```
def grader_df():
    flag_shape = df_audio.shape==(2000,2)
    flag_columns = all(df_audio.columns==['path', 'label'])
    list_values = list(df_audio.label.value_counts())
    flag_label = len(list_values)==10
    flag_label2 = all([i==200 for i in list_values])
    final_flag = flag_shape and flag_columns and flag_label and flag_label2
```

```
    return final_flag
grader_df()
```

Out[8]:

True

In [9]:

```
from sklearn.utils import shuffle
df_audio = shuffle(df_audio, random_state=33)#don't change the random state
```

Train and Validation split

In [10]:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test=train_test_split(df_audio['path'],df_audio['label'],test_size=0.30,random_state=45,stratify=df_a
udio[['label']])
#split the data into train and validation and save in X_train, X_test, y_train, y_test
#use stratify sampling
#use random state of 45
#use test size of 30%
```

Grader function 3

In [11]:

```
def grader_split():
    flag_len = (len(X_train)==1400) and (len(X_test)==600) and (len(y_train)==1400) and (len(y_test)==600)
    values_ytrain = list(y_train.value_counts())
    flag_ytrain = (len(values_ytrain)==10) and (all([i==140 for i in values_ytrain]))
    values_ytest = list(y_test.value_counts())
    flag_ytest = (len(values_ytest)==10) and (all([i==60 for i in values_ytest]))
    final_flag = flag_len and flag_ytrain and flag_ytest
    return final_flag
grader_split()
```

Out[11]:

True

In [12]:

```
# y_train=tf.keras.utils.to_categorical(y_train, 10)
# y_test =tf.keras.utils.to_categorical(y_test, 10)
```

Preprocessing

All files are in the "WAV" format. We will read those raw data files using the librosa

In [13]:

```
sample_rate = 22050
def load_wav(x, get_duration=True):
    '''This return the array values of audio with sampling rate of 22050 and Duration'''
    #loading the wav file with sampling rate of 22050
    samples, sample_rate = librosa.load(x, sr=22050)
    if get_duration:
        duration = librosa.get_duration(samples, sample_rate)
        return [samples, duration]
    else:
        return samples
```

In [14]:

```
#use load_wav function that was written above to get every wave.
#save it in X_train_processed and X_test_processed
# X_train_processed/X_test_processed should be dataframes with two columns(raw_data, duration) with same index of X_train/y_train
X_train_processed=pd.DataFrame(columns=['raw_data', 'duration'])
X_test_processed=pd.DataFrame(columns=['raw_data', 'duration'])
for i in tqdm(X_train.index):
    X_train_processed.loc[i]=load_wav(X_train[i])
for i in tqdm(X_test.index):
    X_test_processed.loc[i]=load_wav(X_test[i])
```

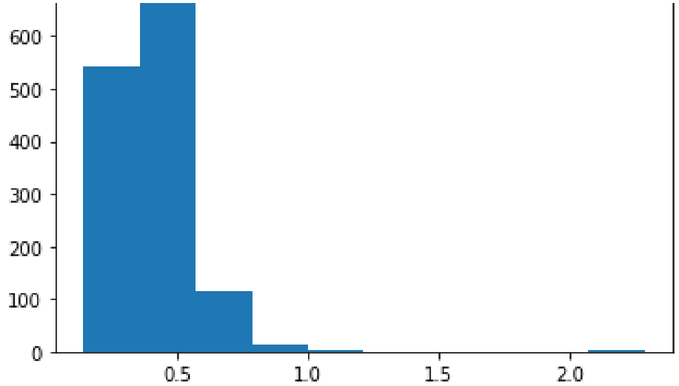
```
100%|██████████| 1400/1400 [06:46<00:00,  3.44it/s]
100%|██████████| 600/600 [02:47<00:00,  3.59it/s]
```

In [15]:

```
#plot the histogram of the duration for trian
import matplotlib.pyplot as plt
import seaborn as sns
plt.hist(X_train_processed['duration'])
```

Out[15]:

```
(array([542., 725., 116.,  13.,   2.,   0.,   0.,   0.,   0.,   2.]),
 array([0.14353741, 0.35746032, 0.57138322, 0.78530612, 0.99922902,
        1.21315193, 1.42707483, 1.64099773, 1.85492063, 2.06884354,
        2.28276644]),
 <a list of 10 Patch objects>)
```

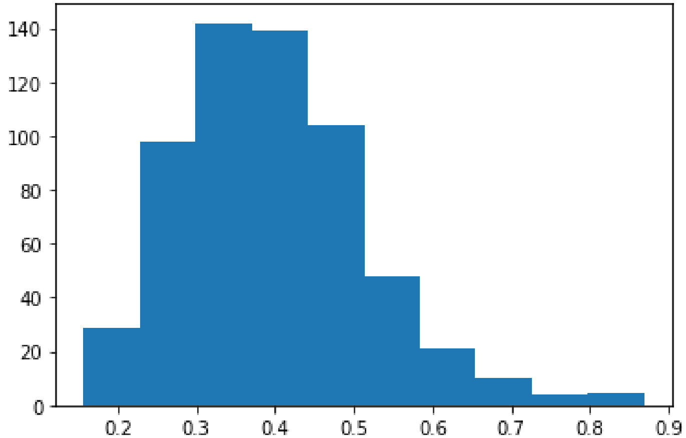


In [16]:

```
#plot the histogram of the duration for trian
import matplotlib.pyplot as plt
import seaborn as sns
plt.hist(X_test_processed['duration'])
```

Out[16]:

```
(array([ 29.,  98., 142., 139., 104.,  48.,  21.,  10.,   4.,   5.]),
 array([0.15641723, 0.22769161, 0.29896599, 0.37024036, 0.44151474,
        0.51278912, 0.58406349, 0.65533787, 0.72661224, 0.79788662,
        0.869161  ]),
 <a list of 10 Patch objects>)
```



In [17]:

```
#print 0 to 100 percentile values with step size of 10 for train data duration.
for j in range(0,101,10):
    print('{} th percentile is {}'.format(j,np.percentile(X_train_processed['duration'],j)))
```

0 th percentile is 0.1435374149659864

```
10 th percentile is 0.2620090702947846
20 th percentile is 0.30259410430839
30 th percentile is 0.33474376417233564
40 th percentile is 0.36007256235827667
50 th percentile is 0.3903854875283447
60 th percentile is 0.41844897959183674
70 th percentile is 0.44775510204081626
80 th percentile is 0.48462585034013606
90 th percentile is 0.5619183673469388
100 th percentile is 2.282766439909297
```

In [18]:

```
##print 90 to 100 percentile values with step size of 1.
for j in range(90,101,1):
    print('{} th percentile is {}'.format(j,np.percentile(X_train_processed['duration'],j)))
```

```
90 th percentile is 0.5619183673469388
91 th percentile is 0.576156009070295
92 th percentile is 0.5863274376417237
93 th percentile is 0.6037156462585042
94 th percentile is 0.617911111111111
95 th percentile is 0.6330226757369615
96 th percentile is 0.6436226757369614
97 th percentile is 0.66358231292517
98 th percentile is 0.689717006802721
99 th percentile is 0.7961165532879818
100 th percentile is 2.282766439909297
```

Grader function 4

In [19]:

```
def grader_processed():
    flag_columns = (all(X_train_processed.columns==['raw_data', 'duration'])) and (all(X_test_processed.columns==['raw_data', 'du
ration']))
    flag_shape = (X_train_processed.shape ==(1400, 2)) and (X_test_processed.shape==(600,2))
    return flag_columns and flag_shape
grader_processed()
```

Out[19]:

```
True
```

```
Based on our analysis 99 percentile values are less than 0.8sec so we will limit maximum length of X_train_processed and X_t
est_processed to 0.8 sec. It is similar to pad_sequence for a text dataset.

While loading the audio files, we are using sampling rate of 22050 so one sec will give array of length 22050. so, our maxim
```

um length is 0.8*22050 = 17640

Pad with Zero if length of sequence is less than 17640 else Truncate the number.

Also create a masking vector for train and test.

masking vector value = 1 if it is real value, 0 if it is pad value. Masking vector data type must be bool.

In [20]:

```
max_length  = 17640
```

In [21]:

```
print(X_train_processed['raw_data'].shape[0])
```

```
1400
```

In [22]:

```
def mask(data):
    maska = []
    for i in tqdm(data):
        masks=[]
        for j in i:
            if j!=0:
                masks.append(True)
            else:
                masks.append(False)
        maska.append(np.array(masks))
    return(np.array(maska))
```

In [23]:

```
## as discussed above, Pad with Zero if length of sequence is less than 17640 else Truncate the number.
## save in the X_train_pad_seq, X_test_pad_seq
## also Create masking vector X_train_mask, X_test_mask
from tensorflow.keras.preprocessing.sequence import pad_sequences
X_train_pad_seq=pad_sequences(X_train_processed['raw_data'], maxlen=max_length, dtype='float32', padding='post',truncating='pre')
X_test_pad_seq =pad_sequences(X_test_processed['raw_data'], maxlen=max_length, dtype='float32', padding='post',truncating='pre')
X_train_mask = mask(X_train_pad_seq)
X_test_mask  = mask(X_test_pad_seq)
## all the X_train_pad_seq, X_test_pad_seq, X_train_mask, X_test_mask will be numpy arrays mask vector dtype must be bool.
```

```
100%|██████████| 1400/1400 [00:42<00:00, 32.96it/s]
100%|██████████| 600/600 [00:17<00:00, 33.43it/s]
```

Grader function 5

In [24]:

```
def grader_padoutput():
    flag_padshape = (X_train_pad_seq.shape==(1400, 17640)) and (X_test_pad_seq.shape==(600, 17640)) and (y_train.shape==(1400,))
    flag_maskshape = (X_train_mask.shape==(1400, 17640)) and (X_test_mask.shape==(600, 17640)) and (y_test.shape==(600,))
    flag_dtype = (X_train_mask.dtype==bool) and (X_test_mask.dtype==bool)
    return flag_padshape and flag_maskshape and flag_dtype
grader_padoutput()
```

Out[24]:

```
True
```

1. Giving Raw data directly.

```
Now we have

Train data: X_train_pad_seq, X_train_mask and y_train
Test data: X_test_pad_seq, X_test_mask and y_test

We will create a LSTM model which takes this input.

Task:

1. Create an LSTM network which takes "X_train_pad_seq" as input, "X_train_mask" as mask input. You can use any number of LS
TM cells. Please read LSTM documentation(https://www.tensorflow.org/api_docs/python/tf/keras/layers/LSTM) in tensorflow to k
now more about mask and also https://www.tensorflow.org/guide/keras/masking_and_padding
2. Get the final output of the LSTM and give it to Dense layer of any size and then give it to Dense layer of size 10(becaus
e we have 10 outputs) and then compile with the sparse categorical cross entropy( because we are not converting it to one ho
t vectors).
3. Use tensorboard to plot the graphs of loss and metric(use micro F1 score as metric) and histograms of gradients.
4. make sure that it won't overfit.
5. You are free to include any regularization
```

In [25]:

```
from tensorflow.keras.layers import Input, LSTM, Dense, Flatten
from tensorflow.keras.models import Model
import tensorflow as tf
```

In [26]:

```
## as discussed above, please write the LSTM
input= Input(shape=(17640,1,),dtype=np.float32)
mask1  = Input(shape=(17640,),dtype='bool')
lstm_output=LSTM(20)(inputs=input,mask=mask1)
Dense1=Dense(10,activation='softmax')(lstm_output)
model= Model(inputs=[input,mask1],outputs=Dense1)
model.summary()
```

```
Model: "model"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_1 (InputLayer)            [(None, 17640, 1)]   0
__________________________________________________________________________________________________
input_2 (InputLayer)            [(None, 17640)]      0
__________________________________________________________________________________________________
lstm (LSTM)                     (None, 20)           1760        input_1[0][0]
                                                                 input_2[0][0]
__________________________________________________________________________________________________
dense (Dense)                   (None, 10)           210         lstm[0][0]
==================================================================================================
Total params: 1,970
Trainable params: 1,970
Non-trainable params: 0
__________________________________________________________________________________________________
```

In [27]:

```
from sklearn.metrics import f1_score
def f1_score_func(y_true,y_pred):
    return f1_score(y_true, y_pred,average='micro')

def f1_scores(y_true,y_pred):
    y_pred=tf.math.argmax(y_pred,axis=1)
    return(tf.py_function(f1_score_func,(y_true,y_pred),tf.double))
```

In [28]:

```
re_X_train_pad_seq=X_train_pad_seq.reshape(1400,17640,1)
re_X_test_pad_seq=X_test_pad_seq.reshape(600,17640,1)
```

In [29]:

```
from datetime import datetime
model.compile(loss='SparseCategoricalCrossentropy',optimizer='adam',metrics=['accuracy',f1_scores])
logdir = "logs/fit/" + datetime.now().strftime("%Y%m%d-%H%M%S")
```

In [30]:

```
tensorboard_callback = tf.keras.callbacks.TensorBoard(log_dir=logdir)
model.fit([re_X_train_pad_seq,X_train_mask],y_train,validation_data=([re_X_test_pad_seq,X_test_mask],y_test),batch_size=100,epochs
=7,callbacks=[tensorboard_callback])
```

```
Epoch 1/7
14/14 [==============================] - 25s 1s/step - loss: 2.3029 - accuracy: 0.0956 - f1_scores: 0.0956 - val_loss: 2.3026 - va
l_accuracy: 0.1000 - val_f1_scores: 0.1000
Epoch 2/7
14/14 [==============================] - 11s 825ms/step - loss: 2.3025 - accuracy: 0.1061 - f1_scores: 0.1061 - val_loss: 2.3026 -
val_accuracy: 0.0933 - val_f1_scores: 0.0933
Epoch 3/7
14/14 [==============================] - 12s 867ms/step - loss: 2.3026 - accuracy: 0.0875 - f1_scores: 0.0875 - val_loss: 2.3026 -
val_accuracy: 0.0867 - val_f1_scores: 0.0867
Epoch 4/7
14/14 [==============================] - 12s 841ms/step - loss: 2.3027 - accuracy: 0.0816 - f1_scores: 0.0816 - val_loss: 2.3026 -
val_accuracy: 0.1000 - val_f1_scores: 0.1000
Epoch 5/7
14/14 [==============================] - 12s 839ms/step - loss: 2.3026 - accuracy: 0.1061 - f1_scores: 0.1061 - val_loss: 2.3026 -
val_accuracy: 0.1033 - val_f1_scores: 0.1033
Epoch 6/7
14/14 [==============================] - 12s 855ms/step - loss: 2.3026 - accuracy: 0.1018 - f1_scores: 0.1018 - val_loss: 2.3026 -
val_accuracy: 0.0917 - val_f1_scores: 0.0917
Epoch 7/7
14/14 [==============================] - 12s 847ms/step - loss: 2.3027 - accuracy: 0.0974 - f1_scores: 0.0974 - val_loss: 2.3026 -
val_accuracy: 0.1000 - val_f1_scores: 0.1000
```

Out[30]:

```
<tensorflow.python.keras.callbacks.History at 0x7f350bf93b00>
```

In [ ]:

```
%tensorboard --logdir=./
```

2. Converting into spectrogram and giving spectrogram data as input

```
We can use librosa to convert raw data into spectrogram. A spectrogram shows the features in a two-dimensional representatio
n with the
intensity of a frequency at a point in time i.e we are converting Time domain to frequency domain. you can read more about t
his in https://pnsn.org/spectrograms/what-is-a-spectrogram
```

In [32]:

```
def convert_to_spectrogram(raw_data):
    '''converting to spectrogram'''
    spectrum = librosa.feature.melspectrogram(y=raw_data, sr=sample_rate, n_mels=64)
    logmel_spectrum = librosa.power_to_db(S=spectrum, ref=np.max)
    return logmel_spectrum
```

In [33]:

```
##use convert_to_spectrogram and convert every raw sequence in X_train_pad_seq and X_test_pad-seq.
## save those all in the X_train_spectrogram and X_test_spectrogram ( These two arrays must be numpy arrays)
X_train_spectrogram =np.array([np.array(convert_to_spectrogram(i)) for i in tqdm((X_train_pad_seq))])
X_test_spectrogram = np.array([np.array(convert_to_spectrogram(i)) for i in tqdm((X_test_pad_seq))])
```

```
100%|██████████| 1400/1400 [00:06<00:00, 227.22it/s]
100%|██████████| 600/600 [00:02<00:00, 221.80it/s]
```

Grader function 6

In [34]:

```
def grader_spectrogram():
    flag_shape = (X_train_spectrogram.shape==(1400,64, 35)) and (X_test_spectrogram.shape == (600, 64, 35))
    return flag_shape
grader_spectrogram()
```

Out[34]:

```
True
```

```
Now we have

Train data: X_train_spectrogram and y_train
Test data: X_test_spectrogram and y_test

We will create a LSTM model which takes this input.

Task:

1. Create an LSTM network which takes "X_train_spectrogram" as input and has to return output at every time step.
2. Average the output of every time step and give this to the Dense layer of any size.
(ex: Output from LSTM will be  (#., time_steps, features) average the output of every time step i.e, you should get (#.,time
_steps)
and then pass to dense layer )
3. give the above output to Dense layer of size 10( output layer) and train the network with sparse categorical cross entrop
y.
4. Use tensorboard to plot the graphs of loss and metric(use micro F1 score as metric) and histograms of gradients.
5. make sure that it won't overfit.
6. You are free to include any regularization
```

In [35]:

```
print(X_train_spectrogram.shape)
```

```
(1400, 64, 35)
```

In [36]:

```
input= Input(shape=(64, 35,),dtype='float32')
lstm_output=LSTM(100,return_sequences=True)(input)
Dense1=Dense(50,activation='relu')(tf.math.reduce_mean(lstm_output, 2))
Dense2=Dense(10,activation='softmax')(Dense1)
model= Model(inputs=input,outputs=Dense2)
model.summary()
```

```
Model: "model_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_3 (InputLayer)         [(None, 64, 35)]          0
_________________________________________________________________
lstm_1 (LSTM)                (None, 64, 100)           54400
_________________________________________________________________
tf.math.reduce_mean (TFOpLam (None, 64)                0
_________________________________________________________________
dense_1 (Dense)              (None, 50)                3250
_________________________________________________________________
dense_2 (Dense)              (None, 10)                510
=================================================================
Total params: 58,160
Trainable params: 58,160
Non-trainable params: 0
_________________________________________________________________
```

In [37]:

```
from datetime import datetime
model.compile(loss='sparse_categorical_crossentropy',optimizer='adam',metrics=['accuracy',f1_scores])
logdir = "logs/fit/" + datetime.now().strftime("%Y%m%d-%H%M%S")
```

In [38]:

```
tensorboard_callback = tf.keras.callbacks.TensorBoard(log_dir=logdir)
model.fit(X_train_spectrogram,y_train,validation_data=(X_test_spectrogram,y_test),batch_size=10,epochs=200,callbacks=[tensorboard_
callback])
```

```
Epoch 1/200
140/140 [==============================] - 3s 14ms/step - loss: 2.2363 - accuracy: 0.1872 - f1_scores: 0.1872 - val_loss: 1.8299 -
val_accuracy: 0.4383 - val_f1_scores: 0.4383
Epoch 2/200
140/140 [==============================] - 1s 9ms/step - loss: 1.7415 - accuracy: 0.4370 - f1_scores: 0.4370 - val_loss: 1.5125 -
```

```
val_accuracy: 0.5067 - val_f1_scores: 0.5067
Epoch 3/200
140/140 [==============================] - 1s 11ms/step - loss: 1.4716 - accuracy: 0.5215 - f1_scores: 0.5215 - val_loss: 1.3133 -
val_accuracy: 0.5600 - val_f1_scores: 0.5600
Epoch 4/200
140/140 [==============================] - 1s 10ms/step - loss: 1.2977 - accuracy: 0.5604 - f1_scores: 0.5604 - val_loss: 1.2221 -
val_accuracy: 0.6150 - val_f1_scores: 0.6150
Epoch 5/200
140/140 [==============================] - 1s 10ms/step - loss: 1.1500 - accuracy: 0.6293 - f1_scores: 0.6293 - val_loss: 1.0669 -
val_accuracy: 0.6433 - val_f1_scores: 0.6433
Epoch 6/200
140/140 [==============================] - 2s 12ms/step - loss: 1.1296 - accuracy: 0.6285 - f1_scores: 0.6285 - val_loss: 0.9708 -
val_accuracy: 0.7100 - val_f1_scores: 0.7100
Epoch 7/200
140/140 [==============================] - 1s 9ms/step - loss: 0.9856 - accuracy: 0.6893 - f1_scores: 0.6893 - val_loss: 0.9232 -
val_accuracy: 0.7133 - val_f1_scores: 0.7133
Epoch 8/200
140/140 [==============================] - 1s 9ms/step - loss: 0.9193 - accuracy: 0.7164 - f1_scores: 0.7164 - val_loss: 0.8270 -
val_accuracy: 0.7667 - val_f1_scores: 0.7667
Epoch 9/200
140/140 [==============================] - 1s 10ms/step - loss: 0.8394 - accuracy: 0.7243 - f1_scores: 0.7243 - val_loss: 0.8287 -
val_accuracy: 0.7467 - val_f1_scores: 0.7467
Epoch 10/200
140/140 [==============================] - 1s 10ms/step - loss: 0.7801 - accuracy: 0.7603 - f1_scores: 0.7603 - val_loss: 0.7703 -
val_accuracy: 0.7567 - val_f1_scores: 0.7567
Epoch 11/200
140/140 [==============================] - 2s 11ms/step - loss: 0.7735 - accuracy: 0.7270 - f1_scores: 0.7270 - val_loss: 0.6870 -
val_accuracy: 0.8067 - val_f1_scores: 0.8067
Epoch 12/200
140/140 [==============================] - 1s 11ms/step - loss: 0.6858 - accuracy: 0.7706 - f1_scores: 0.7706 - val_loss: 0.6515 -
val_accuracy: 0.8167 - val_f1_scores: 0.8167
Epoch 13/200
140/140 [==============================] - 2s 12ms/step - loss: 0.6748 - accuracy: 0.7764 - f1_scores: 0.7764 - val_loss: 0.6766 -
val_accuracy: 0.7817 - val_f1_scores: 0.7817
Epoch 14/200
140/140 [==============================] - 2s 11ms/step - loss: 0.6513 - accuracy: 0.8040 - f1_scores: 0.8040 - val_loss: 0.6883 -
val_accuracy: 0.8017 - val_f1_scores: 0.8017
Epoch 15/200
140/140 [==============================] - 1s 10ms/step - loss: 0.6278 - accuracy: 0.8042 - f1_scores: 0.8042 - val_loss: 0.6166 -
val_accuracy: 0.8333 - val_f1_scores: 0.8333
Epoch 16/200
140/140 [==============================] - 1s 9ms/step - loss: 0.6397 - accuracy: 0.8013 - f1_scores: 0.8013 - val_loss: 0.5557 -
val_accuracy: 0.8533 - val_f1_scores: 0.8533
Epoch 17/200
140/140 [==============================] - 1s 11ms/step - loss: 0.5777 - accuracy: 0.8081 - f1_scores: 0.8081 - val_loss: 0.5212 -
val_accuracy: 0.8533 - val_f1_scores: 0.8533
Epoch 18/200
140/140 [==============================] - 1s 10ms/step - loss: 0.5418 - accuracy: 0.8391 - f1_scores: 0.8391 - val_loss: 0.5985 -
val_accuracy: 0.7883 - val_f1_scores: 0.7883
Epoch 19/200
140/140 [==============================] - 1s 10ms/step - loss: 0.5326 - accuracy: 0.8228 - f1_scores: 0.8228 - val_loss: 0.5302 -
```

```
140/140 [==============================] - 1s 10ms/step - loss: 0.5520 - accuracy: 0.8228 - f1_scores: 0.8228 - val_loss: 0.5302 -
val_accuracy: 0.8367 - val_f1_scores: 0.8367
Epoch 20/200
140/140 [==============================] - 1s 9ms/step - loss: 0.5446 - accuracy: 0.8158 - f1_scores: 0.8158 - val_loss: 0.5230 -
val_accuracy: 0.8283 - val_f1_scores: 0.8283
Epoch 21/200
140/140 [==============================] - 1s 9ms/step - loss: 0.5071 - accuracy: 0.8388 - f1_scores: 0.8388 - val_loss: 0.5081 -
val_accuracy: 0.8450 - val_f1_scores: 0.8450
Epoch 22/200
140/140 [==============================] - 2s 12ms/step - loss: 0.5078 - accuracy: 0.8490 - f1_scores: 0.8490 - val_loss: 0.5015 -
val_accuracy: 0.8400 - val_f1_scores: 0.8400
Epoch 23/200
140/140 [==============================] - 1s 10ms/step - loss: 0.4672 - accuracy: 0.8829 - f1_scores: 0.8829 - val_loss: 0.4690 -
val_accuracy: 0.8600 - val_f1_scores: 0.8600
Epoch 24/200
140/140 [==============================] - 1s 10ms/step - loss: 0.4778 - accuracy: 0.8605 - f1_scores: 0.8605 - val_loss: 0.4542 -
val_accuracy: 0.8533 - val_f1_scores: 0.8533
Epoch 25/200
140/140 [==============================] - 1s 10ms/step - loss: 0.4423 - accuracy: 0.8644 - f1_scores: 0.8644 - val_loss: 0.4477 -
val_accuracy: 0.8717 - val_f1_scores: 0.8717
Epoch 26/200
140/140 [==============================] - 1s 9ms/step - loss: 0.4198 - accuracy: 0.8795 - f1_scores: 0.8795 - val_loss: 0.4446 -
val_accuracy: 0.8633 - val_f1_scores: 0.8633
Epoch 27/200
140/140 [==============================] - 1s 10ms/step - loss: 0.4334 - accuracy: 0.8676 - f1_scores: 0.8676 - val_loss: 0.4154 -
val_accuracy: 0.8667 - val_f1_scores: 0.8667
Epoch 28/200
140/140 [==============================] - 1s 10ms/step - loss: 0.4230 - accuracy: 0.8764 - f1_scores: 0.8764 - val_loss: 0.4025 -
val_accuracy: 0.8667 - val_f1_scores: 0.8667
Epoch 29/200
140/140 [==============================] - 1s 9ms/step - loss: 0.3799 - accuracy: 0.8918 - f1_scores: 0.8918 - val_loss: 0.4085 -
val_accuracy: 0.8717 - val_f1_scores: 0.8717
Epoch 30/200
140/140 [==============================] - 1s 11ms/step - loss: 0.3629 - accuracy: 0.9064 - f1_scores: 0.9064 - val_loss: 0.4755 -
val_accuracy: 0.8433 - val_f1_scores: 0.8433
Epoch 31/200
140/140 [==============================] - 1s 11ms/step - loss: 0.3819 - accuracy: 0.8933 - f1_scores: 0.8933 - val_loss: 0.4022 -
val_accuracy: 0.8683 - val_f1_scores: 0.8683
Epoch 32/200
140/140 [==============================] - 1s 10ms/step - loss: 0.3762 - accuracy: 0.8826 - f1_scores: 0.8826 - val_loss: 0.3723 -
val_accuracy: 0.8783 - val_f1_scores: 0.8783
Epoch 33/200
140/140 [==============================] - 1s 10ms/step - loss: 0.3497 - accuracy: 0.8933 - f1_scores: 0.8933 - val_loss: 0.3983 -
val_accuracy: 0.8783 - val_f1_scores: 0.8783
Epoch 34/200
140/140 [==============================] - 1s 11ms/step - loss: 0.3731 - accuracy: 0.8865 - f1_scores: 0.8865 - val_loss: 0.3846 -
val_accuracy: 0.8917 - val_f1_scores: 0.8917
Epoch 35/200
140/140 [==============================] - 1s 10ms/step - loss: 0.3429 - accuracy: 0.9002 - f1_scores: 0.9002 - val_loss: 0.4073 -
val_accuracy: 0.8600 - val_f1_scores: 0.8600
Epoch 36/200
```

```
140/140 [==============================] - 2s 11ms/step - loss: 0.3751 - accuracy: 0.8921 - f1_scores: 0.8921 - val_loss: 0.4191 -
val_accuracy: 0.8567 - val_f1_scores: 0.8567
Epoch 37/200
140/140 [==============================] - 2s 16ms/step - loss: 0.3473 - accuracy: 0.9143 - f1_scores: 0.9143 - val_loss: 0.3479 -
val_accuracy: 0.9083 - val_f1_scores: 0.9083
Epoch 38/200
140/140 [==============================] - 2s 11ms/step - loss: 0.3375 - accuracy: 0.9065 - f1_scores: 0.9065 - val_loss: 0.3623 -
val_accuracy: 0.8950 - val_f1_scores: 0.8950
Epoch 39/200
140/140 [==============================] - 1s 9ms/step - loss: 0.3115 - accuracy: 0.9096 - f1_scores: 0.9096 - val_loss: 0.3488 -
val_accuracy: 0.9000 - val_f1_scores: 0.9000
Epoch 40/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2989 - accuracy: 0.9148 - f1_scores: 0.9148 - val_loss: 0.3478 -
val_accuracy: 0.9000 - val_f1_scores: 0.9000
Epoch 41/200
140/140 [==============================] - 1s 10ms/step - loss: 0.2875 - accuracy: 0.9202 - f1_scores: 0.9202 - val_loss: 0.3567 -
val_accuracy: 0.8833 - val_f1_scores: 0.8833
Epoch 42/200
140/140 [==============================] - 1s 10ms/step - loss: 0.3021 - accuracy: 0.9212 - f1_scores: 0.9212 - val_loss: 0.3260 -
val_accuracy: 0.9000 - val_f1_scores: 0.9000
Epoch 43/200
140/140 [==============================] - 1s 9ms/step - loss: 0.3030 - accuracy: 0.9115 - f1_scores: 0.9115 - val_loss: 0.3559 -
val_accuracy: 0.8817 - val_f1_scores: 0.8817
Epoch 44/200
140/140 [==============================] - 1s 9ms/step - loss: 0.3186 - accuracy: 0.9029 - f1_scores: 0.9029 - val_loss: 0.3536 -
val_accuracy: 0.8783 - val_f1_scores: 0.8783
Epoch 45/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2680 - accuracy: 0.9197 - f1_scores: 0.9197 - val_loss: 0.3487 -
val_accuracy: 0.8833 - val_f1_scores: 0.8833
Epoch 46/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2540 - accuracy: 0.9369 - f1_scores: 0.9369 - val_loss: 0.3504 -
val_accuracy: 0.8983 - val_f1_scores: 0.8983
Epoch 47/200
140/140 [==============================] - 1s 10ms/step - loss: 0.2787 - accuracy: 0.9252 - f1_scores: 0.9252 - val_loss: 0.3347 -
val_accuracy: 0.8967 - val_f1_scores: 0.8967
Epoch 48/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2800 - accuracy: 0.9287 - f1_scores: 0.9287 - val_loss: 0.3386 -
val_accuracy: 0.8883 - val_f1_scores: 0.8883
Epoch 49/200
140/140 [==============================] - 2s 11ms/step - loss: 0.2956 - accuracy: 0.9167 - f1_scores: 0.9167 - val_loss: 0.3014 -
val_accuracy: 0.9133 - val_f1_scores: 0.9133
Epoch 50/200
140/140 [==============================] - 2s 15ms/step - loss: 0.2583 - accuracy: 0.9261 - f1_scores: 0.9261 - val_loss: 0.2938 -
val_accuracy: 0.9033 - val_f1_scores: 0.9033
Epoch 51/200
140/140 [==============================] - 2s 14ms/step - loss: 0.2625 - accuracy: 0.9321 - f1_scores: 0.9321 - val_loss: 0.3366 -
val_accuracy: 0.8867 - val_f1_scores: 0.8867
Epoch 52/200
140/140 [==============================] - 2s 14ms/step - loss: 0.2549 - accuracy: 0.9229 - f1_scores: 0.9229 - val_loss: 0.3148 -
val_accuracy: 0.9017 - val_f1_scores: 0.9017
Epoch 53/200
```

```
Epoch 53/200
140/140 [==============================] - 2s 13ms/step - loss: 0.2524 - accuracy: 0.9295 - f1_scores: 0.9295 - val_loss: 0.3095 -
val_accuracy: 0.8950 - val_f1_scores: 0.8950
Epoch 54/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2713 - accuracy: 0.9071 - f1_scores: 0.9071 - val_loss: 0.3306 -
val_accuracy: 0.9083 - val_f1_scores: 0.9083
Epoch 55/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2342 - accuracy: 0.9428 - f1_scores: 0.9428 - val_loss: 0.2892 -
val_accuracy: 0.9050 - val_f1_scores: 0.9050
Epoch 56/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2480 - accuracy: 0.9310 - f1_scores: 0.9310 - val_loss: 0.3094 -
val_accuracy: 0.9067 - val_f1_scores: 0.9067
Epoch 57/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2763 - accuracy: 0.9239 - f1_scores: 0.9239 - val_loss: 0.3314 -
val_accuracy: 0.8900 - val_f1_scores: 0.8900
Epoch 58/200
140/140 [==============================] - 2s 14ms/step - loss: 0.2610 - accuracy: 0.9273 - f1_scores: 0.9273 - val_loss: 0.2831 -
val_accuracy: 0.8917 - val_f1_scores: 0.8917
Epoch 59/200
140/140 [==============================] - 2s 15ms/step - loss: 0.2320 - accuracy: 0.9304 - f1_scores: 0.9304 - val_loss: 0.2837 -
val_accuracy: 0.9000 - val_f1_scores: 0.9000
Epoch 60/200
140/140 [==============================] - 2s 13ms/step - loss: 0.2165 - accuracy: 0.9396 - f1_scores: 0.9396 - val_loss: 0.2821 -
val_accuracy: 0.9150 - val_f1_scores: 0.9150
Epoch 61/200
140/140 [==============================] - 2s 13ms/step - loss: 0.2021 - accuracy: 0.9367 - f1_scores: 0.9367 - val_loss: 0.2535 -
val_accuracy: 0.9300 - val_f1_scores: 0.9300
Epoch 62/200
140/140 [==============================] - 2s 14ms/step - loss: 0.2247 - accuracy: 0.9452 - f1_scores: 0.9452 - val_loss: 0.2928 -
val_accuracy: 0.9083 - val_f1_scores: 0.9083
Epoch 63/200
140/140 [==============================] - 2s 13ms/step - loss: 0.2187 - accuracy: 0.9312 - f1_scores: 0.9312 - val_loss: 0.3144 -
val_accuracy: 0.8917 - val_f1_scores: 0.8917
Epoch 64/200
140/140 [==============================] - 2s 14ms/step - loss: 0.2427 - accuracy: 0.9313 - f1_scores: 0.9313 - val_loss: 0.2892 -
val_accuracy: 0.9117 - val_f1_scores: 0.9117
Epoch 65/200
140/140 [==============================] - 2s 13ms/step - loss: 0.2120 - accuracy: 0.9420 - f1_scores: 0.9420 - val_loss: 0.3027 -
val_accuracy: 0.9083 - val_f1_scores: 0.9083
Epoch 66/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2276 - accuracy: 0.9339 - f1_scores: 0.9339 - val_loss: 0.3104 -
val_accuracy: 0.8983 - val_f1_scores: 0.8983
Epoch 67/200
140/140 [==============================] - 1s 10ms/step - loss: 0.2077 - accuracy: 0.9413 - f1_scores: 0.9413 - val_loss: 0.2684 -
val_accuracy: 0.9117 - val_f1_scores: 0.9117
Epoch 68/200
140/140 [==============================] - 1s 10ms/step - loss: 0.2115 - accuracy: 0.9485 - f1_scores: 0.9485 - val_loss: 0.2902 -
val_accuracy: 0.9067 - val_f1_scores: 0.9067
Epoch 69/200
140/140 [==============================] - 2s 11ms/step - loss: 0.1978 - accuracy: 0.9432 - f1_scores: 0.9432 - val_loss: 0.2592 -
val_accuracy: 0.9233 - val_f1_scores: 0.9233
```

```
Epoch 70/200
140/140 [==============================] - 2s 14ms/step - loss: 0.2134 - accuracy: 0.9323 - f1_scores: 0.9323 - val_loss: 0.2620 -
val_accuracy: 0.9150 - val_f1_scores: 0.9150
Epoch 71/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1961 - accuracy: 0.9480 - f1_scores: 0.9480 - val_loss: 0.2961 -
val_accuracy: 0.9067 - val_f1_scores: 0.9067
Epoch 72/200
140/140 [==============================] - 2s 13ms/step - loss: 0.2159 - accuracy: 0.9426 - f1_scores: 0.9426 - val_loss: 0.2789 -
val_accuracy: 0.9017 - val_f1_scores: 0.9017
Epoch 73/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1975 - accuracy: 0.9473 - f1_scores: 0.9473 - val_loss: 0.2715 -
val_accuracy: 0.9033 - val_f1_scores: 0.9033
Epoch 74/200
140/140 [==============================] - 2s 16ms/step - loss: 0.2002 - accuracy: 0.9454 - f1_scores: 0.9454 - val_loss: 0.2833 -
val_accuracy: 0.9100 - val_f1_scores: 0.9100
Epoch 75/200
140/140 [==============================] - 2s 14ms/step - loss: 0.2311 - accuracy: 0.9415 - f1_scores: 0.9415 - val_loss: 0.2621 -
val_accuracy: 0.9117 - val_f1_scores: 0.9117
Epoch 76/200
140/140 [==============================] - 2s 13ms/step - loss: 0.2220 - accuracy: 0.9221 - f1_scores: 0.9221 - val_loss: 0.2806 -
val_accuracy: 0.9033 - val_f1_scores: 0.9033
Epoch 77/200
140/140 [==============================] - 1s 10ms/step - loss: 0.2430 - accuracy: 0.9234 - f1_scores: 0.9234 - val_loss: 0.3139 -
val_accuracy: 0.9050 - val_f1_scores: 0.9050
Epoch 78/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2246 - accuracy: 0.9396 - f1_scores: 0.9396 - val_loss: 0.3298 -
val_accuracy: 0.8783 - val_f1_scores: 0.8783
Epoch 79/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2145 - accuracy: 0.9365 - f1_scores: 0.9365 - val_loss: 0.2810 -
val_accuracy: 0.9150 - val_f1_scores: 0.9150
Epoch 80/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1887 - accuracy: 0.9463 - f1_scores: 0.9463 - val_loss: 0.2693 -
val_accuracy: 0.9267 - val_f1_scores: 0.9267
Epoch 81/200
140/140 [==============================] - 2s 13ms/step - loss: 0.2161 - accuracy: 0.9418 - f1_scores: 0.9418 - val_loss: 0.2882 -
val_accuracy: 0.9133 - val_f1_scores: 0.9133
Epoch 82/200
140/140 [==============================] - 2s 14ms/step - loss: 0.2298 - accuracy: 0.9344 - f1_scores: 0.9344 - val_loss: 0.2729 -
val_accuracy: 0.9083 - val_f1_scores: 0.9083
Epoch 83/200
140/140 [==============================] - 2s 15ms/step - loss: 0.2058 - accuracy: 0.9490 - f1_scores: 0.9490 - val_loss: 0.2839 -
val_accuracy: 0.9183 - val_f1_scores: 0.9183
Epoch 84/200
140/140 [==============================] - 2s 13ms/step - loss: 0.2135 - accuracy: 0.9408 - f1_scores: 0.9408 - val_loss: 0.2725 -
val_accuracy: 0.9217 - val_f1_scores: 0.9217
Epoch 85/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1889 - accuracy: 0.9546 - f1_scores: 0.9546 - val_loss: 0.2907 -
val_accuracy: 0.8983 - val_f1_scores: 0.8983
Epoch 86/200
140/140 [==============================] - 2s 14ms/step - loss: 0.2068 - accuracy: 0.9386 - f1_scores: 0.9386 - val_loss: 0.2833 -
val_accuracy: 0.9150 - val_f1_scores: 0.9150
```

```
Epoch 87/200
140/140 [==============================] - 2s 13ms/step - loss: 0.1818 - accuracy: 0.9543 - f1_scores: 0.9543 - val_loss: 0.2694 -
val_accuracy: 0.9067 - val_f1_scores: 0.9067
Epoch 88/200
140/140 [==============================] - 2s 13ms/step - loss: 0.2071 - accuracy: 0.9487 - f1_scores: 0.9487 - val_loss: 0.2820 -
val_accuracy: 0.8950 - val_f1_scores: 0.8950
Epoch 89/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2044 - accuracy: 0.9464 - f1_scores: 0.9464 - val_loss: 0.2661 -
val_accuracy: 0.9217 - val_f1_scores: 0.9217
Epoch 90/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1801 - accuracy: 0.9542 - f1_scores: 0.9542 - val_loss: 0.2775 -
val_accuracy: 0.9100 - val_f1_scores: 0.9100
Epoch 91/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2034 - accuracy: 0.9487 - f1_scores: 0.9487 - val_loss: 0.2609 -
val_accuracy: 0.9233 - val_f1_scores: 0.9233
Epoch 92/200
140/140 [==============================] - 2s 12ms/step - loss: 0.1935 - accuracy: 0.9520 - f1_scores: 0.9520 - val_loss: 0.2988 -
val_accuracy: 0.9017 - val_f1_scores: 0.9017
Epoch 93/200
140/140 [==============================] - 2s 14ms/step - loss: 0.2061 - accuracy: 0.9485 - f1_scores: 0.9485 - val_loss: 0.2740 -
val_accuracy: 0.9150 - val_f1_scores: 0.9150
Epoch 94/200
140/140 [==============================] - 2s 13ms/step - loss: 0.1935 - accuracy: 0.9573 - f1_scores: 0.9573 - val_loss: 0.2817 -
val_accuracy: 0.9183 - val_f1_scores: 0.9183
Epoch 95/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1866 - accuracy: 0.9490 - f1_scores: 0.9490 - val_loss: 0.2562 -
val_accuracy: 0.9183 - val_f1_scores: 0.9183
Epoch 96/200
140/140 [==============================] - 2s 13ms/step - loss: 0.1880 - accuracy: 0.9449 - f1_scores: 0.9449 - val_loss: 0.2894 -
val_accuracy: 0.9050 - val_f1_scores: 0.9050
Epoch 97/200
140/140 [==============================] - 2s 13ms/step - loss: 0.2223 - accuracy: 0.9442 - f1_scores: 0.9442 - val_loss: 0.2457 -
val_accuracy: 0.9283 - val_f1_scores: 0.9283
Epoch 98/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1924 - accuracy: 0.9472 - f1_scores: 0.9472 - val_loss: 0.2902 -
val_accuracy: 0.9167 - val_f1_scores: 0.9167
Epoch 99/200
140/140 [==============================] - 2s 15ms/step - loss: 0.1784 - accuracy: 0.9488 - f1_scores: 0.9488 - val_loss: 0.2837 -
val_accuracy: 0.9050 - val_f1_scores: 0.9050
Epoch 100/200
140/140 [==============================] - 2s 11ms/step - loss: 0.2033 - accuracy: 0.9393 - f1_scores: 0.9393 - val_loss: 0.2274 -
val_accuracy: 0.9333 - val_f1_scores: 0.9333
Epoch 101/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1690 - accuracy: 0.9459 - f1_scores: 0.9459 - val_loss: 0.2469 -
val_accuracy: 0.9283 - val_f1_scores: 0.9283
Epoch 102/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1907 - accuracy: 0.9471 - f1_scores: 0.9471 - val_loss: 0.2631 -
val_accuracy: 0.9217 - val_f1_scores: 0.9217
Epoch 103/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1719 - accuracy: 0.9537 - f1_scores: 0.9537 - val_loss: 0.2546 -
```

```
val_accuracy: 0.9267 - val_f1_scores: 0.9267
Epoch 104/200
140/140 [==============================] - 2s 13ms/step - loss: 0.1676 - accuracy: 0.9543 - f1_scores: 0.9543 - val_loss: 0.2451 -
val_accuracy: 0.9283 - val_f1_scores: 0.9283
Epoch 105/200
140/140 [==============================] - 2s 13ms/step - loss: 0.1867 - accuracy: 0.9396 - f1_scores: 0.9396 - val_loss: 0.2863 -
val_accuracy: 0.9150 - val_f1_scores: 0.9150
Epoch 106/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1572 - accuracy: 0.9541 - f1_scores: 0.9541 - val_loss: 0.2575 -
val_accuracy: 0.9167 - val_f1_scores: 0.9167
Epoch 107/200
140/140 [==============================] - 2s 15ms/step - loss: 0.1594 - accuracy: 0.9550 - f1_scores: 0.9550 - val_loss: 0.2951 -
val_accuracy: 0.9067 - val_f1_scores: 0.9067
Epoch 108/200
140/140 [==============================] - 2s 13ms/step - loss: 0.1944 - accuracy: 0.9447 - f1_scores: 0.9447 - val_loss: 0.2654 -
val_accuracy: 0.9183 - val_f1_scores: 0.9183
Epoch 109/200
140/140 [==============================] - 2s 16ms/step - loss: 0.1965 - accuracy: 0.9480 - f1_scores: 0.9480 - val_loss: 0.2718 -
val_accuracy: 0.9150 - val_f1_scores: 0.9150
Epoch 110/200
140/140 [==============================] - 2s 15ms/step - loss: 0.1853 - accuracy: 0.9535 - f1_scores: 0.9535 - val_loss: 0.2569 -
val_accuracy: 0.9150 - val_f1_scores: 0.9150
Epoch 111/200
140/140 [==============================] - 2s 12ms/step - loss: 0.1581 - accuracy: 0.9597 - f1_scores: 0.9597 - val_loss: 0.2674 -
val_accuracy: 0.9167 - val_f1_scores: 0.9167
Epoch 112/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1740 - accuracy: 0.9487 - f1_scores: 0.9487 - val_loss: 0.2431 -
val_accuracy: 0.9317 - val_f1_scores: 0.9317
Epoch 113/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1609 - accuracy: 0.9503 - f1_scores: 0.9503 - val_loss: 0.2491 -
val_accuracy: 0.9267 - val_f1_scores: 0.9267
Epoch 114/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1882 - accuracy: 0.9370 - f1_scores: 0.9370 - val_loss: 0.2606 -
val_accuracy: 0.9250 - val_f1_scores: 0.9250
Epoch 115/200
140/140 [==============================] - 2s 12ms/step - loss: 0.1612 - accuracy: 0.9592 - f1_scores: 0.9592 - val_loss: 0.2715 -
val_accuracy: 0.9183 - val_f1_scores: 0.9183
Epoch 116/200
140/140 [==============================] - 2s 15ms/step - loss: 0.1748 - accuracy: 0.9495 - f1_scores: 0.9495 - val_loss: 0.2619 -
val_accuracy: 0.9167 - val_f1_scores: 0.9167
Epoch 117/200
140/140 [==============================] - 2s 13ms/step - loss: 0.1770 - accuracy: 0.9442 - f1_scores: 0.9442 - val_loss: 0.3022 -
val_accuracy: 0.8950 - val_f1_scores: 0.8950
Epoch 118/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1701 - accuracy: 0.9522 - f1_scores: 0.9522 - val_loss: 0.2562 -
val_accuracy: 0.9183 - val_f1_scores: 0.9183
Epoch 119/200
140/140 [==============================] - 2s 13ms/step - loss: 0.1914 - accuracy: 0.9415 - f1_scores: 0.9415 - val_loss: 0.2650 -
val_accuracy: 0.9000 - val_f1_scores: 0.9000
Epoch 120/200
140/140 [==============================] - 2s 13ms/step - loss: 0.1439 - accuracy: 0.9581 - f1_scores: 0.9581 - val_loss: 0.2804 -
```

```
val_accuracy: 0.9050 - val_f1_scores: 0.9050
Epoch 121/200
140/140 [==============================] - 2s 15ms/step - loss: 0.1868 - accuracy: 0.9474 - f1_scores: 0.9474 - val_loss: 0.2541 -
val_accuracy: 0.9200 - val_f1_scores: 0.9200
Epoch 122/200
140/140 [==============================] - 2s 13ms/step - loss: 0.1505 - accuracy: 0.9582 - f1_scores: 0.9582 - val_loss: 0.2457 -
val_accuracy: 0.9233 - val_f1_scores: 0.9233
Epoch 123/200
140/140 [==============================] - 2s 13ms/step - loss: 0.1478 - accuracy: 0.9631 - f1_scores: 0.9631 - val_loss: 0.2714 -
val_accuracy: 0.9183 - val_f1_scores: 0.9183
Epoch 124/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1562 - accuracy: 0.9622 - f1_scores: 0.9622 - val_loss: 0.2610 -
val_accuracy: 0.9217 - val_f1_scores: 0.9217
Epoch 125/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1654 - accuracy: 0.9587 - f1_scores: 0.9587 - val_loss: 0.2490 -
val_accuracy: 0.9183 - val_f1_scores: 0.9183
Epoch 126/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1563 - accuracy: 0.9621 - f1_scores: 0.9621 - val_loss: 0.2411 -
val_accuracy: 0.9283 - val_f1_scores: 0.9283
Epoch 127/200
140/140 [==============================] - 2s 12ms/step - loss: 0.1278 - accuracy: 0.9701 - f1_scores: 0.9701 - val_loss: 0.2449 -
val_accuracy: 0.9250 - val_f1_scores: 0.9250
Epoch 128/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1345 - accuracy: 0.9645 - f1_scores: 0.9645 - val_loss: 0.2466 -
val_accuracy: 0.9217 - val_f1_scores: 0.9217
Epoch 129/200
140/140 [==============================] - 2s 13ms/step - loss: 0.1608 - accuracy: 0.9604 - f1_scores: 0.9604 - val_loss: 0.2636 -
val_accuracy: 0.9150 - val_f1_scores: 0.9150
Epoch 130/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1761 - accuracy: 0.9414 - f1_scores: 0.9414 - val_loss: 0.3072 -
val_accuracy: 0.8967 - val_f1_scores: 0.8967
Epoch 131/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1608 - accuracy: 0.9548 - f1_scores: 0.9548 - val_loss: 0.2427 -
val_accuracy: 0.9250 - val_f1_scores: 0.9250
Epoch 132/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1386 - accuracy: 0.9697 - f1_scores: 0.9697 - val_loss: 0.2788 -
val_accuracy: 0.9067 - val_f1_scores: 0.9067
Epoch 133/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1385 - accuracy: 0.9604 - f1_scores: 0.9604 - val_loss: 0.2360 -
val_accuracy: 0.9233 - val_f1_scores: 0.9233
Epoch 134/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1441 - accuracy: 0.9628 - f1_scores: 0.9628 - val_loss: 0.2620 -
val_accuracy: 0.9133 - val_f1_scores: 0.9133
Epoch 135/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1658 - accuracy: 0.9553 - f1_scores: 0.9553 - val_loss: 0.2878 -
val_accuracy: 0.9033 - val_f1_scores: 0.9033
Epoch 136/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1778 - accuracy: 0.9590 - f1_scores: 0.9590 - val_loss: 0.2616 -
val_accuracy: 0.9083 - val_f1_scores: 0.9083
Epoch 137/200
```

```
140/140 [==============================] - 1s 9ms/step - loss: 0.1497 - accuracy: 0.9571 - f1_scores: 0.9571 - val_loss: 0.2517 -
val_accuracy: 0.9183 - val_f1_scores: 0.9183
Epoch 138/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1399 - accuracy: 0.9645 - f1_scores: 0.9645 - val_loss: 0.2407 -
val_accuracy: 0.9183 - val_f1_scores: 0.9183
Epoch 139/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1203 - accuracy: 0.9731 - f1_scores: 0.9731 - val_loss: 0.2545 -
val_accuracy: 0.9233 - val_f1_scores: 0.9233
Epoch 140/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1738 - accuracy: 0.9415 - f1_scores: 0.9415 - val_loss: 0.2398 -
val_accuracy: 0.9167 - val_f1_scores: 0.9167
Epoch 141/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1406 - accuracy: 0.9559 - f1_scores: 0.9559 - val_loss: 0.2626 -
val_accuracy: 0.9117 - val_f1_scores: 0.9117
Epoch 142/200
140/140 [==============================] - 2s 13ms/step - loss: 0.1926 - accuracy: 0.9302 - f1_scores: 0.9302 - val_loss: 0.2699 -
val_accuracy: 0.9133 - val_f1_scores: 0.9133
Epoch 143/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1558 - accuracy: 0.9545 - f1_scores: 0.9545 - val_loss: 0.2945 -
val_accuracy: 0.8983 - val_f1_scores: 0.8983
Epoch 144/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1813 - accuracy: 0.9518 - f1_scores: 0.9518 - val_loss: 0.2459 -
val_accuracy: 0.9183 - val_f1_scores: 0.9183
Epoch 145/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1667 - accuracy: 0.9530 - f1_scores: 0.9530 - val_loss: 0.2718 -
val_accuracy: 0.9183 - val_f1_scores: 0.9183
Epoch 146/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1641 - accuracy: 0.9470 - f1_scores: 0.9470 - val_loss: 0.2621 -
val_accuracy: 0.9200 - val_f1_scores: 0.9200
Epoch 147/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1613 - accuracy: 0.9562 - f1_scores: 0.9562 - val_loss: 0.2682 -
val_accuracy: 0.9100 - val_f1_scores: 0.9100
Epoch 148/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1682 - accuracy: 0.9530 - f1_scores: 0.9530 - val_loss: 0.2502 -
val_accuracy: 0.9250 - val_f1_scores: 0.9250
Epoch 149/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1528 - accuracy: 0.9550 - f1_scores: 0.9550 - val_loss: 0.2803 -
val_accuracy: 0.9117 - val_f1_scores: 0.9117
Epoch 150/200
140/140 [==============================] - 2s 13ms/step - loss: 0.1728 - accuracy: 0.9518 - f1_scores: 0.9518 - val_loss: 0.2734 -
val_accuracy: 0.9133 - val_f1_scores: 0.9133
Epoch 151/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1733 - accuracy: 0.9439 - f1_scores: 0.9439 - val_loss: 0.2894 -
val_accuracy: 0.8950 - val_f1_scores: 0.8950
Epoch 152/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1499 - accuracy: 0.9523 - f1_scores: 0.9523 - val_loss: 0.2959 -
val_accuracy: 0.9050 - val_f1_scores: 0.9050
Epoch 153/200
140/140 [==============================] - 2s 15ms/step - loss: 0.1625 - accuracy: 0.9509 - f1_scores: 0.9509 - val_loss: 0.2447 -
val_accuracy: 0.9250 - val_f1_scores: 0.9250
Epoch 154/200
```

```
140/140 [==============================] - 2s 14ms/step - loss: 0.1571 - accuracy: 0.9486 - f1_scores: 0.9486 - val_loss: 0.2344 - val_accuracy: 0.9233 - val_f1_scores: 0.9233
Epoch 155/200
140/140 [==============================] - 2s 15ms/step - loss: 0.1540 - accuracy: 0.9524 - f1_scores: 0.9524 - val_loss: 0.2695 - val_accuracy: 0.9083 - val_f1_scores: 0.9083
Epoch 156/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1468 - accuracy: 0.9681 - f1_scores: 0.9681 - val_loss: 0.2643 - val_accuracy: 0.9050 - val_f1_scores: 0.9050
Epoch 157/200
140/140 [==============================] - 2s 13ms/step - loss: 0.1799 - accuracy: 0.9420 - f1_scores: 0.9420 - val_loss: 0.2601 - val_accuracy: 0.9167 - val_f1_scores: 0.9167
Epoch 158/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1676 - accuracy: 0.9498 - f1_scores: 0.9498 - val_loss: 0.2366 - val_accuracy: 0.9217 - val_f1_scores: 0.9217
Epoch 159/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1517 - accuracy: 0.9489 - f1_scores: 0.9489 - val_loss: 0.2544 - val_accuracy: 0.9217 - val_f1_scores: 0.9217
Epoch 160/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1528 - accuracy: 0.9581 - f1_scores: 0.9581 - val_loss: 0.2525 - val_accuracy: 0.9167 - val_f1_scores: 0.9167
Epoch 161/200
140/140 [==============================] - 2s 11ms/step - loss: 0.1533 - accuracy: 0.9593 - f1_scores: 0.9593 - val_loss: 0.2665 - val_accuracy: 0.9133 - val_f1_scores: 0.9133
Epoch 162/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1493 - accuracy: 0.9505 - f1_scores: 0.9505 - val_loss: 0.2617 - val_accuracy: 0.9217 - val_f1_scores: 0.9217
Epoch 163/200
140/140 [==============================] - 2s 13ms/step - loss: 0.1634 - accuracy: 0.9503 - f1_scores: 0.9503 - val_loss: 0.2581 - val_accuracy: 0.9100 - val_f1_scores: 0.9100
Epoch 164/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1962 - accuracy: 0.9498 - f1_scores: 0.9498 - val_loss: 0.2308 - val_accuracy: 0.9217 - val_f1_scores: 0.9217
Epoch 165/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1460 - accuracy: 0.9692 - f1_scores: 0.9692 - val_loss: 0.2606 - val_accuracy: 0.9167 - val_f1_scores: 0.9167
Epoch 166/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1619 - accuracy: 0.9535 - f1_scores: 0.9535 - val_loss: 0.2502 - val_accuracy: 0.9150 - val_f1_scores: 0.9150
Epoch 167/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1625 - accuracy: 0.9501 - f1_scores: 0.9501 - val_loss: 0.2527 - val_accuracy: 0.9183 - val_f1_scores: 0.9183
Epoch 168/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1288 - accuracy: 0.9604 - f1_scores: 0.9604 - val_loss: 0.2304 - val_accuracy: 0.9183 - val_f1_scores: 0.9183
Epoch 169/200
140/140 [==============================] - 2s 12ms/step - loss: 0.1481 - accuracy: 0.9691 - f1_scores: 0.9691 - val_loss: 0.2457 - val_accuracy: 0.9033 - val_f1_scores: 0.9033
Epoch 170/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1333 - accuracy: 0.9677 - f1_scores: 0.9677 - val_loss: 0.2571 - val_accuracy: 0.9100 - val_f1_scores: 0.9100
Epoch 171/200
```

```
Epoch 171/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1478 - accuracy: 0.9471 - f1_scores: 0.9471 - val_loss: 0.2383 -
val_accuracy: 0.9250 - val_f1_scores: 0.9250
Epoch 172/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1361 - accuracy: 0.9622 - f1_scores: 0.9622 - val_loss: 0.2285 -
val_accuracy: 0.9183 - val_f1_scores: 0.9183
Epoch 173/200
140/140 [==============================] - 2s 12ms/step - loss: 0.1462 - accuracy: 0.9615 - f1_scores: 0.9615 - val_loss: 0.2435 -
val_accuracy: 0.9150 - val_f1_scores: 0.9150
Epoch 174/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1364 - accuracy: 0.9564 - f1_scores: 0.9564 - val_loss: 0.2399 -
val_accuracy: 0.9217 - val_f1_scores: 0.9217
Epoch 175/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1579 - accuracy: 0.9459 - f1_scores: 0.9459 - val_loss: 0.2352 -
val_accuracy: 0.9217 - val_f1_scores: 0.9217
Epoch 176/200
140/140 [==============================] - 2s 13ms/step - loss: 0.1436 - accuracy: 0.9567 - f1_scores: 0.9567 - val_loss: 0.2254 -
val_accuracy: 0.9283 - val_f1_scores: 0.9283
Epoch 177/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1196 - accuracy: 0.9744 - f1_scores: 0.9744 - val_loss: 0.2292 -
val_accuracy: 0.9267 - val_f1_scores: 0.9267
Epoch 178/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1500 - accuracy: 0.9580 - f1_scores: 0.9580 - val_loss: 0.2571 -
val_accuracy: 0.9150 - val_f1_scores: 0.9150
Epoch 179/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1440 - accuracy: 0.9626 - f1_scores: 0.9626 - val_loss: 0.2467 -
val_accuracy: 0.9183 - val_f1_scores: 0.9183
Epoch 180/200
140/140 [==============================] - 2s 15ms/step - loss: 0.1454 - accuracy: 0.9616 - f1_scores: 0.9616 - val_loss: 0.2388 -
val_accuracy: 0.9250 - val_f1_scores: 0.9250
Epoch 181/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1705 - accuracy: 0.9555 - f1_scores: 0.9555 - val_loss: 0.2454 -
val_accuracy: 0.9233 - val_f1_scores: 0.9233
Epoch 182/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1774 - accuracy: 0.9429 - f1_scores: 0.9429 - val_loss: 0.2339 -
val_accuracy: 0.9250 - val_f1_scores: 0.9250
Epoch 183/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1176 - accuracy: 0.9703 - f1_scores: 0.9703 - val_loss: 0.2271 -
val_accuracy: 0.9283 - val_f1_scores: 0.9283
Epoch 184/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1421 - accuracy: 0.9636 - f1_scores: 0.9636 - val_loss: 0.2180 -
val_accuracy: 0.9167 - val_f1_scores: 0.9167
Epoch 185/200
140/140 [==============================] - 2s 13ms/step - loss: 0.1215 - accuracy: 0.9710 - f1_scores: 0.9710 - val_loss: 0.2564 -
val_accuracy: 0.9217 - val_f1_scores: 0.9217
Epoch 186/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1458 - accuracy: 0.9591 - f1_scores: 0.9591 - val_loss: 0.2552 -
val_accuracy: 0.9183 - val_f1_scores: 0.9183
Epoch 187/200
140/140 [==============================] - 2s 15ms/step - loss: 0.1370 - accuracy: 0.9565 - f1_scores: 0.9565 - val_loss: 0.2327 -
val_accuracy: 0.9250 - val_f1_scores: 0.9250
```

```
Epoch 188/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1113 - accuracy: 0.9743 - f1_scores: 0.9743 - val_loss: 0.2167 - val_accuracy: 0.9350 - val_f1_scores: 0.9350
Epoch 189/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1091 - accuracy: 0.9676 - f1_scores: 0.9676 - val_loss: 0.2248 - val_accuracy: 0.9217 - val_f1_scores: 0.9217
Epoch 190/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1302 - accuracy: 0.9613 - f1_scores: 0.9613 - val_loss: 0.2434 - val_accuracy: 0.9217 - val_f1_scores: 0.9217
Epoch 191/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1361 - accuracy: 0.9628 - f1_scores: 0.9628 - val_loss: 0.2242 - val_accuracy: 0.9250 - val_f1_scores: 0.9250
Epoch 192/200
140/140 [==============================] - 2s 13ms/step - loss: 0.1146 - accuracy: 0.9728 - f1_scores: 0.9728 - val_loss: 0.2351 - val_accuracy: 0.9267 - val_f1_scores: 0.9267
Epoch 193/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1287 - accuracy: 0.9678 - f1_scores: 0.9678 - val_loss: 0.2408 - val_accuracy: 0.9200 - val_f1_scores: 0.9200
Epoch 194/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1231 - accuracy: 0.9596 - f1_scores: 0.9596 - val_loss: 0.2223 - val_accuracy: 0.9300 - val_f1_scores: 0.9300
Epoch 195/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1358 - accuracy: 0.9624 - f1_scores: 0.9624 - val_loss: 0.2337 - val_accuracy: 0.9267 - val_f1_scores: 0.9267
Epoch 196/200
140/140 [==============================] - 2s 12ms/step - loss: 0.1147 - accuracy: 0.9705 - f1_scores: 0.9705 - val_loss: 0.2373 - val_accuracy: 0.9317 - val_f1_scores: 0.9317
Epoch 197/200
140/140 [==============================] - 2s 13ms/step - loss: 0.1045 - accuracy: 0.9753 - f1_scores: 0.9753 - val_loss: 0.2715 - val_accuracy: 0.9233 - val_f1_scores: 0.9233
Epoch 198/200
140/140 [==============================] - 2s 15ms/step - loss: 0.1075 - accuracy: 0.9750 - f1_scores: 0.9750 - val_loss: 0.2282 - val_accuracy: 0.9317 - val_f1_scores: 0.9317
Epoch 199/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1092 - accuracy: 0.9735 - f1_scores: 0.9735 - val_loss: 0.2157 - val_accuracy: 0.9367 - val_f1_scores: 0.9367
Epoch 200/200
140/140 [==============================] - 2s 14ms/step - loss: 0.1038 - accuracy: 0.9756 - f1_scores: 0.9756 - val_loss: 0.2746 - val_accuracy: 0.8983 - val_f1_scores: 0.8983
```

Out[38]:

```
<tensorflow.python.keras.callbacks.History at 0x7f34abcb5198>
```

In [ ]:

```
%tensorboard --logdir=./
```

3. data augmentation

Till now we have done with 2000 samples only. It is very less data. We are giving the process of generating augmented data below.

There are two types of augmentation:
1. time stretching - Time stretching either increases or decreases the length of the file. For time stretching we move the file 30% faster or slower
2. pitch shifting - pitch shifting moves the frequencies higher or lower. For pitch shifting we shift up or down one half-step.

In [40]:

```
## generating augmented data.
def generate_augmented_data(file_path):
    augmented_data = []
    samples = load_wav(file_path,get_duration=False)
    for time_value in [0.7, 1, 1.3]:
        for pitch_value in [-1, 0, 1]:
            time_stretch_data = librosa.effects.time_stretch(samples, rate=time_value)
            final_data = librosa.effects.pitch_shift(time_stretch_data, sr=sample_rate, n_steps=pitch_value)
            augmented_data.append(final_data)
    return augmented_data
```

In [41]:

```
temp_path = df_audio.iloc[0].path
aug_temp = generate_augmented_data(temp_path)
```

In [42]:

```
len(aug_temp)
```

Out[42]:

```
9
```

As discussed above, for one data point, we will get 9 augmented data points.

Split data into train and test (80-20 split)

We have 2000 data points(1600 train points, 400 test points)

Do augmentation only on train data, after augmentation we will get 14400 train points.

do the above steps i.e training with raw data and spectrogram data with augmentation.

In [43]:

```
x_aug=np.zeros((18000,17640))
y_aug=[]
k=0
for i in tqdm(range(df_audio.shape[0])):
  temp_path=df_audio.iloc[i].path
  aug_temp = generate_augmented_data(temp_path)
  for j in aug_temp:
    if len(j)<=17640:
      x_aug[k]=np.array(j.tolist()+[0]*(17640-len(j)))
      y_aug.append(df_audio.iloc[i].label)
    else:
      x_aug[k]=np.array((j.tolist())[:17640])
      y_aug.append(df_audio.iloc[i].label)
    k+=1
```

```
100%|██████████| 2000/2000 [07:43<00:00,  4.32it/s]
```

In [44]:

```
X_aug_train, X_aug_test, y_aug_train, y_aug_test=train_test_split(x_aug,y_aug,test_size=0.20,random_state=45,stratify=y_aug)
```

In [45]:

```
y_aug_train=np.array(y_aug_train)
y_aug_test=np.array(y_aug_test)
```

In [46]:

```
X_aug_train_mask = np.array(mask(X_aug_train))
X_aug_test_mask  = np.array(mask(X_aug_test))
```

```
100%|██████████| 14400/14400 [02:18<00:00, 104.24it/s]
100%|██████████| 3600/3600 [00:32<00:00, 110.87it/s]
```

In [47]:

```
input= Input(shape=(17640,1,),dtype=np.float32)
mask1  = Input(shape=(17640,),dtype='bool')
lstm_output=LSTM(20)(inputs=input,mask=mask1)
Dense1=Dense(10,activation='softmax')(lstm_output)
model= Model(inputs=[input,mask1],outputs=Dense1)
model.summary()
```

```
Model: "model_2"
____________________________________________________________________________________________________
Layer (type)                    Output Shape          Param #     Connected to
====================================================================================================
input_4 (InputLayer)            [(None, 17640, 1)]    0
____________________________________________________________________________________________________
input_5 (InputLayer)            [(None, 17640)]       0
```

```
________________________________________________________________________________________________
lstm_2 (LSTM)                   (None, 20)           1760        input_4[0][0]
                                                                 input_5[0][0]
________________________________________________________________________________________________
dense_3 (Dense)                 (None, 10)           210         lstm_2[0][0]
================================================================================================
Total params: 1,970
Trainable params: 1,970
Non-trainable params: 0
________________________________________________________________________________________________
```

In [54]:

```
re_X_aug_train=X_aug_train.reshape(14400,17640,1)
re_X_aug_test=X_aug_test.reshape(3600,17640,1)
```

In [55]:

```
from datetime import datetime
model.compile(loss='SparseCategoricalCrossentropy',optimizer='adam',metrics=['accuracy',f1_scores])
logdir = "logs/fit/" + datetime.now().strftime("%Y%m%d-%H%M%S")
```

In [56]:

```
tensorboard_callback = tf.keras.callbacks.TensorBoard(log_dir=logdir)
model.fit([re_X_aug_train,X_aug_train_mask],y_aug_train,validation_data=([re_X_aug_test,X_aug_test_mask],y_aug_test),batch_size=10
0,epochs=7,callbacks=[tensorboard_callback])
```

```
Epoch 1/7
144/144 [==============================] - 120s 817ms/step - loss: 2.3027 - accuracy: 0.0965 - f1_scores: 0.0965 - val_loss: 2.302
6 - val_accuracy: 0.1028 - val_f1_scores: 0.1028
Epoch 2/7
144/144 [==============================] - 115s 802ms/step - loss: 2.3026 - accuracy: 0.1034 - f1_scores: 0.1034 - val_loss: 2.302
6 - val_accuracy: 0.1075 - val_f1_scores: 0.1075
Epoch 3/7
144/144 [==============================] - 115s 799ms/step - loss: 2.3027 - accuracy: 0.1009 - f1_scores: 0.1009 - val_loss: 2.302
6 - val_accuracy: 0.1053 - val_f1_scores: 0.1053
Epoch 4/7
144/144 [==============================] - 115s 798ms/step - loss: 2.3026 - accuracy: 0.1010 - f1_scores: 0.1010 - val_loss: 2.302
6 - val_accuracy: 0.0992 - val_f1_scores: 0.0992
Epoch 5/7
144/144 [==============================] - 114s 795ms/step - loss: 2.3027 - accuracy: 0.0991 - f1_scores: 0.0991 - val_loss: 2.302
6 - val_accuracy: 0.1011 - val_f1_scores: 0.1011
Epoch 6/7
144/144 [==============================] - 114s 792ms/step - loss: 2.3027 - accuracy: 0.0991 - f1_scores: 0.0991 - val_loss: 2.302
6 - val_accuracy: 0.1164 - val_f1_scores: 0.1164
Epoch 7/7
144/144 [==============================] - 115s 800ms/step - loss: 2.3027 - accuracy: 0.1030 - f1_scores: 0.1030 - val_loss: 2.302
6 - val_accuracy: 0.0956 - val_f1_scores: 0.0956
```

Out[56]:

```
<tensorflow.python.keras.callbacks.History at 0x7f34aa14c7b8>
```

In [ ]:

```
%tensorboard --logdir=./
```

In [59]:

```
X_aug_train_spectrogram =np.array([convert_to_spectrogram(i) for i in (X_aug_train)])
X_aug_test_spectrogram = np.array([convert_to_spectrogram(i) for i in (X_aug_test)])
```

In [60]:

```
input= Input(shape=(64, 35,),dtype=np.float32)
lstm_output=LSTM(200,return_sequences=True)(input)
Dense1=Dense(50,activation='relu')(tf.math.reduce_mean(lstm_output, 2))
Dense2=Dense(10,activation='softmax')(Dense1)
model= Model(inputs=input,outputs=Dense2)
model.summary()
```

```
Model: "model_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_6 (InputLayer)         [(None, 64, 35)]          0
_________________________________________________________________
lstm_3 (LSTM)                (None, 64, 200)           188800
_________________________________________________________________
tf.math.reduce_mean_1 (TFOpL (None, 64)                0
_________________________________________________________________
dense_4 (Dense)              (None, 50)                3250
_________________________________________________________________
dense_5 (Dense)              (None, 10)                510
=================================================================
Total params: 192,560
Trainable params: 192,560
Non-trainable params: 0
_________________________________________________________________
```

In [61]:

```
import keras
```

In [62]:

```
from datetime import datetime
model.compile(loss='sparse_categorical_crossentropy',optimizer='adam',metrics=['accuracy',f1_scores])
logdir = "logs/fit/" + datetime.now().strftime("%Y%m%d-%H%M%S")
```

In [63]:

```
tensorboard_callback = tf.keras.callbacks.TensorBoard(log_dir=logdir)
model.fit(X_train_spectrogram,y_train,validation_data=(X_test_spectrogram,y_test),batch_size=10,epochs=200,callbacks=[tensorboard_
callback])
```

```
Epoch 1/200
140/140 [==============================] - 3s 14ms/step - loss: 2.2733 - accuracy: 0.1552 - f1_scores: 0.1552 - val_loss: 2.1258 -
val_accuracy: 0.1917 - val_f1_scores: 0.1917
Epoch 2/200
140/140 [==============================] - 1s 9ms/step - loss: 2.0236 - accuracy: 0.2701 - f1_scores: 0.2701 - val_loss: 1.7782 -
val_accuracy: 0.4383 - val_f1_scores: 0.4383
Epoch 3/200
140/140 [==============================] - 1s 9ms/step - loss: 1.7204 - accuracy: 0.4410 - f1_scores: 0.4410 - val_loss: 1.5572 -
val_accuracy: 0.4867 - val_f1_scores: 0.4867
Epoch 4/200
140/140 [==============================] - 1s 9ms/step - loss: 1.4538 - accuracy: 0.5084 - f1_scores: 0.5084 - val_loss: 1.3211 -
val_accuracy: 0.5800 - val_f1_scores: 0.5800
Epoch 5/200
140/140 [==============================] - 1s 9ms/step - loss: 1.2808 - accuracy: 0.5908 - f1_scores: 0.5908 - val_loss: 1.1478 -
val_accuracy: 0.6683 - val_f1_scores: 0.6683
Epoch 6/200
140/140 [==============================] - 1s 9ms/step - loss: 1.1242 - accuracy: 0.6513 - f1_scores: 0.6513 - val_loss: 1.0780 -
val_accuracy: 0.6600 - val_f1_scores: 0.6600
Epoch 7/200
140/140 [==============================] - 1s 9ms/step - loss: 1.0226 - accuracy: 0.6817 - f1_scores: 0.6817 - val_loss: 1.0283 -
val_accuracy: 0.6433 - val_f1_scores: 0.6433
Epoch 8/200
140/140 [==============================] - 1s 9ms/step - loss: 0.9328 - accuracy: 0.6979 - f1_scores: 0.6979 - val_loss: 0.8667 -
val_accuracy: 0.7350 - val_f1_scores: 0.7350
Epoch 9/200
140/140 [==============================] - 1s 9ms/step - loss: 0.8705 - accuracy: 0.7203 - f1_scores: 0.7203 - val_loss: 0.8416 -
val_accuracy: 0.7183 - val_f1_scores: 0.7183
Epoch 10/200
140/140 [==============================] - 1s 10ms/step - loss: 0.8757 - accuracy: 0.7014 - f1_scores: 0.7014 - val_loss: 0.7727 -
val_accuracy: 0.7717 - val_f1_scores: 0.7717
Epoch 11/200
140/140 [==============================] - 1s 9ms/step - loss: 0.7351 - accuracy: 0.7570 - f1_scores: 0.7570 - val_loss: 0.9272 -
val_accuracy: 0.6783 - val_f1_scores: 0.6783
Epoch 12/200
140/140 [==============================] - 1s 9ms/step - loss: 0.7925 - accuracy: 0.7552 - f1_scores: 0.7552 - val_loss: 0.7339 -
val_accuracy: 0.7900 - val_f1_scores: 0.7900
Epoch 13/200
140/140 [==============================] - 1s 9ms/step - loss: 0.7320 - accuracy: 0.7628 - f1_scores: 0.7628 - val_loss: 0.6673 -
val_accuracy: 0.8000 - val_f1_scores: 0.8000
Epoch 14/200
140/140 [==============================] - 1s 9ms/step - loss: 0.6623 - accuracy: 0.8012 - f1_scores: 0.8012 - val_loss: 0.6265 -
val_accuracy: 0.8000 - val_f1_scores: 0.8000
Epoch 15/200
140/140 [==============================] - 1s 9ms/step - loss: 0.6681 - accuracy: 0.7894 - f1_scores: 0.7894 - val_loss: 0.6324 -
```

```
val_accuracy: 0.7950 - val_f1_scores: 0.7950
Epoch 16/200
140/140 [==============================] - 1s 9ms/step - loss: 0.6179 - accuracy: 0.8092 - f1_scores: 0.8092 - val_loss: 0.6476 -
val_accuracy: 0.7833 - val_f1_scores: 0.7833
Epoch 17/200
140/140 [==============================] - 1s 9ms/step - loss: 0.5957 - accuracy: 0.8172 - f1_scores: 0.8172 - val_loss: 0.6165 -
val_accuracy: 0.7983 - val_f1_scores: 0.7983
Epoch 18/200
140/140 [==============================] - 1s 9ms/step - loss: 0.5675 - accuracy: 0.8208 - f1_scores: 0.8208 - val_loss: 0.5815 -
val_accuracy: 0.8067 - val_f1_scores: 0.8067
Epoch 19/200
140/140 [==============================] - 1s 9ms/step - loss: 0.5637 - accuracy: 0.8225 - f1_scores: 0.8225 - val_loss: 0.5497 -
val_accuracy: 0.8417 - val_f1_scores: 0.8417
Epoch 20/200
140/140 [==============================] - 1s 9ms/step - loss: 0.5290 - accuracy: 0.8415 - f1_scores: 0.8415 - val_loss: 0.5273 -
val_accuracy: 0.8433 - val_f1_scores: 0.8433
Epoch 21/200
140/140 [==============================] - 1s 9ms/step - loss: 0.5576 - accuracy: 0.8359 - f1_scores: 0.8359 - val_loss: 0.5114 -
val_accuracy: 0.8567 - val_f1_scores: 0.8567
Epoch 22/200
140/140 [==============================] - 1s 9ms/step - loss: 0.5203 - accuracy: 0.8532 - f1_scores: 0.8532 - val_loss: 0.5209 -
val_accuracy: 0.8167 - val_f1_scores: 0.8167
Epoch 23/200
140/140 [==============================] - 1s 9ms/step - loss: 0.4857 - accuracy: 0.8488 - f1_scores: 0.8488 - val_loss: 0.4873 -
val_accuracy: 0.8683 - val_f1_scores: 0.8683
Epoch 24/200
140/140 [==============================] - 1s 9ms/step - loss: 0.4997 - accuracy: 0.8529 - f1_scores: 0.8529 - val_loss: 0.5317 -
val_accuracy: 0.8467 - val_f1_scores: 0.8467
Epoch 25/200
140/140 [==============================] - 1s 9ms/step - loss: 0.4854 - accuracy: 0.8604 - f1_scores: 0.8604 - val_loss: 0.4666 -
val_accuracy: 0.8550 - val_f1_scores: 0.8550
Epoch 26/200
140/140 [==============================] - 1s 9ms/step - loss: 0.4600 - accuracy: 0.8697 - f1_scores: 0.8697 - val_loss: 0.4710 -
val_accuracy: 0.8383 - val_f1_scores: 0.8383
Epoch 27/200
140/140 [==============================] - 1s 10ms/step - loss: 0.4411 - accuracy: 0.8611 - f1_scores: 0.8611 - val_loss: 0.4679 -
val_accuracy: 0.8567 - val_f1_scores: 0.8567
Epoch 28/200
140/140 [==============================] - 1s 9ms/step - loss: 0.4321 - accuracy: 0.8761 - f1_scores: 0.8761 - val_loss: 0.4522 -
val_accuracy: 0.8500 - val_f1_scores: 0.8500
Epoch 29/200
140/140 [==============================] - 1s 9ms/step - loss: 0.4394 - accuracy: 0.8727 - f1_scores: 0.8727 - val_loss: 0.4428 -
val_accuracy: 0.8567 - val_f1_scores: 0.8567
Epoch 30/200
140/140 [==============================] - 1s 10ms/step - loss: 0.4175 - accuracy: 0.8936 - f1_scores: 0.8936 - val_loss: 0.4485 -
val_accuracy: 0.8650 - val_f1_scores: 0.8650
Epoch 31/200
140/140 [==============================] - 1s 10ms/step - loss: 0.4180 - accuracy: 0.8842 - f1_scores: 0.8842 - val_loss: 0.4100 -
val_accuracy: 0.8567 - val_f1_scores: 0.8567
Epoch 32/200
140/140 [==============================] - 1s 9ms/step - loss: 0.4076 - accuracy: 0.8796 - f1_scores: 0.8796 - val_loss: 0.4123 -
```

```
val_accuracy: 0.8650 - val_f1_scores: 0.8650
Epoch 33/200
140/140 [==============================] - 1s 9ms/step - loss: 0.4044 - accuracy: 0.8825 - f1_scores: 0.8825 - val_loss: 0.4294 -
val_accuracy: 0.8567 - val_f1_scores: 0.8567
Epoch 34/200
140/140 [==============================] - 1s 9ms/step - loss: 0.3626 - accuracy: 0.8907 - f1_scores: 0.8907 - val_loss: 0.4396 -
val_accuracy: 0.8533 - val_f1_scores: 0.8533
Epoch 35/200
140/140 [==============================] - 1s 9ms/step - loss: 0.4207 - accuracy: 0.8804 - f1_scores: 0.8804 - val_loss: 0.3854 -
val_accuracy: 0.8800 - val_f1_scores: 0.8800
Epoch 36/200
140/140 [==============================] - 1s 10ms/step - loss: 0.3998 - accuracy: 0.8881 - f1_scores: 0.8881 - val_loss: 0.4206 -
val_accuracy: 0.8717 - val_f1_scores: 0.8717
Epoch 37/200
140/140 [==============================] - 1s 9ms/step - loss: 0.3729 - accuracy: 0.8875 - f1_scores: 0.8875 - val_loss: 0.4014 -
val_accuracy: 0.8850 - val_f1_scores: 0.8850
Epoch 38/200
140/140 [==============================] - 1s 9ms/step - loss: 0.3799 - accuracy: 0.8974 - f1_scores: 0.8974 - val_loss: 0.4214 -
val_accuracy: 0.8617 - val_f1_scores: 0.8617
Epoch 39/200
140/140 [==============================] - 1s 10ms/step - loss: 0.3537 - accuracy: 0.8924 - f1_scores: 0.8924 - val_loss: 0.3993 -
val_accuracy: 0.8717 - val_f1_scores: 0.8717
Epoch 40/200
140/140 [==============================] - 1s 9ms/step - loss: 0.3518 - accuracy: 0.8950 - f1_scores: 0.8950 - val_loss: 0.3887 -
val_accuracy: 0.8933 - val_f1_scores: 0.8933
Epoch 41/200
140/140 [==============================] - 1s 9ms/step - loss: 0.3513 - accuracy: 0.9079 - f1_scores: 0.9079 - val_loss: 0.3818 -
val_accuracy: 0.8900 - val_f1_scores: 0.8900
Epoch 42/200
140/140 [==============================] - 1s 9ms/step - loss: 0.3798 - accuracy: 0.8987 - f1_scores: 0.8987 - val_loss: 0.3671 -
val_accuracy: 0.8883 - val_f1_scores: 0.8883
Epoch 43/200
140/140 [==============================] - 1s 9ms/step - loss: 0.3155 - accuracy: 0.9049 - f1_scores: 0.9049 - val_loss: 0.3650 -
val_accuracy: 0.8783 - val_f1_scores: 0.8783
Epoch 44/200
140/140 [==============================] - 1s 9ms/step - loss: 0.3561 - accuracy: 0.8971 - f1_scores: 0.8971 - val_loss: 0.3737 -
val_accuracy: 0.8900 - val_f1_scores: 0.8900
Epoch 45/200
140/140 [==============================] - 1s 9ms/step - loss: 0.3331 - accuracy: 0.9066 - f1_scores: 0.9066 - val_loss: 0.3446 -
val_accuracy: 0.8933 - val_f1_scores: 0.8933
Epoch 46/200
140/140 [==============================] - 1s 9ms/step - loss: 0.3378 - accuracy: 0.9025 - f1_scores: 0.9025 - val_loss: 0.3481 -
val_accuracy: 0.8917 - val_f1_scores: 0.8917
Epoch 47/200
140/140 [==============================] - 1s 11ms/step - loss: 0.3309 - accuracy: 0.8935 - f1_scores: 0.8935 - val_loss: 0.3663 -
val_accuracy: 0.8800 - val_f1_scores: 0.8800
Epoch 48/200
140/140 [==============================] - 1s 9ms/step - loss: 0.3523 - accuracy: 0.8999 - f1_scores: 0.8999 - val_loss: 0.3678 -
val_accuracy: 0.8700 - val_f1_scores: 0.8700
Epoch 49/200
```

```
140/140 [==============================] - 1s 9ms/step - loss: 0.3321 - accuracy: 0.8993 - f1_scores: 0.8993 - val_loss: 0.3723 -
val_accuracy: 0.8767 - val_f1_scores: 0.8767
Epoch 50/200
140/140 [==============================] - 1s 10ms/step - loss: 0.3513 - accuracy: 0.8984 - f1_scores: 0.8984 - val_loss: 0.3534 -
val_accuracy: 0.8950 - val_f1_scores: 0.8950
Epoch 51/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2922 - accuracy: 0.9230 - f1_scores: 0.9230 - val_loss: 0.3618 -
val_accuracy: 0.8783 - val_f1_scores: 0.8783
Epoch 52/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2959 - accuracy: 0.9137 - f1_scores: 0.9137 - val_loss: 0.3658 -
val_accuracy: 0.8867 - val_f1_scores: 0.8867
Epoch 53/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2763 - accuracy: 0.9260 - f1_scores: 0.9260 - val_loss: 0.3491 -
val_accuracy: 0.8750 - val_f1_scores: 0.8750
Epoch 54/200
140/140 [==============================] - 1s 9ms/step - loss: 0.3008 - accuracy: 0.9083 - f1_scores: 0.9083 - val_loss: 0.3671 -
val_accuracy: 0.8717 - val_f1_scores: 0.8717
Epoch 55/200
140/140 [==============================] - 1s 9ms/step - loss: 0.3294 - accuracy: 0.9073 - f1_scores: 0.9073 - val_loss: 0.3441 -
val_accuracy: 0.8917 - val_f1_scores: 0.8917
Epoch 56/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2723 - accuracy: 0.9270 - f1_scores: 0.9270 - val_loss: 0.3323 -
val_accuracy: 0.9000 - val_f1_scores: 0.9000
Epoch 57/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2904 - accuracy: 0.9148 - f1_scores: 0.9148 - val_loss: 0.3214 -
val_accuracy: 0.8867 - val_f1_scores: 0.8867
Epoch 58/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2903 - accuracy: 0.9130 - f1_scores: 0.9130 - val_loss: 0.3501 -
val_accuracy: 0.8933 - val_f1_scores: 0.8933
Epoch 59/200
140/140 [==============================] - 1s 10ms/step - loss: 0.3113 - accuracy: 0.9247 - f1_scores: 0.9247 - val_loss: 0.2943 -
val_accuracy: 0.9067 - val_f1_scores: 0.9067
Epoch 60/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2636 - accuracy: 0.9181 - f1_scores: 0.9181 - val_loss: 0.3114 -
val_accuracy: 0.8950 - val_f1_scores: 0.8950
Epoch 61/200
140/140 [==============================] - 1s 10ms/step - loss: 0.2571 - accuracy: 0.9229 - f1_scores: 0.9229 - val_loss: 0.3029 -
val_accuracy: 0.9000 - val_f1_scores: 0.9000
Epoch 62/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2804 - accuracy: 0.9202 - f1_scores: 0.9202 - val_loss: 0.3253 -
val_accuracy: 0.8833 - val_f1_scores: 0.8833
Epoch 63/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2864 - accuracy: 0.9114 - f1_scores: 0.9114 - val_loss: 0.3216 -
val_accuracy: 0.9067 - val_f1_scores: 0.9067
Epoch 64/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2505 - accuracy: 0.9290 - f1_scores: 0.9290 - val_loss: 0.3389 -
val_accuracy: 0.8983 - val_f1_scores: 0.8983
Epoch 65/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2646 - accuracy: 0.9273 - f1_scores: 0.9273 - val_loss: 0.3526 -
val_accuracy: 0.8783 - val_f1_scores: 0.8783
Epoch 66/200
```

```
140/140 [==============================] - 1s 9ms/step - loss: 0.2849 - accuracy: 0.9127 - f1_scores: 0.9127 - val_loss: 0.3284 -
val_accuracy: 0.9017 - val_f1_scores: 0.9017
Epoch 67/200
140/140 [==============================] - 1s 10ms/step - loss: 0.2784 - accuracy: 0.9141 - f1_scores: 0.9141 - val_loss: 0.3269 -
val_accuracy: 0.8817 - val_f1_scores: 0.8817
Epoch 68/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2609 - accuracy: 0.9292 - f1_scores: 0.9292 - val_loss: 0.3031 -
val_accuracy: 0.9067 - val_f1_scores: 0.9067
Epoch 69/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2474 - accuracy: 0.9377 - f1_scores: 0.9377 - val_loss: 0.2987 -
val_accuracy: 0.9017 - val_f1_scores: 0.9017
Epoch 70/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2409 - accuracy: 0.9278 - f1_scores: 0.9278 - val_loss: 0.3041 -
val_accuracy: 0.8967 - val_f1_scores: 0.8967
Epoch 71/200
140/140 [==============================] - 1s 10ms/step - loss: 0.2709 - accuracy: 0.9202 - f1_scores: 0.9202 - val_loss: 0.2980 -
val_accuracy: 0.9050 - val_f1_scores: 0.9050
Epoch 72/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2425 - accuracy: 0.9304 - f1_scores: 0.9304 - val_loss: 0.3100 -
val_accuracy: 0.9017 - val_f1_scores: 0.9017
Epoch 73/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2546 - accuracy: 0.9468 - f1_scores: 0.9468 - val_loss: 0.2966 -
val_accuracy: 0.9017 - val_f1_scores: 0.9017
Epoch 74/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2157 - accuracy: 0.9346 - f1_scores: 0.9346 - val_loss: 0.2945 -
val_accuracy: 0.9050 - val_f1_scores: 0.9050
Epoch 75/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2365 - accuracy: 0.9361 - f1_scores: 0.9361 - val_loss: 0.3186 -
val_accuracy: 0.9017 - val_f1_scores: 0.9017
Epoch 76/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2562 - accuracy: 0.9269 - f1_scores: 0.9269 - val_loss: 0.3367 -
val_accuracy: 0.8833 - val_f1_scores: 0.8833
Epoch 77/200
140/140 [==============================] - 1s 10ms/step - loss: 0.2451 - accuracy: 0.9354 - f1_scores: 0.9354 - val_loss: 0.2899 -
val_accuracy: 0.9067 - val_f1_scores: 0.9067
Epoch 78/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2280 - accuracy: 0.9352 - f1_scores: 0.9352 - val_loss: 0.2750 -
val_accuracy: 0.9167 - val_f1_scores: 0.9167
Epoch 79/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2367 - accuracy: 0.9320 - f1_scores: 0.9320 - val_loss: 0.2825 -
val_accuracy: 0.9083 - val_f1_scores: 0.9083
Epoch 80/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2168 - accuracy: 0.9405 - f1_scores: 0.9405 - val_loss: 0.2975 -
val_accuracy: 0.9067 - val_f1_scores: 0.9067
Epoch 81/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2127 - accuracy: 0.9439 - f1_scores: 0.9439 - val_loss: 0.2920 -
val_accuracy: 0.8983 - val_f1_scores: 0.8983
Epoch 82/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2467 - accuracy: 0.9307 - f1_scores: 0.9307 - val_loss: 0.2873 -
val_accuracy: 0.9067 - val_f1_scores: 0.9067
Epoch 83/200
```

```
Epoch 83/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2127 - accuracy: 0.9448 - f1_scores: 0.9448 - val_loss: 0.2969 -
val_accuracy: 0.9167 - val_f1_scores: 0.9167
Epoch 84/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2115 - accuracy: 0.9470 - f1_scores: 0.9470 - val_loss: 0.3197 -
val_accuracy: 0.8883 - val_f1_scores: 0.8883
Epoch 85/200
140/140 [==============================] - 1s 11ms/step - loss: 0.2213 - accuracy: 0.9379 - f1_scores: 0.9379 - val_loss: 0.2854 -
val_accuracy: 0.9133 - val_f1_scores: 0.9133
Epoch 86/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2356 - accuracy: 0.9353 - f1_scores: 0.9353 - val_loss: 0.2735 -
val_accuracy: 0.9117 - val_f1_scores: 0.9117
Epoch 87/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2307 - accuracy: 0.9287 - f1_scores: 0.9287 - val_loss: 0.2905 -
val_accuracy: 0.9000 - val_f1_scores: 0.9000
Epoch 88/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2300 - accuracy: 0.9380 - f1_scores: 0.9380 - val_loss: 0.3071 -
val_accuracy: 0.9000 - val_f1_scores: 0.9000
Epoch 89/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2235 - accuracy: 0.9407 - f1_scores: 0.9407 - val_loss: 0.2983 -
val_accuracy: 0.9083 - val_f1_scores: 0.9083
Epoch 90/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2333 - accuracy: 0.9454 - f1_scores: 0.9454 - val_loss: 0.2737 -
val_accuracy: 0.9200 - val_f1_scores: 0.9200
Epoch 91/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2561 - accuracy: 0.9319 - f1_scores: 0.9319 - val_loss: 0.2981 -
val_accuracy: 0.9017 - val_f1_scores: 0.9017
Epoch 92/200
140/140 [==============================] - 1s 10ms/step - loss: 0.2449 - accuracy: 0.9296 - f1_scores: 0.9296 - val_loss: 0.2726 -
val_accuracy: 0.9133 - val_f1_scores: 0.9133
Epoch 93/200
140/140 [==============================] - 1s 10ms/step - loss: 0.2350 - accuracy: 0.9349 - f1_scores: 0.9349 - val_loss: 0.2987 -
val_accuracy: 0.9117 - val_f1_scores: 0.9117
Epoch 94/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2195 - accuracy: 0.9442 - f1_scores: 0.9442 - val_loss: 0.3092 -
val_accuracy: 0.9033 - val_f1_scores: 0.9033
Epoch 95/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2224 - accuracy: 0.9303 - f1_scores: 0.9303 - val_loss: 0.2914 -
val_accuracy: 0.8983 - val_f1_scores: 0.8983
Epoch 96/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2118 - accuracy: 0.9338 - f1_scores: 0.9339 - val_loss: 0.2627 -
val_accuracy: 0.9200 - val_f1_scores: 0.9200
Epoch 97/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1727 - accuracy: 0.9562 - f1_scores: 0.9562 - val_loss: 0.2656 -
val_accuracy: 0.9117 - val_f1_scores: 0.9117
Epoch 98/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1819 - accuracy: 0.9457 - f1_scores: 0.9457 - val_loss: 0.2754 -
val_accuracy: 0.9167 - val_f1_scores: 0.9167
Epoch 99/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2012 - accuracy: 0.9441 - f1_scores: 0.9441 - val_loss: 0.2776 -
val_accuracy: 0.9067 - val_f1_scores: 0.9067
```

```
Epoch 100/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2043 - accuracy: 0.9457 - f1_scores: 0.9457 - val_loss: 0.3108 -
val_accuracy: 0.8983 - val_f1_scores: 0.8983
Epoch 101/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2149 - accuracy: 0.9359 - f1_scores: 0.9359 - val_loss: 0.2639 -
val_accuracy: 0.9117 - val_f1_scores: 0.9117
Epoch 102/200
140/140 [==============================] - 1s 10ms/step - loss: 0.2003 - accuracy: 0.9423 - f1_scores: 0.9423 - val_loss: 0.2569 -
val_accuracy: 0.9117 - val_f1_scores: 0.9117
Epoch 103/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2102 - accuracy: 0.9353 - f1_scores: 0.9353 - val_loss: 0.2819 -
val_accuracy: 0.9083 - val_f1_scores: 0.9083
Epoch 104/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2238 - accuracy: 0.9365 - f1_scores: 0.9365 - val_loss: 0.2806 -
val_accuracy: 0.9167 - val_f1_scores: 0.9167
Epoch 105/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1986 - accuracy: 0.9388 - f1_scores: 0.9388 - val_loss: 0.2798 -
val_accuracy: 0.9083 - val_f1_scores: 0.9083
Epoch 106/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1945 - accuracy: 0.9460 - f1_scores: 0.9460 - val_loss: 0.2676 -
val_accuracy: 0.9183 - val_f1_scores: 0.9183
Epoch 107/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1886 - accuracy: 0.9437 - f1_scores: 0.9437 - val_loss: 0.2856 -
val_accuracy: 0.9017 - val_f1_scores: 0.9017
Epoch 108/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1901 - accuracy: 0.9445 - f1_scores: 0.9445 - val_loss: 0.2893 -
val_accuracy: 0.9100 - val_f1_scores: 0.9100
Epoch 109/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1887 - accuracy: 0.9532 - f1_scores: 0.9532 - val_loss: 0.2974 -
val_accuracy: 0.9117 - val_f1_scores: 0.9117
Epoch 110/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1951 - accuracy: 0.9430 - f1_scores: 0.9430 - val_loss: 0.2801 -
val_accuracy: 0.9050 - val_f1_scores: 0.9050
Epoch 111/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2192 - accuracy: 0.9444 - f1_scores: 0.9444 - val_loss: 0.2704 -
val_accuracy: 0.9083 - val_f1_scores: 0.9083
Epoch 112/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1819 - accuracy: 0.9504 - f1_scores: 0.9504 - val_loss: 0.2586 -
val_accuracy: 0.9100 - val_f1_scores: 0.9100
Epoch 113/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2042 - accuracy: 0.9449 - f1_scores: 0.9449 - val_loss: 0.2862 -
val_accuracy: 0.9183 - val_f1_scores: 0.9183
Epoch 114/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1712 - accuracy: 0.9519 - f1_scores: 0.9519 - val_loss: 0.2896 -
val_accuracy: 0.8983 - val_f1_scores: 0.8983
Epoch 115/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2018 - accuracy: 0.9453 - f1_scores: 0.9453 - val_loss: 0.3191 -
val_accuracy: 0.8933 - val_f1_scores: 0.8933
Epoch 116/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1783 - accuracy: 0.9537 - f1_scores: 0.9537 - val_loss: 0.2684 -
val_accuracy: 0.9083 - val_f1_scores: 0.9083
```

```
val_accuracy: 0.9083 - val_f1_scores: 0.9083
Epoch 117/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2239 - accuracy: 0.9318 - f1_scores: 0.9318 - val_loss: 0.2751 -
val_accuracy: 0.9167 - val_f1_scores: 0.9167
Epoch 118/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1770 - accuracy: 0.9471 - f1_scores: 0.9471 - val_loss: 0.2563 -
val_accuracy: 0.9233 - val_f1_scores: 0.9233
Epoch 119/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1876 - accuracy: 0.9465 - f1_scores: 0.9465 - val_loss: 0.2728 -
val_accuracy: 0.9050 - val_f1_scores: 0.9050
Epoch 120/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1858 - accuracy: 0.9483 - f1_scores: 0.9483 - val_loss: 0.2717 -
val_accuracy: 0.8983 - val_f1_scores: 0.8983
Epoch 121/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1751 - accuracy: 0.9455 - f1_scores: 0.9455 - val_loss: 0.2794 -
val_accuracy: 0.9167 - val_f1_scores: 0.9167
Epoch 122/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1579 - accuracy: 0.9651 - f1_scores: 0.9651 - val_loss: 0.3000 -
val_accuracy: 0.8917 - val_f1_scores: 0.8917
Epoch 123/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1618 - accuracy: 0.9582 - f1_scores: 0.9582 - val_loss: 0.3041 -
val_accuracy: 0.8917 - val_f1_scores: 0.8917
Epoch 124/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1805 - accuracy: 0.9495 - f1_scores: 0.9495 - val_loss: 0.2765 -
val_accuracy: 0.9017 - val_f1_scores: 0.9017
Epoch 125/200
140/140 [==============================] - 1s 10ms/step - loss: 0.2009 - accuracy: 0.9420 - f1_scores: 0.9420 - val_loss: 0.2777 -
val_accuracy: 0.9133 - val_f1_scores: 0.9133
Epoch 126/200
140/140 [==============================] - 1s 11ms/step - loss: 0.1767 - accuracy: 0.9560 - f1_scores: 0.9560 - val_loss: 0.3132 -
val_accuracy: 0.8967 - val_f1_scores: 0.8967
Epoch 127/200
140/140 [==============================] - 1s 10ms/step - loss: 0.2106 - accuracy: 0.9501 - f1_scores: 0.9501 - val_loss: 0.2803 -
val_accuracy: 0.9183 - val_f1_scores: 0.9183
Epoch 128/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1996 - accuracy: 0.9424 - f1_scores: 0.9424 - val_loss: 0.2791 -
val_accuracy: 0.9133 - val_f1_scores: 0.9133
Epoch 129/200
140/140 [==============================] - 1s 11ms/step - loss: 0.1772 - accuracy: 0.9481 - f1_scores: 0.9481 - val_loss: 0.2869 -
val_accuracy: 0.9050 - val_f1_scores: 0.9050
Epoch 130/200
140/140 [==============================] - 1s 10ms/step - loss: 0.2069 - accuracy: 0.9430 - f1_scores: 0.9430 - val_loss: 0.2908 -
val_accuracy: 0.8983 - val_f1_scores: 0.8983
Epoch 131/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1938 - accuracy: 0.9450 - f1_scores: 0.9450 - val_loss: 0.3142 -
val_accuracy: 0.8833 - val_f1_scores: 0.8833
Epoch 132/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1997 - accuracy: 0.9356 - f1_scores: 0.9356 - val_loss: 0.2760 -
val_accuracy: 0.9167 - val_f1_scores: 0.9167
Epoch 133/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1820 - accuracy: 0.9504 - f1_scores: 0.9504 - val_loss: 0.2859 -
```

```
val_accuracy: 0.9100 - val_f1_scores: 0.9100
Epoch 134/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1970 - accuracy: 0.9414 - f1_scores: 0.9414 - val_loss: 0.2553 -
val_accuracy: 0.9133 - val_f1_scores: 0.9133
Epoch 135/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1782 - accuracy: 0.9567 - f1_scores: 0.9567 - val_loss: 0.2556 -
val_accuracy: 0.9150 - val_f1_scores: 0.9150
Epoch 136/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1701 - accuracy: 0.9638 - f1_scores: 0.9638 - val_loss: 0.2839 -
val_accuracy: 0.9067 - val_f1_scores: 0.9067
Epoch 137/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1817 - accuracy: 0.9533 - f1_scores: 0.9533 - val_loss: 0.2703 -
val_accuracy: 0.9067 - val_f1_scores: 0.9067
Epoch 138/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1570 - accuracy: 0.9629 - f1_scores: 0.9629 - val_loss: 0.3166 -
val_accuracy: 0.8950 - val_f1_scores: 0.8950
Epoch 139/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1893 - accuracy: 0.9550 - f1_scores: 0.9550 - val_loss: 0.2970 -
val_accuracy: 0.8967 - val_f1_scores: 0.8967
Epoch 140/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1866 - accuracy: 0.9444 - f1_scores: 0.9444 - val_loss: 0.2658 -
val_accuracy: 0.9150 - val_f1_scores: 0.9150
Epoch 141/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1717 - accuracy: 0.9475 - f1_scores: 0.9475 - val_loss: 0.2986 -
val_accuracy: 0.9000 - val_f1_scores: 0.9000
Epoch 142/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1748 - accuracy: 0.9515 - f1_scores: 0.9515 - val_loss: 0.2783 -
val_accuracy: 0.9117 - val_f1_scores: 0.9117
Epoch 143/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1813 - accuracy: 0.9460 - f1_scores: 0.9460 - val_loss: 0.2540 -
val_accuracy: 0.9117 - val_f1_scores: 0.9117
Epoch 144/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1673 - accuracy: 0.9501 - f1_scores: 0.9501 - val_loss: 0.2642 -
val_accuracy: 0.9183 - val_f1_scores: 0.9183
Epoch 145/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1571 - accuracy: 0.9558 - f1_scores: 0.9558 - val_loss: 0.2674 -
val_accuracy: 0.9083 - val_f1_scores: 0.9083
Epoch 146/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1895 - accuracy: 0.9450 - f1_scores: 0.9450 - val_loss: 0.2672 -
val_accuracy: 0.9133 - val_f1_scores: 0.9133
Epoch 147/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1749 - accuracy: 0.9530 - f1_scores: 0.9530 - val_loss: 0.2627 -
val_accuracy: 0.9133 - val_f1_scores: 0.9133
Epoch 148/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1614 - accuracy: 0.9462 - f1_scores: 0.9462 - val_loss: 0.2499 -
val_accuracy: 0.9233 - val_f1_scores: 0.9233
Epoch 149/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1733 - accuracy: 0.9533 - f1_scores: 0.9533 - val_loss: 0.2766 -
val_accuracy: 0.9133 - val_f1_scores: 0.9133
Epoch 150/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1748 - accuracy: 0.9483 - f1_scores: 0.9483 - val_loss: 0.2520 -
```

```
140/140 [==============================] - 1s 9ms/step - loss: 0.1748 - accuracy: 0.9465 - f1_scores: 0.9465 - val_loss: 0.2526 -
val_accuracy: 0.9250 - val_f1_scores: 0.9250
Epoch 151/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1750 - accuracy: 0.9551 - f1_scores: 0.9551 - val_loss: 0.2346 -
val_accuracy: 0.9267 - val_f1_scores: 0.9267
Epoch 152/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1319 - accuracy: 0.9657 - f1_scores: 0.9657 - val_loss: 0.2898 -
val_accuracy: 0.8983 - val_f1_scores: 0.8983
Epoch 153/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1497 - accuracy: 0.9631 - f1_scores: 0.9631 - val_loss: 0.2419 -
val_accuracy: 0.9233 - val_f1_scores: 0.9233
Epoch 154/200
140/140 [==============================] - 1s 9ms/step - loss: 0.2033 - accuracy: 0.9359 - f1_scores: 0.9359 - val_loss: 0.3166 -
val_accuracy: 0.8750 - val_f1_scores: 0.8750
Epoch 155/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1639 - accuracy: 0.9519 - f1_scores: 0.9519 - val_loss: 0.2831 -
val_accuracy: 0.9017 - val_f1_scores: 0.9017
Epoch 156/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1719 - accuracy: 0.9460 - f1_scores: 0.9460 - val_loss: 0.2684 -
val_accuracy: 0.9100 - val_f1_scores: 0.9100
Epoch 157/200
140/140 [==============================] - 2s 11ms/step - loss: 0.1669 - accuracy: 0.9548 - f1_scores: 0.9548 - val_loss: 0.2611 -
val_accuracy: 0.9000 - val_f1_scores: 0.9000
Epoch 158/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1514 - accuracy: 0.9621 - f1_scores: 0.9621 - val_loss: 0.3026 -
val_accuracy: 0.8933 - val_f1_scores: 0.8933
Epoch 159/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1574 - accuracy: 0.9581 - f1_scores: 0.9581 - val_loss: 0.2574 -
val_accuracy: 0.9083 - val_f1_scores: 0.9083
Epoch 160/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1556 - accuracy: 0.9605 - f1_scores: 0.9605 - val_loss: 0.2505 -
val_accuracy: 0.9267 - val_f1_scores: 0.9267
Epoch 161/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1671 - accuracy: 0.9511 - f1_scores: 0.9511 - val_loss: 0.2443 -
val_accuracy: 0.9267 - val_f1_scores: 0.9267
Epoch 162/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1594 - accuracy: 0.9511 - f1_scores: 0.9511 - val_loss: 0.2695 -
val_accuracy: 0.9183 - val_f1_scores: 0.9183
Epoch 163/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1451 - accuracy: 0.9600 - f1_scores: 0.9600 - val_loss: 0.2481 -
val_accuracy: 0.9233 - val_f1_scores: 0.9233
Epoch 164/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1326 - accuracy: 0.9745 - f1_scores: 0.9745 - val_loss: 0.2737 -
val_accuracy: 0.9100 - val_f1_scores: 0.9100
Epoch 165/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1344 - accuracy: 0.9658 - f1_scores: 0.9658 - val_loss: 0.3126 -
val_accuracy: 0.9017 - val_f1_scores: 0.9017
Epoch 166/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1520 - accuracy: 0.9621 - f1_scores: 0.9621 - val_loss: 0.2674 -
val_accuracy: 0.9033 - val_f1_scores: 0.9033
Epoch 167/200
```

```
140/140 [==============================] - 1s 9ms/step - loss: 0.1557 - accuracy: 0.9662 - f1_scores: 0.9662 - val_loss: 0.2601 -
val_accuracy: 0.9183 - val_f1_scores: 0.9183
Epoch 168/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1649 - accuracy: 0.9478 - f1_scores: 0.9478 - val_loss: 0.2681 -
val_accuracy: 0.9067 - val_f1_scores: 0.9067
Epoch 169/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1385 - accuracy: 0.9648 - f1_scores: 0.9648 - val_loss: 0.2397 -
val_accuracy: 0.9250 - val_f1_scores: 0.9250
Epoch 170/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1357 - accuracy: 0.9678 - f1_scores: 0.9678 - val_loss: 0.2668 -
val_accuracy: 0.9150 - val_f1_scores: 0.9150
Epoch 171/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1414 - accuracy: 0.9600 - f1_scores: 0.9600 - val_loss: 0.2343 -
val_accuracy: 0.9183 - val_f1_scores: 0.9183
Epoch 172/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1335 - accuracy: 0.9640 - f1_scores: 0.9640 - val_loss: 0.2567 -
val_accuracy: 0.9183 - val_f1_scores: 0.9183
Epoch 173/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1630 - accuracy: 0.9493 - f1_scores: 0.9493 - val_loss: 0.2536 -
val_accuracy: 0.9267 - val_f1_scores: 0.9267
Epoch 174/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1568 - accuracy: 0.9635 - f1_scores: 0.9635 - val_loss: 0.2555 -
val_accuracy: 0.9067 - val_f1_scores: 0.9067
Epoch 175/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1774 - accuracy: 0.9467 - f1_scores: 0.9467 - val_loss: 0.2369 -
val_accuracy: 0.9250 - val_f1_scores: 0.9250
Epoch 176/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1467 - accuracy: 0.9526 - f1_scores: 0.9526 - val_loss: 0.2569 -
val_accuracy: 0.9167 - val_f1_scores: 0.9167
Epoch 177/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1445 - accuracy: 0.9651 - f1_scores: 0.9651 - val_loss: 0.2670 -
val_accuracy: 0.9050 - val_f1_scores: 0.9050
Epoch 178/200
140/140 [==============================] - 1s 11ms/step - loss: 0.1430 - accuracy: 0.9573 - f1_scores: 0.9573 - val_loss: 0.2553 -
val_accuracy: 0.9100 - val_f1_scores: 0.9100
Epoch 179/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1582 - accuracy: 0.9555 - f1_scores: 0.9555 - val_loss: 0.2300 -
val_accuracy: 0.9267 - val_f1_scores: 0.9267
Epoch 180/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1491 - accuracy: 0.9594 - f1_scores: 0.9594 - val_loss: 0.2954 -
val_accuracy: 0.8950 - val_f1_scores: 0.8950
Epoch 181/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1664 - accuracy: 0.9527 - f1_scores: 0.9527 - val_loss: 0.2345 -
val_accuracy: 0.9217 - val_f1_scores: 0.9217
Epoch 182/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1245 - accuracy: 0.9714 - f1_scores: 0.9714 - val_loss: 0.2583 -
val_accuracy: 0.9133 - val_f1_scores: 0.9133
Epoch 183/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1467 - accuracy: 0.9621 - f1_scores: 0.9621 - val_loss: 0.2634 -
val_accuracy: 0.9083 - val_f1_scores: 0.9083
Epoch 184/200
```

```
Epoch 184/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1437 - accuracy: 0.9592 - f1_scores: 0.9592 - val_loss: 0.2501 -
val_accuracy: 0.9150 - val_f1_scores: 0.9150
Epoch 185/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1365 - accuracy: 0.9688 - f1_scores: 0.9688 - val_loss: 0.2531 -
val_accuracy: 0.9250 - val_f1_scores: 0.9250
Epoch 186/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1293 - accuracy: 0.9629 - f1_scores: 0.9629 - val_loss: 0.2532 -
val_accuracy: 0.9050 - val_f1_scores: 0.9050
Epoch 187/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1177 - accuracy: 0.9713 - f1_scores: 0.9713 - val_loss: 0.2886 -
val_accuracy: 0.9100 - val_f1_scores: 0.9100
Epoch 188/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1361 - accuracy: 0.9609 - f1_scores: 0.9609 - val_loss: 0.2423 -
val_accuracy: 0.9200 - val_f1_scores: 0.9200
Epoch 189/200
140/140 [==============================] - 1s 11ms/step - loss: 0.1433 - accuracy: 0.9573 - f1_scores: 0.9573 - val_loss: 0.2588 -
val_accuracy: 0.9150 - val_f1_scores: 0.9150
Epoch 190/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1314 - accuracy: 0.9578 - f1_scores: 0.9578 - val_loss: 0.2787 -
val_accuracy: 0.9033 - val_f1_scores: 0.9033
Epoch 191/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1389 - accuracy: 0.9677 - f1_scores: 0.9677 - val_loss: 0.2426 -
val_accuracy: 0.9283 - val_f1_scores: 0.9283
Epoch 192/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1361 - accuracy: 0.9619 - f1_scores: 0.9619 - val_loss: 0.2840 -
val_accuracy: 0.8950 - val_f1_scores: 0.8950
Epoch 193/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1522 - accuracy: 0.9604 - f1_scores: 0.9604 - val_loss: 0.2536 -
val_accuracy: 0.9083 - val_f1_scores: 0.9083
Epoch 194/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1465 - accuracy: 0.9572 - f1_scores: 0.9572 - val_loss: 0.2654 -
val_accuracy: 0.9100 - val_f1_scores: 0.9100
Epoch 195/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1582 - accuracy: 0.9504 - f1_scores: 0.9504 - val_loss: 0.2556 -
val_accuracy: 0.9100 - val_f1_scores: 0.9100
Epoch 196/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1131 - accuracy: 0.9767 - f1_scores: 0.9767 - val_loss: 0.2782 -
val_accuracy: 0.9083 - val_f1_scores: 0.9083
Epoch 197/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1418 - accuracy: 0.9594 - f1_scores: 0.9594 - val_loss: 0.2911 -
val_accuracy: 0.9050 - val_f1_scores: 0.9050
Epoch 198/200
140/140 [==============================] - 1s 10ms/step - loss: 0.1398 - accuracy: 0.9593 - f1_scores: 0.9593 - val_loss: 0.2404 -
val_accuracy: 0.9183 - val_f1_scores: 0.9183
Epoch 199/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1282 - accuracy: 0.9597 - f1_scores: 0.9597 - val_loss: 0.2797 -
val_accuracy: 0.9100 - val_f1_scores: 0.9100
Epoch 200/200
140/140 [==============================] - 1s 9ms/step - loss: 0.1432 - accuracy: 0.9592 - f1_scores: 0.9592 - val_loss: 0.2556 -
val_accuracy: 0.9083 - val_f1_scores: 0.9083
```

Out[63]:

```
<tensorflow.python.keras.callbacks.History at 0x7f331e6a3f98>
```

In [ ]:

```
%tensorboard --logdir=./
```

In [ ]:

```

```