In [1]:

```
!wget --header="Host: storage.googleapis.com" --header="User-Agent: Mozilla/5.0 (Windo
```

```
--2021-04-30 15:08:57--  https://storage.googleapis.com/kagglesdsdata/competitions/7634/46676/train.7z?GoogleAccessId=web-data@kaggle-161607.iam.gserviceaccount.com&Expires=1620017510&Signature=XIuI9aJLJcDgoFuUsl31Iw762mbcQzn1A8sdD3aKFt7le7lP%2BRNAlcMQwsKf2aZPu49EIUxkHjqyT8m2Cdkv74rixFFBXXUefuA7Oe1EcTl7BCUEC5mj3wbw8Cc22c6SaJRLqjeQMLn%2B3H0Gy74h7IvtZLKDsRvIJxHS7jlA8ukFLr3%2FjuoftTmRRCyF7b6AO3mUWMNSh%2Fn3I83CyvKSg1RJg8O1JLvCS8mZ96DeMOqPQRbcvWVyHHJZxFsLI90yhOZIQdOO9Oizi6%2BnO%2B%2BiL%2BIHa7QjjkvxhlWiWU1MeBw2De%2F1VjTdNZnw%2BCwQS6jrglspcunmjZN65gwpgWgvuQ%3D%3D&response-content-disposition=attachment%3B+filename%3Dtrain.7z (https://storage.googleapis.com/kagglesdsdata/competitions/7634/46676/train.7z?GoogleAccessId=web-data@kaggle-161607.iam.gserviceaccount.com&Expires=1620017510&Signature=XIuI9aJLJcDgoFuUsl31Iw762mbcQzn1A8sdD3aKFt7le7lP%2BRNAlcMQwsKf2aZPu49EIUxkHjqyT8m2Cdkv74rixFFBXXUefuA7Oe1EcTl7BCUEC5mj3wbw8Cc22c6SaJRLqjeQMLn%2B3H0Gy74h7IvtZLKDsRvIJxHS7jlA8ukFLr3%2FjuoftTmRRCyF7b6AO3mUWMNSh%2Fn3I83CyvKSg1RJg8O1JLvCS8mZ96DeMOqPQRbcvWVyHHJZxFsLI90yhOZIQdOO9Oizi6%2BnO%2B%2BiL%2BIHa7QjjkvxhlWiWU1MeBw2De%2F1VjTdNZnw%2BCwQS6jrglspcunmjZN65gwpgWgvuQ%3D%3D&response-content-disposition=attachment%3B+filename%3Dtrain.7z)
Resolving storage.googleapis.com (storage.googleapis.com)... 74.125.20.128, 74.125.199.128, 74.125.142.128, ...
Connecting to storage.googleapis.com (storage.googleapis.com)|74.125.20.128|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1121103842 (1.0G) [application/x-7z-compressed]
Saving to: ‘train.7z’

train.7z            100%[===================>]   1.04G  64.2MB/s    in 19s

2021-04-30 15:09:16 (56.7 MB/s) - ‘train.7z’ saved [1121103842/1121103842]
```

In [2]:

```
!apt-get install p7zip-full
!p7zip -d train.7z
```

```
Reading package lists... Done
Building dependency tree
Reading state information... Done
p7zip-full is already the newest version (16.02+dfsg-6).
The following package was automatically installed and is no longer require
d:
  libnvidia-common-460
Use 'apt autoremove' to remove it.
0 upgraded, 0 newly installed, 0 to remove and 34 not upgraded.

7-Zip (a) [64] 16.02 : Copyright (c) 1999-2016 Igor Pavlov : 2016-05-21
p7zip Version 16.02 (locale=en_US.UTF-8,Utf16=on,HugeFiles=on,64 bits,2 CP
Us Intel(R) Xeon(R) CPU @ 2.20GHz (406F0),ASM,AES-NI)

Scanning the drive for archives:
  0M Sca        1 file, 1121103842 bytes (1070 MiB)

Extracting archive: train.7z
--
```

In [3]:

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

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

In [4]:

```
import os
import pandas as pd
import numpy as np
import csv
#!pip install tqdm
from tqdm import tqdm_notebook as tqdm
from collections import Counter
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
from collections import Counter
import librosa
import IPython.display as ipd
import librosa.display
from scipy.io import wavfile
from scipy.fftpack import fft
from scipy import signal

from sklearn.model_selection import train_test_split
from tensorflow.keras.callbacks import ReduceLROnPlateau, TensorBoard
from sklearn.utils import shuffle
from tensorflow.keras.layers import Input, LSTM, Dense , AveragePooling2D ,TimeDistrib
from tensorflow.keras.models import Model
import tensorflow as tf

from sklearn.preprocessing import LabelEncoder

from sklearn.preprocessing import OneHotEncoder
import pickle

from sklearn.metrics import classification_report

os.chdir('/content/train/audio')
```

In [5]:

```
data = pd.read_csv('/content/drive/MyDrive/cs-2/data/file_name+label.csv')
data = data[data['label'] != '_background_noise_']
print(data.shape)
#data.head(3)
```

```
(64721, 2)
```

In [7]:

```
data.head(3)
```

Out[7]:

	file_name	label
0	five/c518d1b1_nohash_1.wav	five
1	five/f17be97f_nohash_1.wav	five
2	five/20d779bf_nohash_0.wav	five

In [ ]:

```
data = shuffle(data)
```

In [ ]:

```
# get all raw data
raw_data = []
for i in tqdm(data['file_name']):
    signal , rate = librosa.load(i , sr=16000)
    #signal = list(signal)
    raw_data.append(signal)
```

```
HBox(children=(FloatProgress(value=0.0, max=64721.0), HTML(value='')))
```

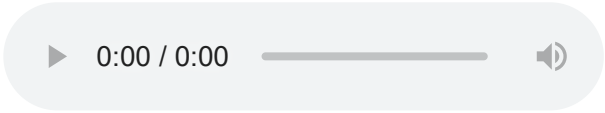
Plot Raw data

In [ ]:

```
idx = 100
samples = raw_data[idx]
sample_rate = 16000
print('Label : ',data['label'][idx:idx+1].values[0])
ipd.Audio(samples, rate = sample_rate)
```

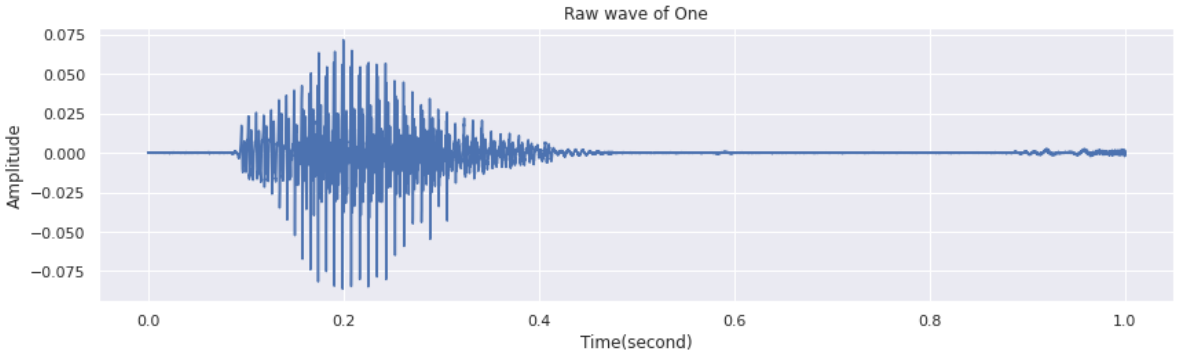
```
Label :  one
```

Out[9]:



In [ ]:

```
sns.set_theme()
fig = plt.figure(figsize=(14, 8))
ax = fig.add_subplot(211)

ax.set_title('Raw wave of '+ 'One')
ax.set_xlabel('Time(second)')
ax.set_ylabel('Amplitude')
ax.plot(np.linspace(0, sample_rate/len(samples), sample_rate), samples)
plt.show()
```



In [ ]:

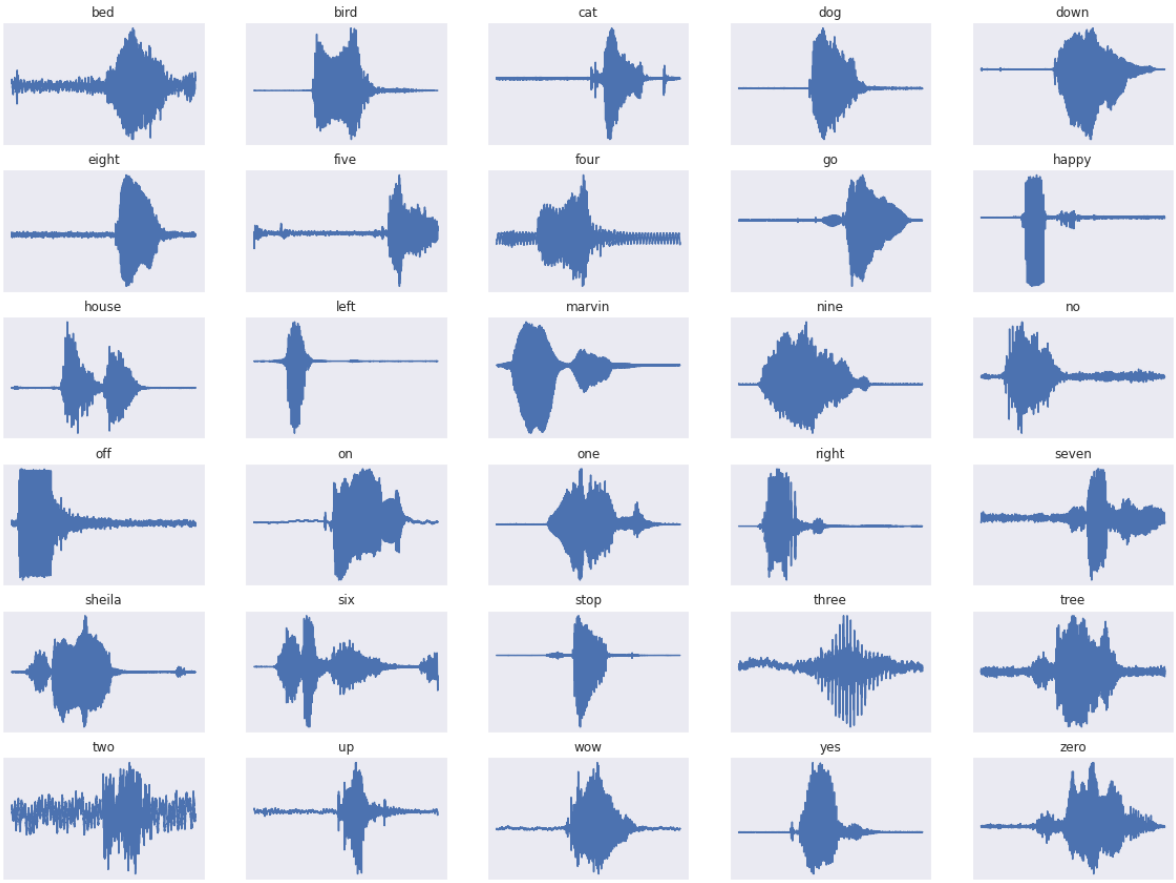
```
def plot_signals(signals):
    fig , ax = plt.subplots(nrows = 6 , ncols = 5 , figsize = (20,15))
    i = 0
    for x in range(6):
        for y in range(5):
            ax[x,y].set_title(list(signals.keys())[i])
            ax[x,y].plot(list(signals.values())[i])
            ax[x,y].get_xaxis().set_visible(False)
            ax[x,y].get_yaxis().set_visible(False)
            i += 1
```

In [ ]:

```
# store all signal in dic
signals = {}
labels = np.unique(data['label'])

# get all signal array except backgroud noise
for name in labels:
    file = data[data['label'] == name ][:1]

    signal , rate = librosa.load(file['file_name'].values[0])

    signals[name] = signal
```

In [ ]:

```
plot_signals(signals)
plt.show()
```



As we see in plot many sound wave is nearly zero at starting. So we trim that silence at staring.

Padding

In [ ]:

```
''' padding - make all file same length(16000) '''
# if len of audio is < 16000 we add zeros.
# if len of audio is > 16000 we truncate.

def padding(data, size):
    if len(data) >= size:
        arr = data[:size]
    else:
        arr = np.zeros(size)
        arr[:len(data)] = data
    return arr
```

In [ ]:

```
pad_seq = []
for i in tqdm(raw_data):
    i = padding(i,16000)
    pad_seq.append(i)
```

```
HBox(children=(FloatProgress(value=0.0, max=64721.0), HTML(value='')))
```

In [ ]:

```
#data['raw_data'] = raw_data
data['pad_seq'] = pad_seq
del raw_data, pad_seq
```

train test split

In [ ]:

```
data = shuffle(data,random_state=33)

y = data['label']
data = data.drop('label',axis = 1)

x_train , x_vali, y_train, y_vali = train_test_split(data['pad_seq'].values, y,test_si
```

In [ ]:

```
print(x_train.shape, y_train.shape)
print(x_vali.shape, y_vali.shape)
```

```
(51776,) (51776,)
(12945,) (12945,)
```

In [ ]:

```
#for error enalysis
np.save('/content/train_pad.npy', x_train)
y_train.to_csv('/content/y_train.csv', index = False)

np.save('/content/vali_pad.npy', x_vali)
y_vali.to_csv('/content/y_vali.csv', index = False)
```

In [7]:

```
del raw_train , raw_vali, raw_test
```

load data

In [ ]:

```
x_train = np.load('/content/pad_train.npy', allow_pickle=True)
x_vali  = np.load('/content/pad_vali.npy', allow_pickle=True)

y_train = pd.read_csv('/content/y_train.csv')
y_vali  = pd.read_csv('/content/y_vali.csv')
```

In [ ]:

```
print(x_train.shape, y_train.shape)
print(x_vali.shape, y_vali.shape )
```

```
(51776,) (51776,)
(12945,) (12945,)
```

log_spectrogram

In [ ]:

```
def log_spectrogram(audio, sample_rate, window_size=20,
                    step_size=10, eps=1e-10):
    nperseg = int(round(window_size * sample_rate / 1e3))
    noverlap = int(round(step_size * sample_rate / 1e3))
    freqs, times, spec = signal.spectrogram(audio,
                                    fs=sample_rate,
                                    window='hann',
                                    nperseg = nperseg,
                                    noverlap = noverlap,
                                    detrend=False)
    return freqs, times, np.log(spec.T.astype(np.float32) + eps)
```

In [ ]:

```
from scipy import signal
```

In [ ]:

```
train_data = []
for i in tqdm(x_train):
    _,_,spec = log_spectrogram(i,16000)
    train_data.append(spec)
train_data = np.array(train_data)
train_data.shape
```

HBox(children=(FloatProgress(value=0.0, max=51776.0), HTML(value='')))

Out[25]:

(51776, 99, 161)

In [ ]:

```
np.save('/content/log_train_data.npy', train_data)
del train_data
```

In [ ]:

```
vali_data = []
for i in tqdm(x_vali):
    _,_,spec = log_spectrogram(i,16000)
    vali_data.append(spec)
vali_data = np.array(vali_data)
vali_data.shape
```

HBox(children=(FloatProgress(value=0.0, max=12945.0), HTML(value='')))

Out[28]:

(12945, 99, 161)

In [ ]:

```
np.save('/content/log_vali_data.npy', vali_data)
```

Load Final Files

In [ ]:

```
train_data = np.load('/content/log_train_data.npy', allow_pickle=True)
vali_data  = np.load('/content/log_vali_data.npy', allow_pickle=True)

y_train = pd.read_csv('/content/y_train.csv')
y_vali = pd.read_csv('/content/y_vali.csv')
```

In [ ]:

```
# reshape because we use conv2d Here
train_data = train_data.reshape(tuple(list(train_data.shape) + [1] ))
train_data.shape
```

Out[3]:

```
(51776, 99, 161, 1)
```

In [ ]:

```
vali_data = vali_data.reshape(tuple(list(vali_data.shape) + [1] ))
vali_data.shape
```

Out[4]:

```
(12945, 99, 161, 1)
```

In [ ]:

```
total = y_train.append(y_vali)
print(len(total.label.unique()))
total.label.unique()
```

```
30
```

Out[5]:

```
array(['go', 'left', 'tree', 'eight', 'yes', 'cat', 'two', 'up', 'down',
       'no', 'dog', 'sheila', 'happy', 'off', 'on', 'nine', 'three',
       'zero', 'bird', 'seven', 'house', 'bed', 'six', 'stop', 'one',
       'marvin', 'right', 'wow', 'five', 'four'], dtype=object)
```

encoder

In [ ]:

```
encoder = LabelEncoder()
encoder.fit(total)
y_train = encoder.transform(y_train)
y_vali = encoder.transform(y_vali)
#y_test = encoder.transform(y_test)
```

In [ ]:

```
y_train.shape , y_vali.shape
```

Out[7]:

```
((51776,), (12945,))
```

Model

In [ ]:

```
model_7 = tf.keras.Sequential()

model_7.add(Input(shape=(99,161,1)))
model_7.add(BatchNormalization())
model_7.add(Conv2D(8,kernel_size=2,activation='relu',kernel_regularizer=tf.keras.regul
model_7.add(Conv2D(8,kernel_size=3,activation='relu',kernel_regularizer=tf.keras.regul
model_7.add(MaxPooling2D(pool_size=(2,2)))
model_7.add(Dropout(rate=0.2))

model_7.add(Conv2D(16,kernel_size=5,activation='relu',kernel_regularizer=tf.keras.regu
model_7.add(Conv2D(16,kernel_size=6,activation='relu',kernel_regularizer=tf.keras.regu
model_7.add(MaxPooling2D(pool_size=(2,2)))
model_7.add(Dropout(rate=0.2))

model_7.add(Conv2D(16,kernel_size=6,activation='relu',kernel_regularizer=tf.keras.regu
model_7.add(MaxPooling2D(pool_size=(2,2)))

model_7.add(Flatten())
model_7.add(Dense(256, activation='relu'))
model_7.add(BatchNormalization())
model_7.add(Dense(128, activation='relu'))
model_7.add(BatchNormalization())
model_7.add(Dense(30, activation='softmax'))
```

In [ ]:

```
1 model_7.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
batch_normalization (BatchNo (None, 99, 161, 1)        4
_________________________________________________________________
conv2d (Conv2D)              (None, 98, 160, 8)        40
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 96, 158, 8)        584
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 48, 79, 8)         0
_________________________________________________________________
dropout (Dropout)            (None, 48, 79, 8)         0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 44, 75, 16)        3216
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 39, 70, 16)        9232
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 19, 35, 16)        0
_________________________________________________________________
dropout_1 (Dropout)          (None, 19, 35, 16)        0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 14, 30, 16)        9232
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 7, 15, 16)         0
_________________________________________________________________
flatten (Flatten)            (None, 1680)              0
_________________________________________________________________
dense (Dense)                (None, 256)               430336
_________________________________________________________________
batch_normalization_1 (Batch (None, 256)               1024
_________________________________________________________________
dense_1 (Dense)              (None, 128)               32896
_________________________________________________________________
batch_normalization_2 (Batch (None, 128)               512
_________________________________________________________________
dense_2 (Dense)              (None, 30)                3870
=================================================================
Total params: 490,946
Trainable params: 490,176
Non-trainable params: 770
_________________________________________________________________
```

In [ ]:

```
tensorboard_callback = TensorBoard(log_dir='/content/model-8-3',write_grads=True,histo

def scheduler(epoch, lr):
    if epoch < 5:
        return lr
    else:
        return lr * 0.9

lrs = tf.keras.callbacks.LearningRateScheduler(scheduler,verbose=1)
callbacks = [tensorboard_callback,lrs]
```

```
WARNING:tensorflow:`write_grads` will be ignored in TensorFlow 2.0 for the `
TensorBoard` Callback.
```

In [ ]:

```
optimizer = tf.keras.optimizers.Adam(learning_rate=0.001)
opt = tf.keras.optimizers.SGD(learning_rate=0.001, momentum=0.9)
model_7.compile(optimizer=optimizer,loss='sparse_categorical_crossentropy', metrics =
```

In [ ]:

```
history = model_7.fit( train_data ,y_train,
                    epochs = 20,
                    validation_data = ( vali_data ,y_vali) ,
                    callbacks = callbacks)
```

```
Epoch 1/20

Epoch 00001: LearningRateScheduler reducing learning rate to 0.00100000004
74974513.
1618/1618 [==============================] - 57s 15ms/step - loss: 2.2587
- accuracy: 0.3725 - val_loss: 0.6978 - val_accuracy: 0.8088
Epoch 2/20

Epoch 00002: LearningRateScheduler reducing learning rate to 0.00100000004
74974513.
1618/1618 [==============================] - 22s 14ms/step - loss: 0.7165
- accuracy: 0.8044 - val_loss: 0.4898 - val_accuracy: 0.8742
Epoch 3/20

Epoch 00003: LearningRateScheduler reducing learning rate to 0.00100000004
74974513.
1618/1618 [==============================] - 22s 14ms/step - loss: 0.5403
- accuracy: 0.8530 - val_loss: 0.4718 - val_accuracy: 0.8774
Epoch 4/20

Epoch 00004: LearningRateScheduler reducing learning rate to 0.00100000004
74974513.
1618/1618 [==============================] - 22s 14ms/step - loss: 0.4637
- accuracy: 0.8787 - val_loss: 0.4215 - val_accuracy: 0.9003
Epoch 5/20

Epoch 00005: LearningRateScheduler reducing learning rate to 0.00100000004
74974513.
1618/1618 [==============================] - 23s 14ms/step - loss: 0.4171
- accuracy: 0.8936 - val_loss: 0.4843 - val_accuracy: 0.8793
Epoch 6/20

Epoch 00006: LearningRateScheduler reducing learning rate to 0.00090000004
27477062.
1618/1618 [==============================] - 23s 14ms/step - loss: 0.3848
- accuracy: 0.9029 - val_loss: 0.4282 - val_accuracy: 0.8949
Epoch 7/20

Epoch 00007: LearningRateScheduler reducing learning rate to 0.00081000003
84729356.
1618/1618 [==============================] - 23s 14ms/step - loss: 0.3505
- accuracy: 0.9135 - val_loss: 0.3838 - val_accuracy: 0.9080
Epoch 8/20

Epoch 00008: LearningRateScheduler reducing learning rate to 0.00072900005
03417104.
1618/1618 [==============================] - 23s 14ms/step - loss: 0.3164
- accuracy: 0.9237 - val_loss: 0.3450 - val_accuracy: 0.9190
Epoch 9/20

Epoch 00009: LearningRateScheduler reducing learning rate to 0.00065610007
15009868.
1618/1618 [==============================] - 23s 14ms/step - loss: 0.2878
- accuracy: 0.9297 - val_loss: 0.3622 - val_accuracy: 0.9130
```

```
Epoch 10/20

Epoch 00010: LearningRateScheduler reducing learning rate to 0.00059049004
33961303.
1618/1618 [==============================] - 23s 14ms/step - loss: 0.2725
- accuracy: 0.9353 - val_loss: 0.3316 - val_accuracy: 0.9224
Epoch 11/20

Epoch 00011: LearningRateScheduler reducing learning rate to 0.00053144105
47725857.
1618/1618 [==============================] - 23s 14ms/step - loss: 0.2551
- accuracy: 0.9374 - val_loss: 0.3238 - val_accuracy: 0.9214
Epoch 12/20

Epoch 00012: LearningRateScheduler reducing learning rate to 0.00047829695
977270604.
1618/1618 [==============================] - 23s 14ms/step - loss: 0.2375
- accuracy: 0.9422 - val_loss: 0.3242 - val_accuracy: 0.9236
Epoch 13/20

Epoch 00013: LearningRateScheduler reducing learning rate to 0.00043046725
33180565.
1618/1618 [==============================] - 23s 14ms/step - loss: 0.2252
- accuracy: 0.9452 - val_loss: 0.3025 - val_accuracy: 0.9306
Epoch 14/20

Epoch 00014: LearningRateScheduler reducing learning rate to 0.00038742052
274756136.
1618/1618 [==============================] - 23s 14ms/step - loss: 0.2040
- accuracy: 0.9510 - val_loss: 0.3108 - val_accuracy: 0.9251
Epoch 15/20

Epoch 00015: LearningRateScheduler reducing learning rate to 0.00034867847
57114947.
1618/1618 [==============================] - 23s 14ms/step - loss: 0.1973
- accuracy: 0.9539 - val_loss: 0.3004 - val_accuracy: 0.9285
Epoch 16/20

Epoch 00016: LearningRateScheduler reducing learning rate to 0.00031381062
290165574.
1618/1618 [==============================] - 23s 14ms/step - loss: 0.1857
- accuracy: 0.9560 - val_loss: 0.2937 - val_accuracy: 0.9301
Epoch 17/20

Epoch 00017: LearningRateScheduler reducing learning rate to 0.00028242956
32308349.
1618/1618 [==============================] - 23s 14ms/step - loss: 0.1723
- accuracy: 0.9610 - val_loss: 0.2984 - val_accuracy: 0.9244
Epoch 18/20

Epoch 00018: LearningRateScheduler reducing learning rate to 0.00025418660
952709616.
1618/1618 [==============================] - 23s 14ms/step - loss: 0.1718
- accuracy: 0.9601 - val_loss: 0.2963 - val_accuracy: 0.9293
Epoch 19/20

Epoch 00019: LearningRateScheduler reducing learning rate to 0.00022876793
809700757.
1618/1618 [==============================] - 23s 14ms/step - loss: 0.1584
- accuracy: 0.9645 - val_loss: 0.2992 - val_accuracy: 0.9294
Epoch 20/20
```

```
Epoch 00020: LearningRateScheduler reducing learning rate to 0.00020589114
428730683.
1618/1618 [==============================] - 23s 14ms/step - loss: 0.1578
- accuracy: 0.9646 - val_loss: 0.2910 - val_accuracy: 0.9303
```

In [ ]:

```
model_7.save('/content/drive/MyDrive/cs-2/final_model.h5')
np.save('/content/drive/MyDrive/cs-2/final_model_classes.npy',encoder.classes_)
```

In [ ]:

```
#model 9
%reload_ext tensorboard
%tensorboard --logdir /content/model-8-3

```

```
<IPython.core.display.Javascript object>
```

Prediction on Validation data

In [ ]:

```
predictions = model_7.predict_classes(vali_data, verbose=False)

prediction_labels = encoder.inverse_transform(predictions)

y = encoder.inverse_transform(y_vali)
```

In [ ]:

```
df = pd.DataFrame()
df['prediction'] = prediction_labels
df['true_labels'] = y
```

In [ ]:

```
df.head()
```

Out[20]:

	prediction	true_labels
0	four	four
1	on	on
2	nine	nine
3	marvin	marvin
4	stop	stop

In [ ]:

```
df.to_csv('/content/df.csv', index=False)
```

In [ ]:

```
# load validation pad data for plotting
pad_vali = np.load('/content/vali_pad.npy', allow_pickle=True)
```

In [ ]:

```
df['pad_data'] = pad_vali
```

Create Dataframe with misclassified points

In [ ]:

```
FP = df[df['prediction'] != df['true_labels']]
FP.shape
```

Out[29]:

```
(902, 3)
```

In [ ]:

```
FP.head()
```

Out[30]:

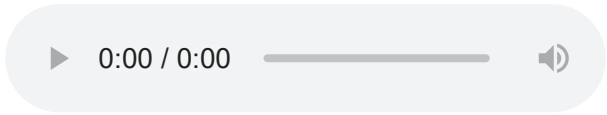
	prediction	true_labels	pad_data
29	on	one	[-3.0517578e-05, -9.1552734e-05, -9.1552734e-0...
53	dog	down	[0.0049743652, 0.0056762695, 0.005432129, 0.00...
61	nine	marvin	[-0.00021362305, -0.0014343262, -0.002105713, ...
62	up	cat	[0.0, 0.0, 0.0, 3.0517578e-05, 0.0, 0.0, 0.0, ...
83	nine	marvin	[-3.0517578e-05, -3.0517578e-05, 6.1035156e-05...

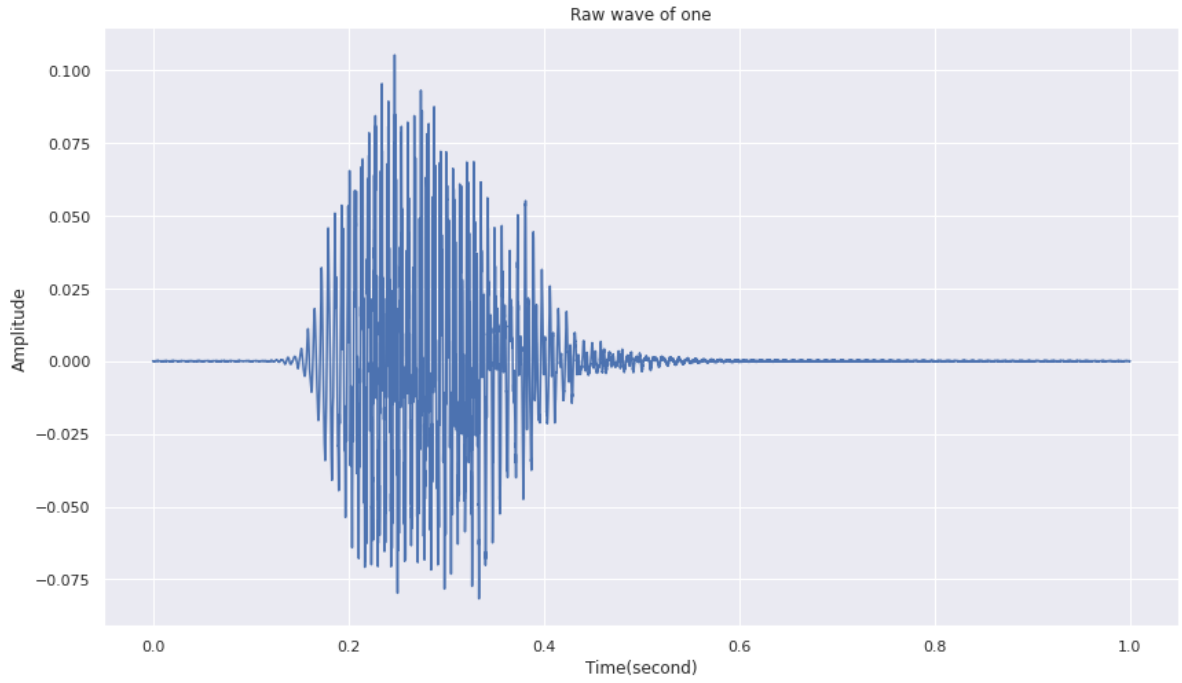
Plot Misclassified Points

In [ ]:

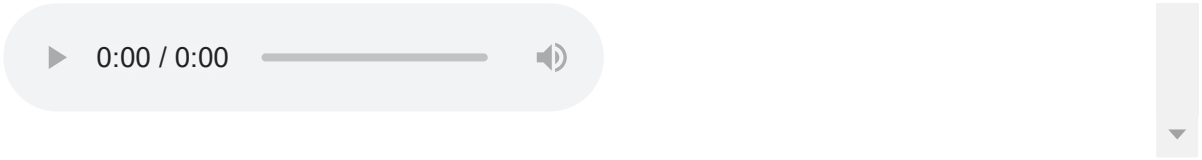
```
sns.set_theme()

for i in range(20):
    idx = i
    print()
    samples = FP['pad_data'].values[idx]
    sample_rate = 16000
    print('True Label      : ',FP['true_labels'][idx:idx+1].values[0])
    print('Predicted label : ', FP['prediction'][idx:idx+1].values[0])

    ipd.display(ipd.Audio(samples, rate = sample_rate))
    print()


    a = 211+i
    fig = plt.figure(figsize=(14, 8))

    plt.title('Raw wave of '+ FP['true_labels'][idx:idx+1].values[0])
    plt.xlabel('Time(second)')
    plt.ylabel('Amplitude')
    plt.plot(np.linspace(0, sample_rate/len(samples), sample_rate), samples)
    plt.show()
```

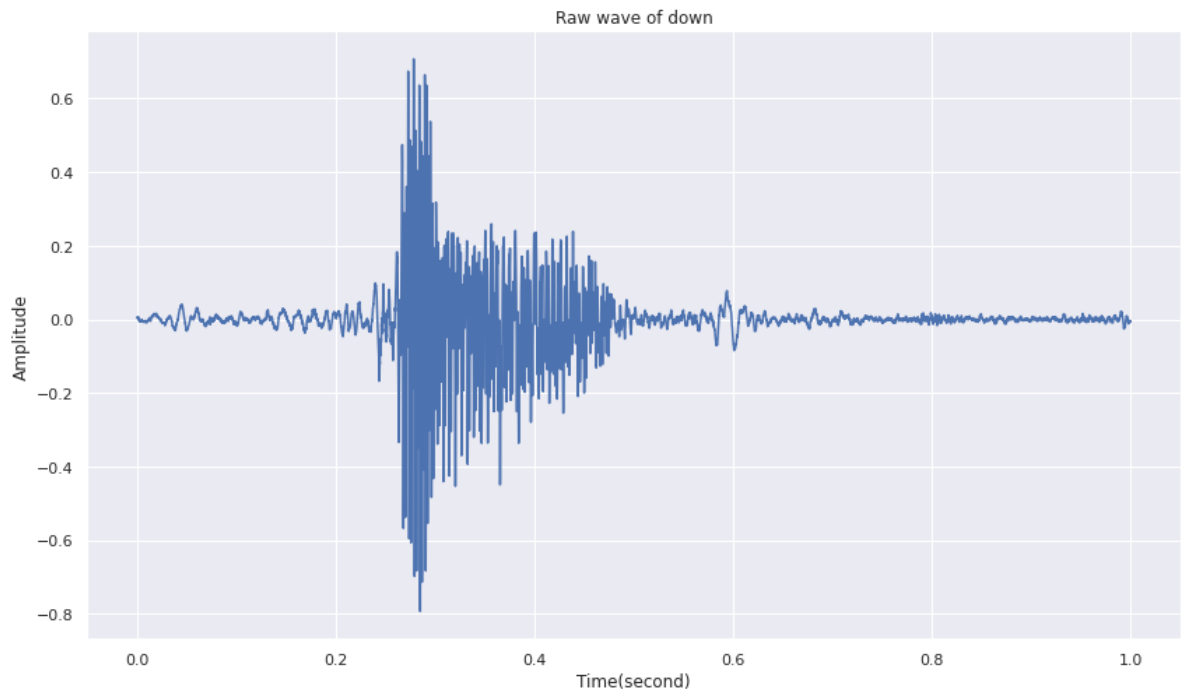
```
True Label      :  one
Predicted label :  on
```



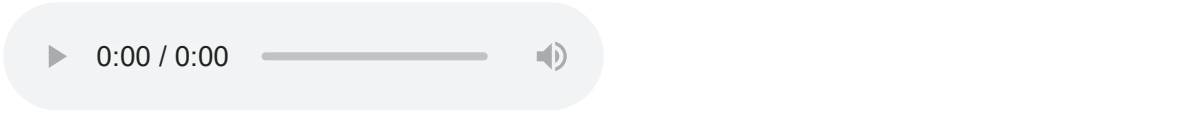


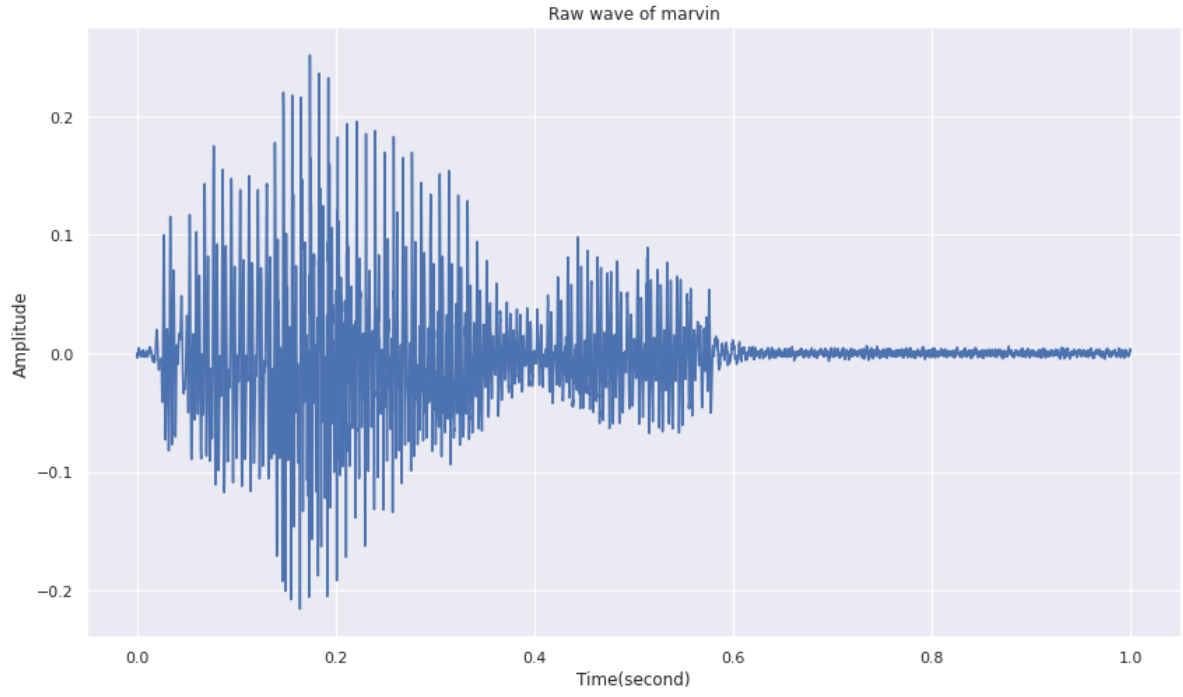

```
True Label      :  down
Predicted label :  dog
```





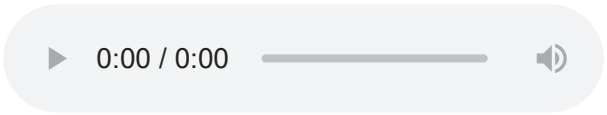

```
True Label      :  marvin
Predicted label :  nine
```

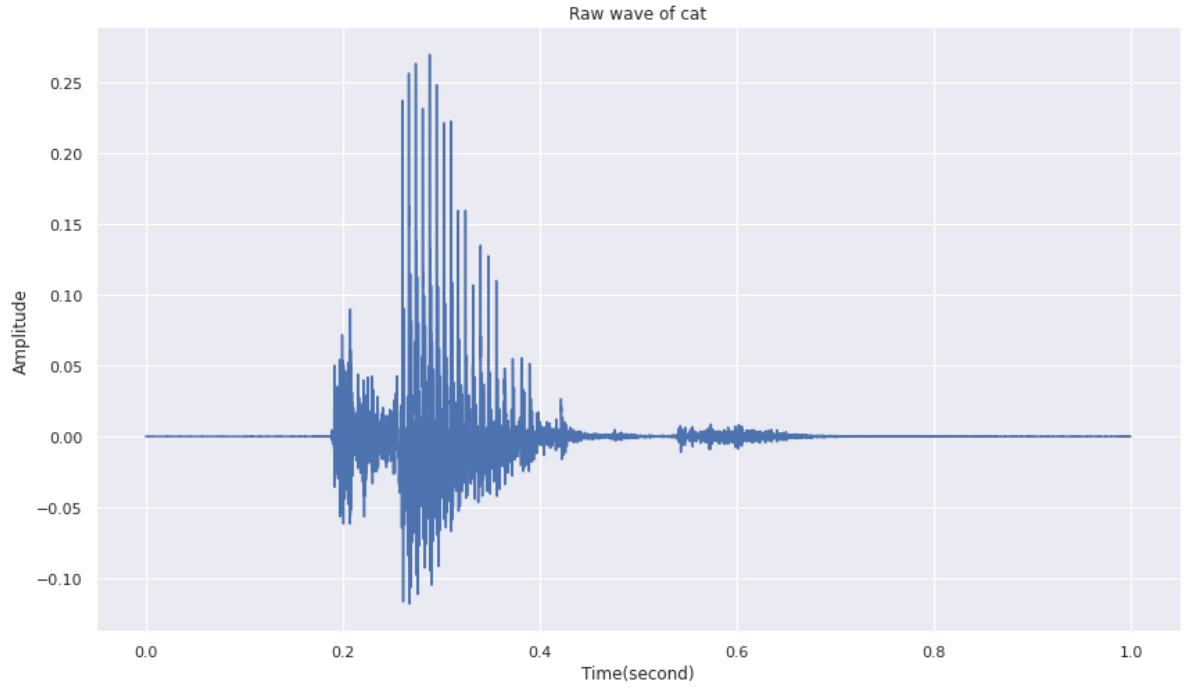




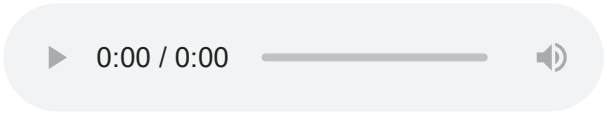

```
True Label      :  cat
```

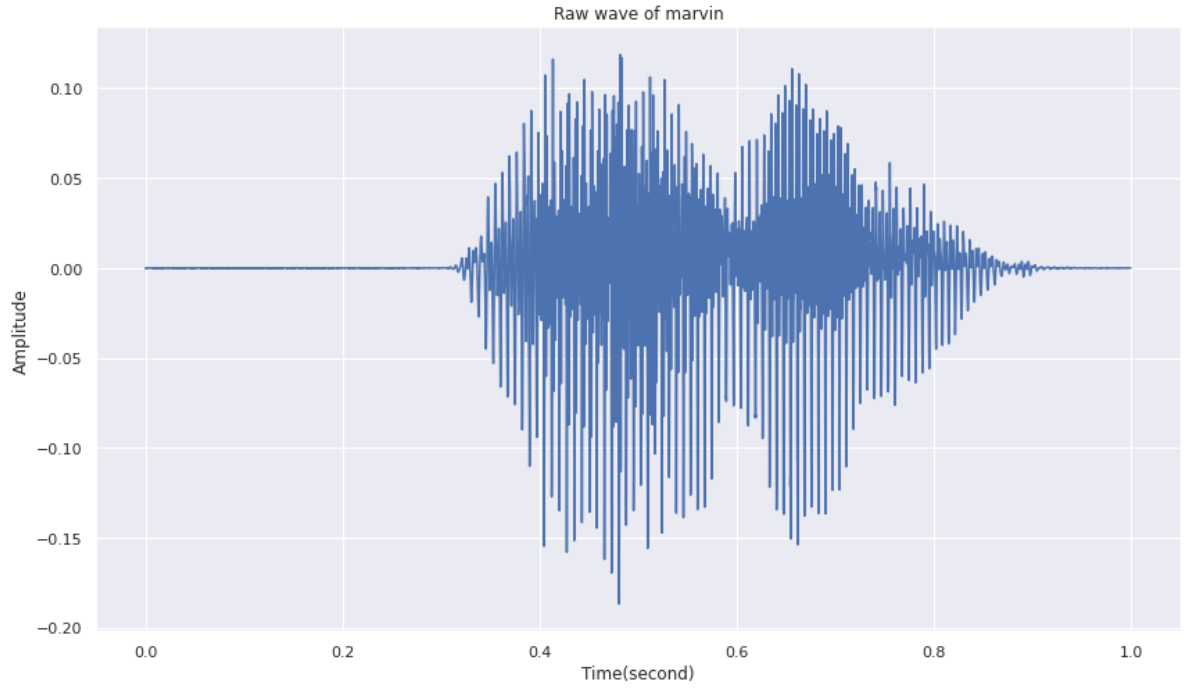
```
Predicted label :  up
```



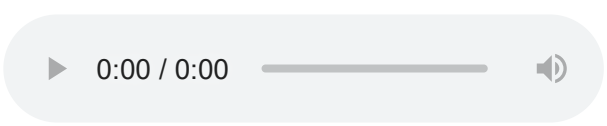


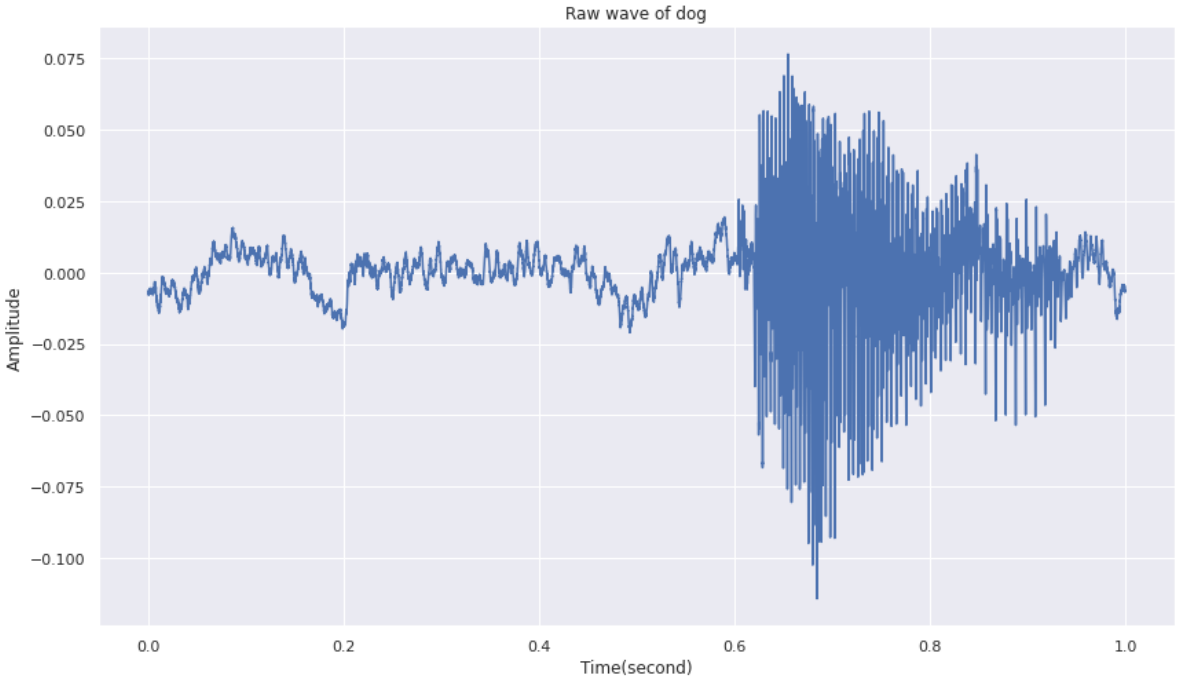

```
True Label      :  marvin
Predicted label :  nine
```



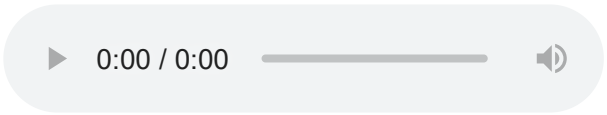


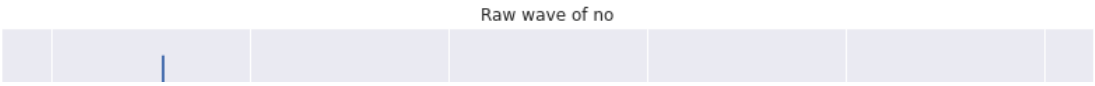
```
True Label      :  dog
Predicted label :  down
```



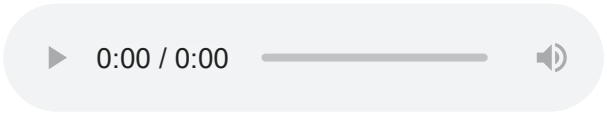


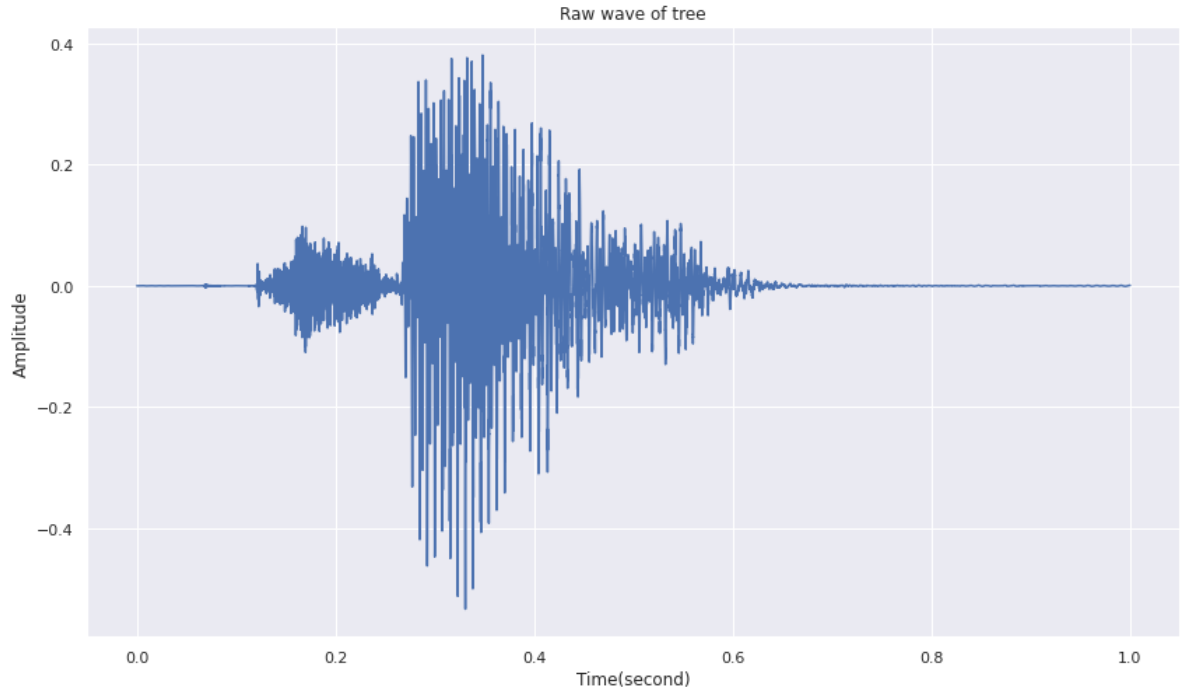

```
True Label      :  no
Predicted label :  down
```



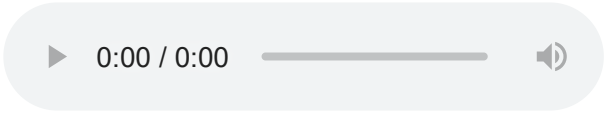



```
True Label      :  tree
Predicted label :  three
```

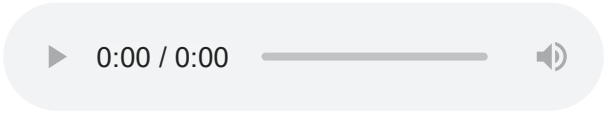


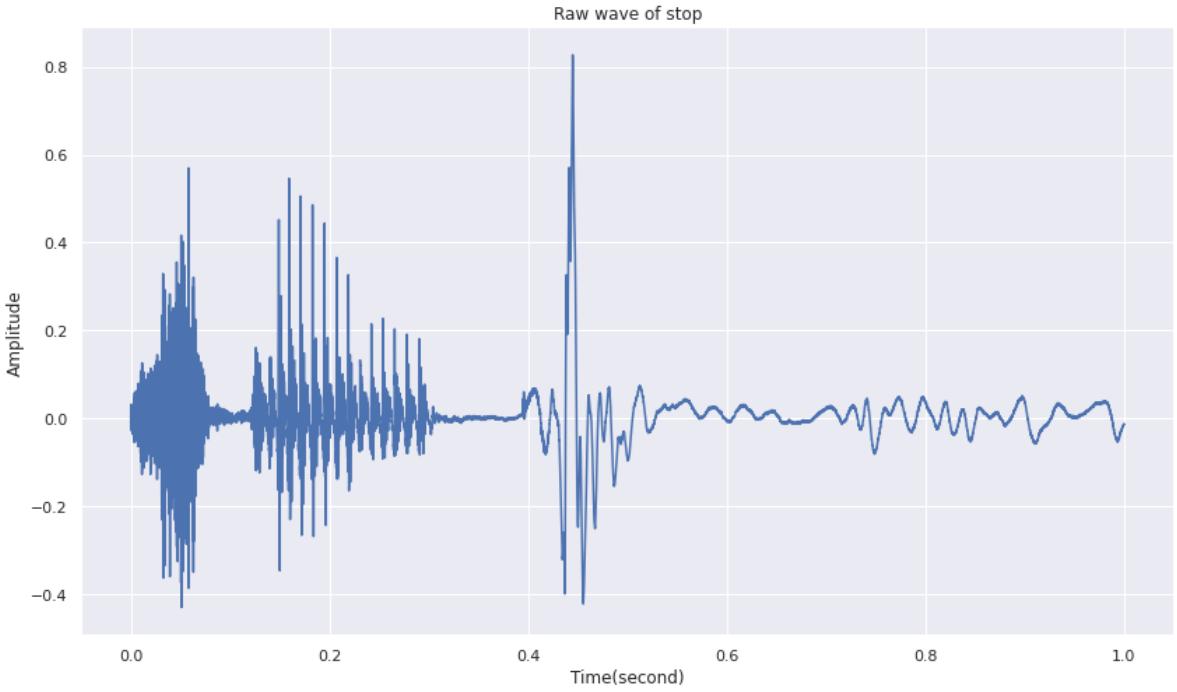



```
True Label      :  wow
Predicted label :  no
```

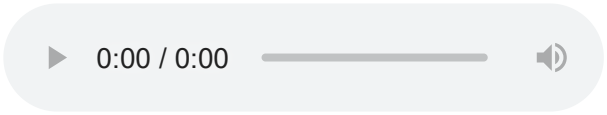



```
True Label      :  stop
Predicted label :  seven
```

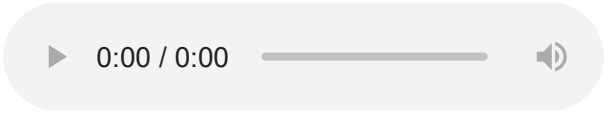


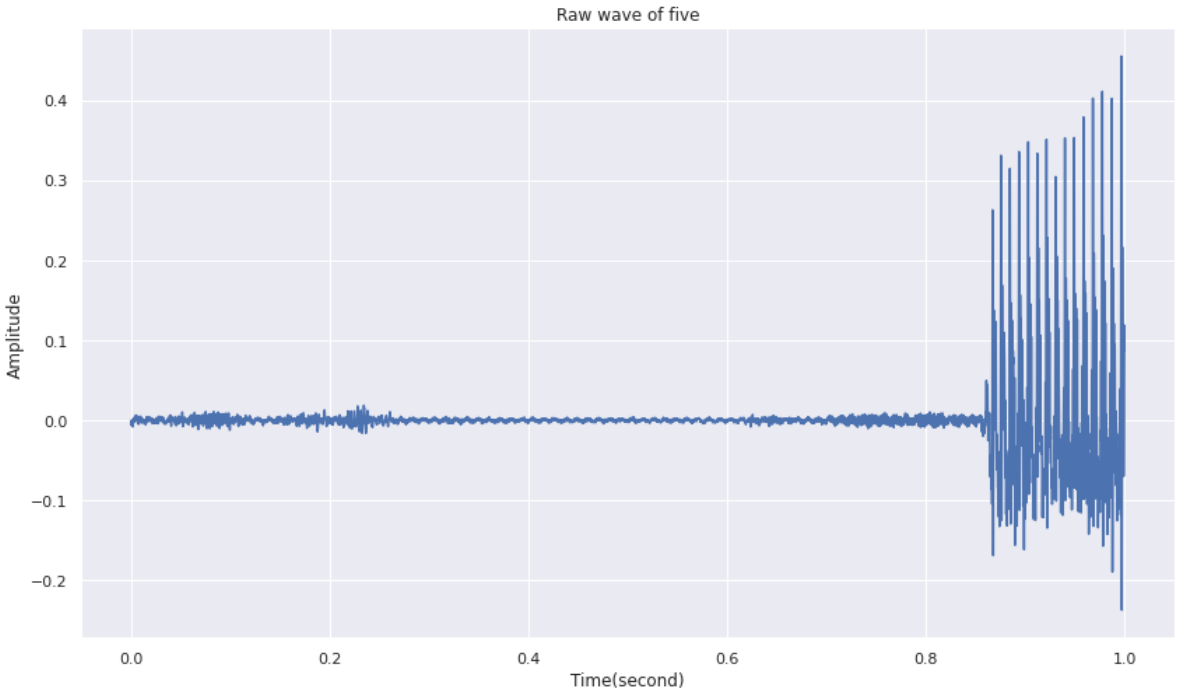



```
True Label      :  on
Predicted label :  one
```

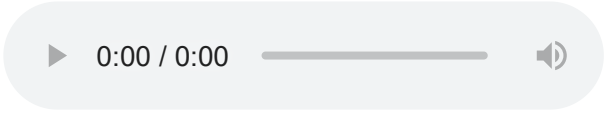


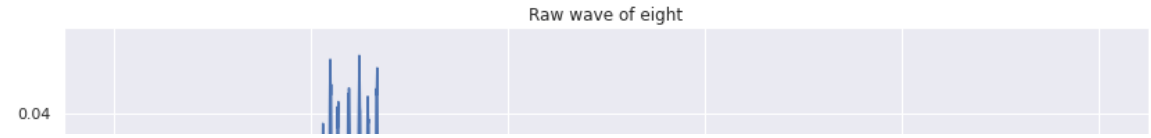

```
True Label      :  five
Predicted label :  off
```



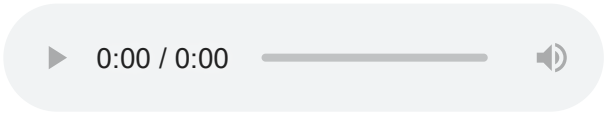


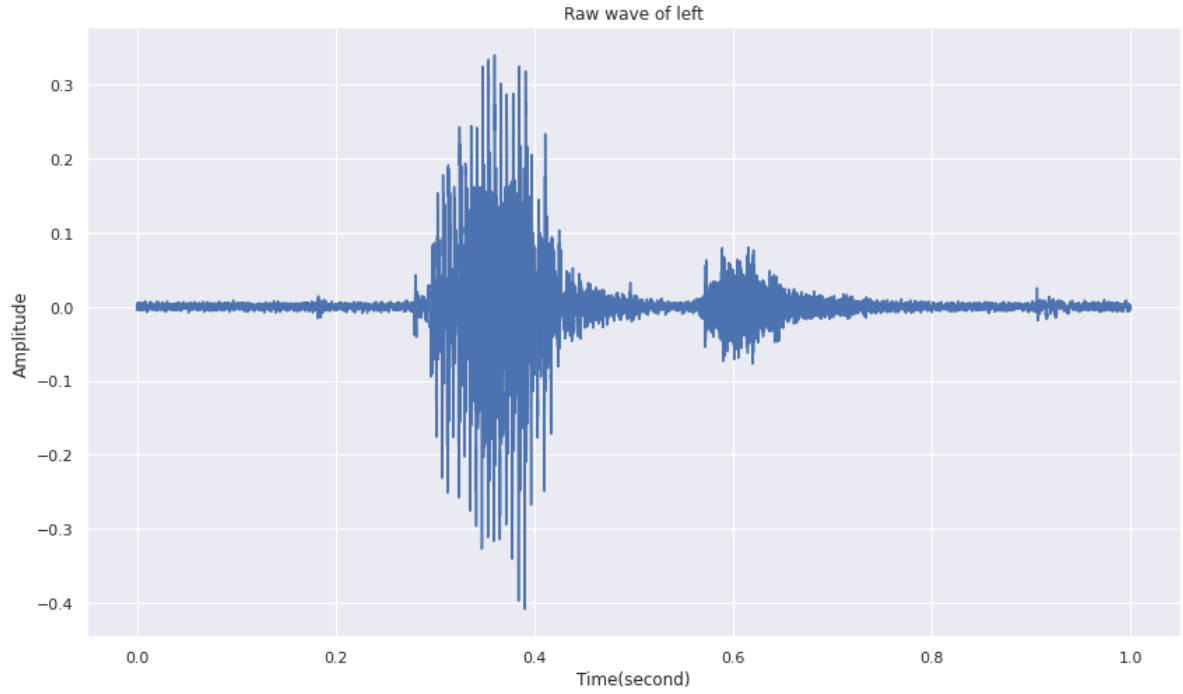

```
True Label      :  eight
Predicted label :  right
```



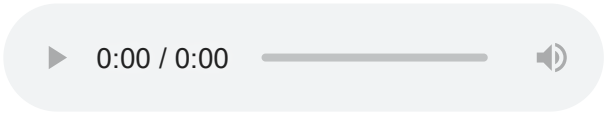


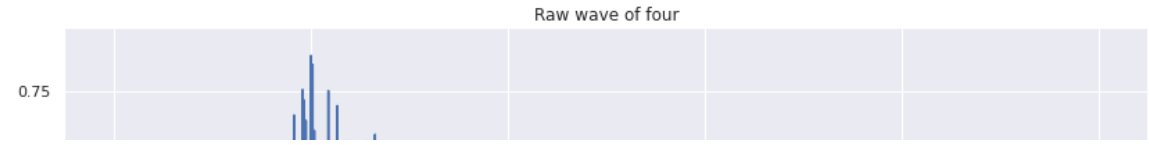

```
True Label      :  left
Predicted label :  up
```



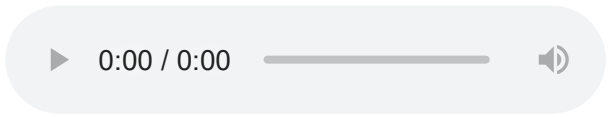


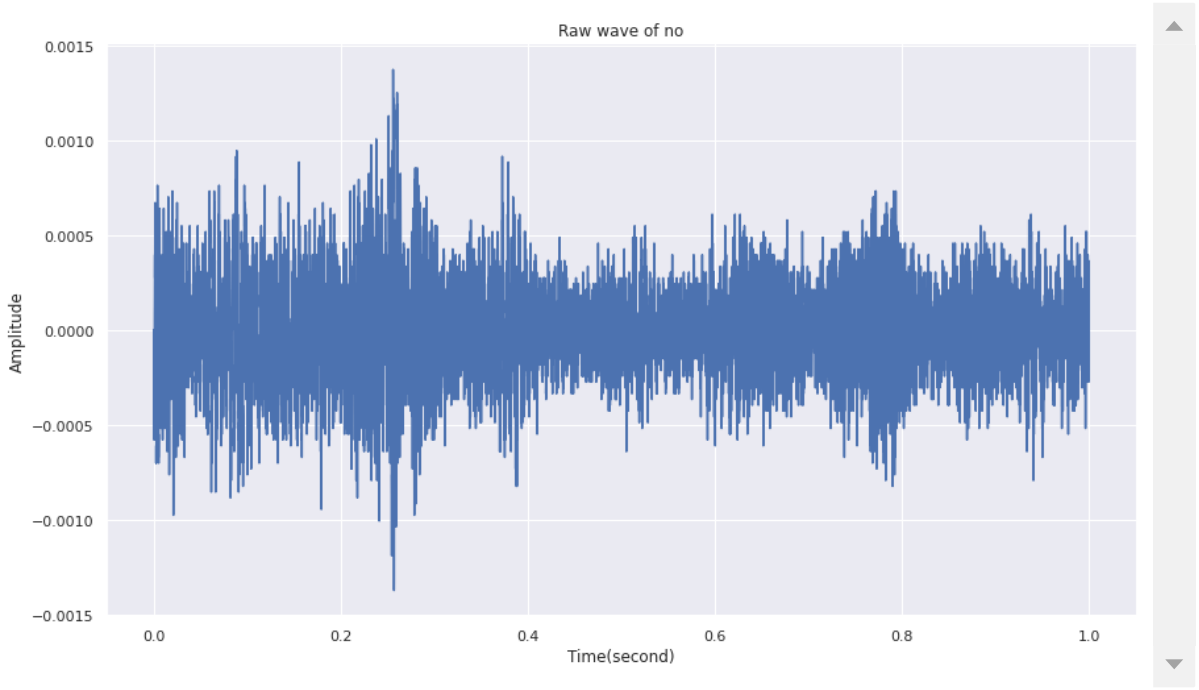

```
True Label      :  four
Predicted label :  stop
```



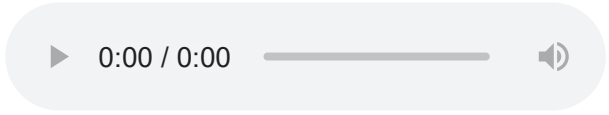


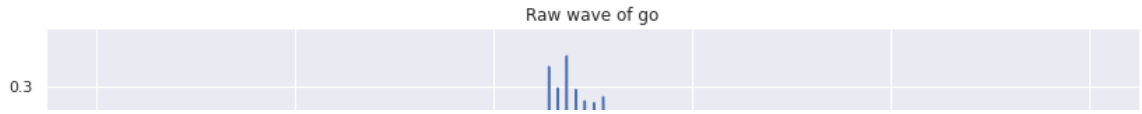

```
True Label      :  no
Predicted label :  right
```



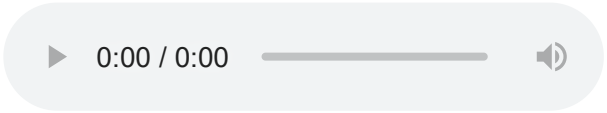


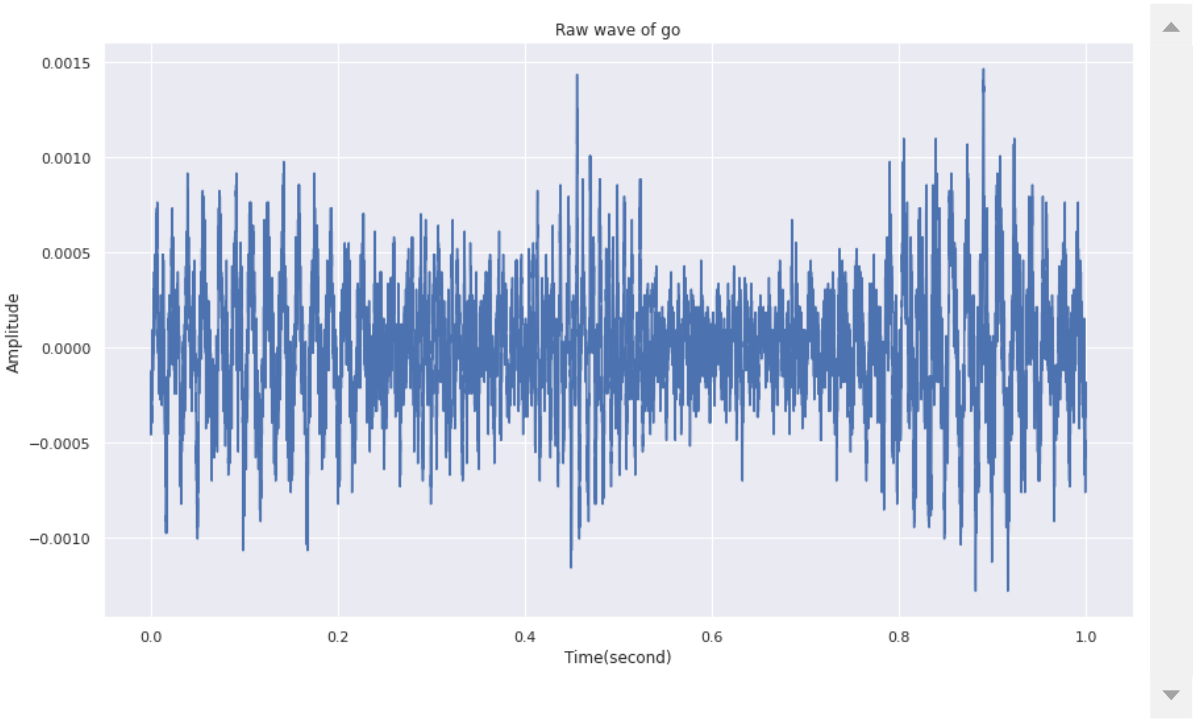

```
True Label      :  go
Predicted label :  no
```



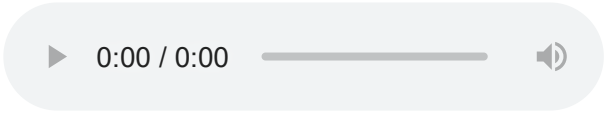



```
True Label      :  go
Predicted label :  right
```



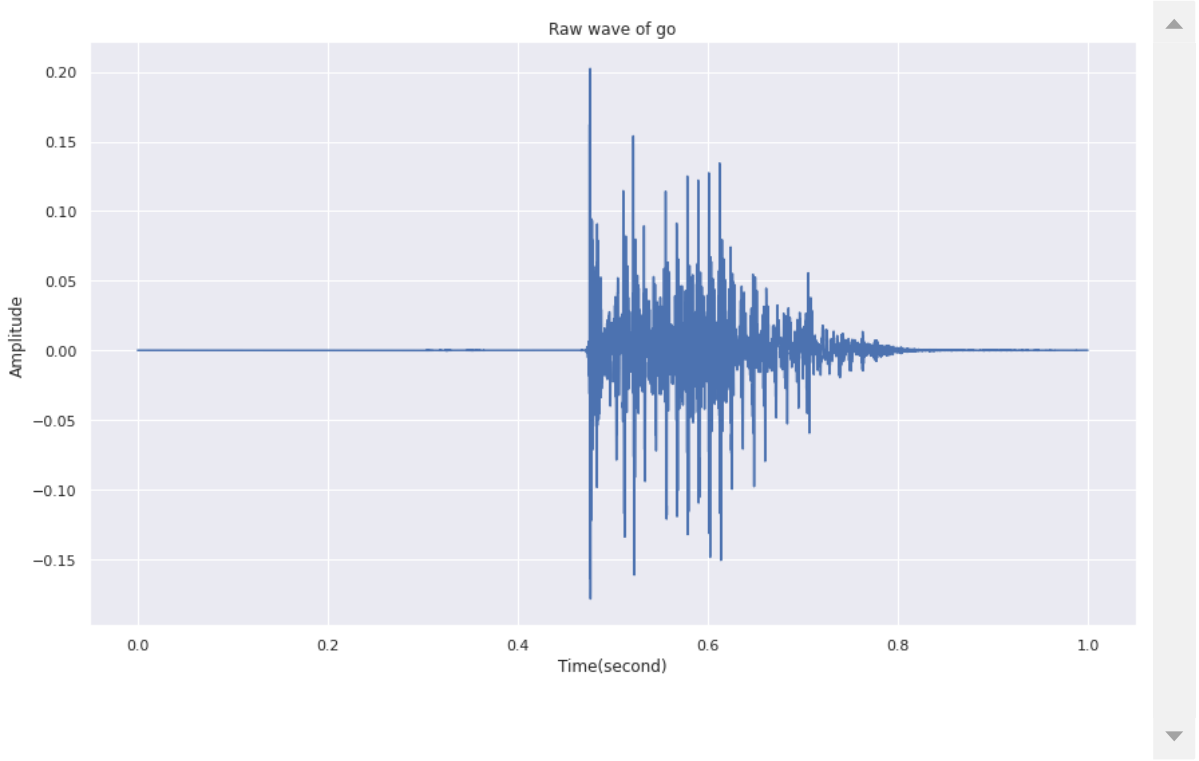



```
True Label      :  right
Predicted label :  left
```




```
True Label      :  go
Predicted label :  dog
```

0:00 / 0:00



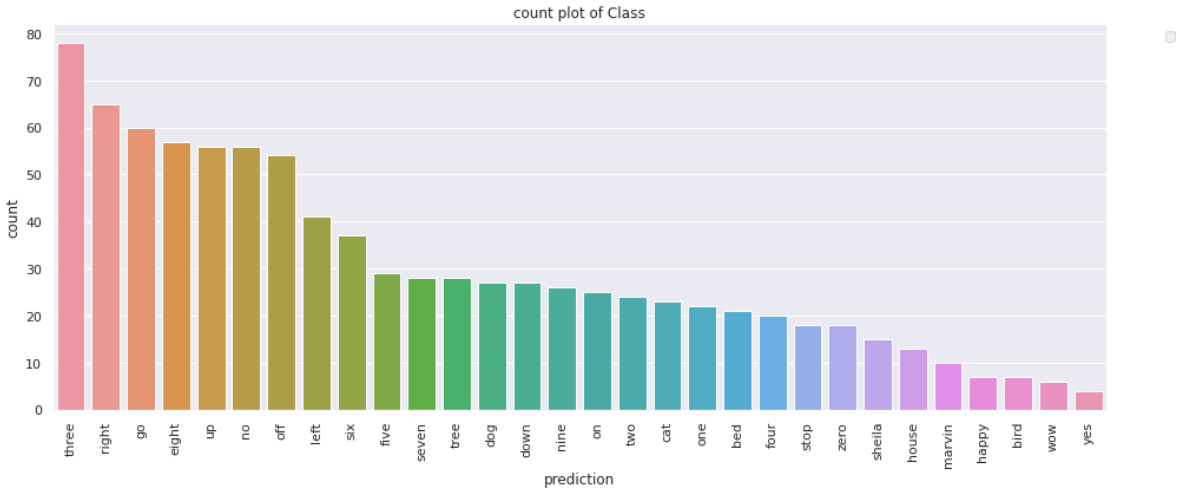
Count plot of misclassified points

In [ ]:

```
def plot(column_name , data, title):
    #get unique_values of each categories
    dic = Counter(data[column_name].values)
    #sort in deccending order by values
    dic = sorted(dic.items(), key=lambda x: x[1], reverse=True)
    column_list = []


    #get name of sorted dic
    for name in dic:
        column_list.append(name[0])

    plt.figure(figsize = (16,6))
    ax = sns.countplot(x = column_name ,  data = data , order = column_list, dodge = F

    h,l = ax.get_legend_handles_labels()
    ax.legend(h ,column_list,bbox_to_anchor=(1.05, 1) ,loc = 'upper left')
    plt.setp(ax.get_xticklabels() , rotation = 90 )
    plt.title('count plot of {}'.format(title))
    plt.show()
    return dic
```

In [ ]:

```
dic = plot('prediction' , FP, 'Class')
```



In [ ]:

```
y_unique = list(set(list(FP['prediction'].values)))
len(y_unique)
```

Out[73]:

```
30
```

Classification_report

In [ ]:

```
matrix = classification_report(df['true_labels'].values,df['prediction'].values)
```

In [ ]:

```
print(matrix)
```

```
              precision    recall  f1-score   support

         bed       0.94      0.93      0.94       343
        bird       0.98      0.93      0.95       346
         cat       0.94      0.96      0.95       347
         dog       0.92      0.92      0.92       349
        down       0.94      0.88      0.91       472
       eight       0.89      0.96      0.92       470
        five       0.94      0.91      0.92       471
        four       0.96      0.96      0.96       475
          go       0.88      0.91      0.89       474
       happy       0.98      0.95      0.97       348
       house       0.96      0.95      0.96       350
        left       0.92      0.94      0.93       471
      marvin       0.97      0.94      0.95       349
        nine       0.94      0.93      0.94       473
          no       0.89      0.94      0.91       475
         off       0.89      0.93      0.91       471
          on       0.94      0.86      0.90       473
         one       0.95      0.94      0.95       474
       right       0.87      0.95      0.91       473
       seven       0.94      0.94      0.94       476
      sheila       0.96      0.97      0.96       347
         six       0.92      0.95      0.94       474
        stop       0.96      0.92      0.94       476
       three       0.84      0.90      0.87       471
        tree       0.91      0.83      0.87       347
         two       0.95      0.95      0.95       475
          up       0.88      0.90      0.89       475
         wow       0.98      0.96      0.97       349
         yes       0.99      0.95      0.97       476
        zero       0.96      0.94      0.95       475

    accuracy                           0.93     12945
   macro avg       0.93      0.93      0.93     12945
weighted avg       0.93      0.93      0.93     12945
```

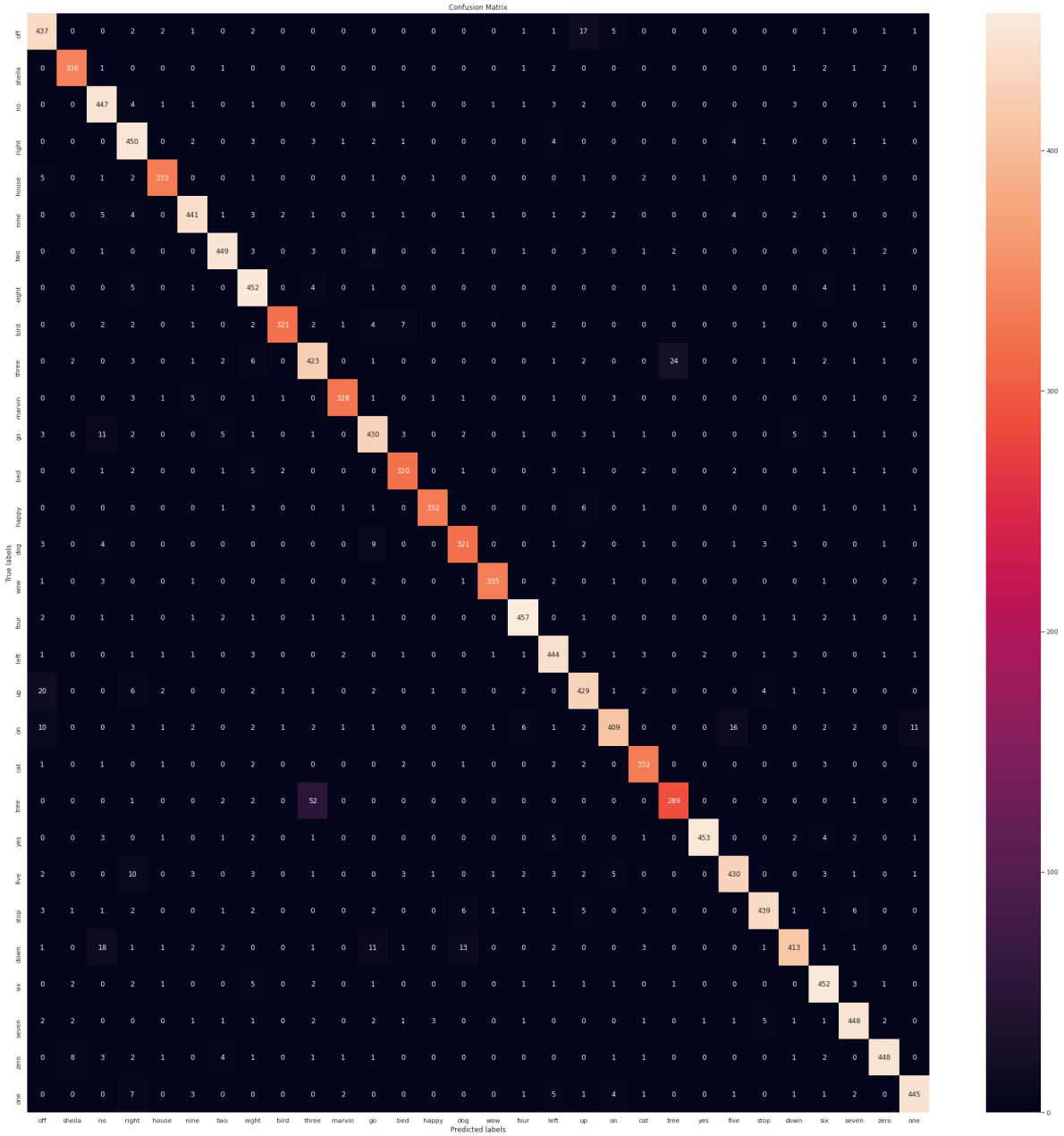
Confusion Matrix

In [ ]:

```
#from sklearn.metrics import confusion_matrix
confusion = confusion_matrix(df['true_labels'].values, df['prediction'].values, labels
```

In [ ]:

```
plt.figure(figsize = (35,35))
ax= plt.subplot()
sns.heatmap(confusion, annot=True, fmt='g', ax=ax);  #annot=True to annotate cells, ft

# labels, title and ticks
ax.set_xlabel('Predicted labels');ax.set_ylabel('True labels');
ax.set_title('Confusion Matrix');
ax.xaxis.set_ticklabels(y_unique); ax.yaxis.set_ticklabels(y_unique);
```



Error Analysis

1. All misclassification points have very low amplitude.
2. Amplitude means : The amplitude of a wave is related to the amount of energy it carries. A low amplitude means wave of signal carries a small amount of energy.
3. Many True label are also labeled incorrectly. Ex: 3rd data point is labeled as marvie(Ture label) and predicted as nine and it is actual nine(concluded by listen). **Here many ground truth labels are wrong.** (cause of error : human error)
4. Many points are silent but labeled as it is not silent.
5. As we see in count plot highly misclassified label is three. and very low misclassified label is yes.
6. From confusion matrix we say that model is not able to classify tree as tree and three as three.
7. we can see in confusion matrix highest misclassification happen in three and tree(both are very similar in pronunciation).
8. From Classification Report we can conclude that,

- **Model Perform Best**(where F1 score >= 95) in this categories : [bird, cat, four, happy, house, marvin, one, sheila, two, wow, yes, zero]
- **Model Perform Medium**(where F1 score >= 90 and < 95) in this categories : [bed, dog, down, eight, five, left, nine, no, off, on, right, seven, six, stop]
- **Model Perform Worst**(maximum misclassify, where F1 score < 91) in this categories : [go, three, tree, up]