PDL LABSHEET-8

NAME:ASHIKA.C

ROLL NO : 225229105

In [1]:
```
!pip install librosa
```

```
Requirement already satisfied: librosa in /usr/local/lib/python3.10/dist
-packages (0.10.0.post2)
Requirement already satisfied: audioread>=2.1.9 in /usr/local/lib/python
3.10/dist-packages (from librosa) (3.0.0)
Requirement already satisfied: numpy!=1.22.0,!=1.22.1,!=1.22.2,>=1.20.3
in /usr/local/lib/python3.10/dist-packages (from librosa) (1.22.4)
Requirement already satisfied: scipy>=1.2.0 in /usr/local/lib/python3.1
0/dist-packages (from librosa) (1.10.1)
Requirement already satisfied: scikit-learn>=0.20.0 in /usr/local/lib/py
thon3.10/dist-packages (from librosa) (1.2.2)
Requirement already satisfied: joblib>=0.14 in /usr/local/lib/python3.1
0/dist-packages (from librosa) (1.3.1)
Requirement already satisfied: decorator>=4.3.0 in /usr/local/lib/python
3.10/dist-packages (from librosa) (4.4.2)
Requirement already satisfied: numba>=0.51.0 in /usr/local/lib/python3.1
0/dist-packages (from librosa) (0.56.4)
Requirement already satisfied: soundfile>=0.12.1 in /usr/local/lib/pytho
n3.10/dist-packages (from librosa) (0.12.1)
Requirement already satisfied: pooch<1.7,>=1.0 in /usr/local/lib/python
3.10/dist-packages (from librosa) (1.6.0)
Requirement already satisfied: soxr>=0.3.2 in /usr/local/lib/python3.10/
dist-packages (from librosa) (0.3.5)
Requirement already satisfied: typing-extensions>=4.1.1 in /usr/local/li
b/python3.10/dist-packages (from librosa) (4.7.1)
Requirement already satisfied: lazy-loader>=0.1 in /usr/local/lib/python
3.10/dist-packages (from librosa) (0.3)
Requirement already satisfied: msgpack>=1.0 in /usr/local/lib/python3.1
0/dist-packages (from librosa) (1.0.5)
Requirement already satisfied: llvmlite<0.40,>=0.39.0dev0 in /usr/local/
lib/python3.10/dist-packages (from numba>=0.51.0->librosa) (0.39.1)
Requirement already satisfied: setuptools in /usr/local/lib/python3.10/d
ist-packages (from numba>=0.51.0->librosa) (67.7.2)
Requirement already satisfied: appdirs>=1.3.0 in /usr/local/lib/python3.
10/dist-packages (from pooch<1.7,>=1.0->librosa) (1.4.4)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python
3.10/dist-packages (from pooch<1.7,>=1.0->librosa) (23.1)
Requirement already satisfied: requests>=2.19.0 in /usr/local/lib/python
3.10/dist-packages (from pooch<1.7,>=1.0->librosa) (2.27.1)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/py
thon3.10/dist-packages (from scikit-learn>=0.20.0->librosa) (3.2.0)
Requirement already satisfied: cffi>=1.0 in /usr/local/lib/python3.10/di
st-packages (from soundfile>=0.12.1->librosa) (1.15.1)
Requirement already satisfied: pycparser in /usr/local/lib/python3.10/di
st-packages (from cffi>=1.0->soundfile>=0.12.1->librosa) (2.21)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/p
ython3.10/dist-packages (from requests>=2.19.0->pooch<1.7,>=1.0->libros
a) (1.26.16)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/pyth
on3.10/dist-packages (from requests>=2.19.0->pooch<1.7,>=1.0->librosa)
(2023.7.22)
Requirement already satisfied: charset-normalizer~=2.0.0 in /usr/local/l
ib/python3.10/dist-packages (from requests>=2.19.0->pooch<1.7,>=1.0->lib
rosa) (2.0.12)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.1
0/dist-packages (from requests>=2.19.0->pooch<1.7,>=1.0->librosa) (3.4)
```

In [2]:
```
import os
import numpy as np
import pandas as pd
import librosa
import scipy.signal
import tensorflow as tf
import time
import matplotlib.pyplot as plt
from keras.layers import Dense, Flatten
from keras.models import Sequential
```

In [4]:
```
def process_audio_files(folder_path):
    audio_data=[]
    file_list=os.listdir(folder_path)
    mp3_files=[file for file in file_list if file.endswith('.mp3')]
    for mp3_file in mp3_files:
        file_path=os.path.join(folder_path,mp3_file)
        y,sr=librosa.load(file_path,sr=None)
        audio_data.append((y, sr))
    return audio_data
```

In [6]:
```
folder_path='sample_data//Try'
audio=process_audio_files(folder_path)
```

In [7]:
```
audio
```

Out[7]:
```
[(array([0.        , 0.        , 0.        , ..., 0.02211756, 0.0202147
5,
         0.01168255], dtype=float32),
  48000),
 (array([0.        , 0.        , 0.        , ..., 0.02002131, 0.0230677
7,
         0.01730498], dtype=float32),
  48000),
 (array([0.        , 0.        , 0.        , ..., 0.02088441, 0.0253963
6,
         0.01974409], dtype=float32),
  48000),
 (array([ 0.        ,  0.        ,  0.        , ..., -0.01336266,
         -0.01330735, -0.0091626 ], dtype=float32),
  48000),
 (array([ 0.        ,  0.        ,  0.        , ..., -0.0163647 ,
         -0.01587155, -0.01154031], dtype=float32),
  48000),
 (array([ 0.        ,  0.        ,  0.        , ..., -0.009379  ,
         -0.0107859 , -0.00566604], dtype=float32),
  48000),
 (array([ 0.        ,  0.        ,  0.        , ..., -0.00780205,
         -0.00998528, -0.00769939], dtype=float32),
  48000),
 (array([0.        , 0.        , 0.        , ..., 0.01750566, 0.0216921
1,
         0.01574345], dtype=float32),
  48000),
 (array([0.        , 0.        , 0.        , ..., 0.00611622, 0.0072817
8,
         0.00598932], dtype=float32),
  48000),
 (array([ 0.        ,  0.        ,  0.        , ..., -0.01377062,
         -0.01656381, -0.01348157], dtype=float32),
  48000)]
```

In [8]:
```
folder_path='sample_data//Cry'
audio1=process_audio_files(folder_path)
```

In [9]:
```
audio1
```

Out[9]:
```
[(array([0.        , 0.        , 0.        , ..., 0.04415061, 0.0518660
6,
         0.04076651], dtype=float32),
  48000),
 (array([ 0.        ,  0.        ,  0.        , ..., -0.02186836,
         -0.02531097, -0.01882702], dtype=float32),
  48000),
 (array([ 0.        ,  0.        ,  0.        , ..., -0.00283113,
         -0.00390778, -0.00514695], dtype=float32),
  48000),
 (array([0.        , 0.        , 0.        , ..., 0.00397619, 0.0086906
5,
         0.00809947], dtype=float32),
  48000),
 (array([0.        , 0.        , 0.        , ..., 0.0245386 , 0.0320041
1,
         0.02712047], dtype=float32),
  48000),
 (array([ 0.        ,  0.        ,  0.        , ..., -0.02233866,
         -0.02426209, -0.02025038], dtype=float32),
  48000),
 (array([0.        , 0.        , 0.        , ..., 0.03597236, 0.0366037
3,
         0.02603437], dtype=float32),
  48000),
 (array([ 0.        ,  0.        ,  0.        , ..., -0.0363427 ,
         -0.03576628, -0.02407473], dtype=float32),
  48000),
 (array([0.        , 0.        , 0.        , ..., 0.05528521, 0.0602350
3,
         0.04488189], dtype=float32),
  48000),
 (array([ 0.        ,  0.        ,  0.        , ..., -0.07375773,
         -0.07672676, -0.05462236], dtype=float32),
  48000)]
```

Step-2

In [10]:
```
def extract_stft_features(audio_data,max_frames,max_bins):
    stft_features=[]
    for audio in audio_data:
        y,sr=audio
        _,_,Zxx =scipy.signal.stft(y, fs=sr)
        stft_magnitude=np.abs(Zxx)
        pad_rows=max_frames-stft_magnitude.shape[0]
        pad_cols=max_bins-stft_magnitude.shape[1]
        stft_magnitude=np.pad(stft_magnitude,((0,max(0,pad_rows)),(0,max(
        stft_magnitude=stft_magnitude[:max_frames,:max_bins]
        stft_features.append(stft_magnitude)
    return np.array(stft_features)
```

In [11]:
```
max_frames=100
max_bins=129
```

In [15]:
```
cat=extract_stft_features(audio,max_frames,max_bins)
rat=extract_stft_features(audio1,max_frames,max_bins)
```

In [13]:
```
cat
```

Out[13]:
```
array([[[0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         2.02262122e-02, 5.01466542e-02, 2.20277067e-02],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         2.58404166e-02, 3.26694921e-02, 1.87698919e-02],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         1.53764291e-02, 1.59718879e-02, 9.06452816e-03],
        ...,
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         9.15302607e-08, 5.67605376e-08, 1.00192395e-07],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         4.69978652e-08, 2.88961459e-08, 1.90485032e-08],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         1.68948855e-08, 1.29001565e-08, 2.12374651e-08]],

       [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         1.57098044e-02, 1.61150210e-02, 1.29205838e-03],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         1.07490625e-02, 1.01810182e-02, 6.71091303e-03],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         7.97351543e-03, 3.92317493e-03, 8.97438265e-03],
        ...,
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         2.01283434e-08, 3.84894037e-08, 1.95681373e-08],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         2.61782884e-09, 1.63584311e-08, 3.95806232e-09],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         5.19900167e-09, 1.08090248e-08, 8.26146351e-09]],

       [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         1.48129249e-02, 8.71927128e-04, 1.96195096e-02],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         1.26565043e-02, 2.80642812e-03, 1.70633551e-02],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         6.53536944e-03, 9.10742849e-04, 8.99430551e-03],
        ...,
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         4.01692049e-08, 5.70189407e-08, 3.69337343e-08],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         7.78952458e-09, 2.39896725e-08, 2.92539903e-09],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         5.82485704e-09, 1.47518842e-08, 2.09306439e-08]],

       ...,

       [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         2.23149192e-02, 7.57935969e-03, 2.58157607e-02],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         1.57520566e-02, 7.39929918e-03, 1.39469607e-02],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         4.95382911e-03, 2.19692825e-03, 4.46815335e-04],
        ...,
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         2.74305592e-08, 6.17155678e-08, 5.11106109e-08],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         2.61571431e-09, 2.37847408e-09, 1.04960751e-08],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         4.95803620e-09, 1.90541858e-08, 1.01211075e-08]],
```

```
[[0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
  3.50751318e-02, 2.19649579e-02, 6.14966359e-03],
 [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
  6.68525398e-02, 4.17798162e-02, 5.13756536e-02],
 [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
  8.61585215e-02, 4.36202483e-03, 6.90896288e-02],
 ...,
 [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
  8.55800195e-08, 1.48482357e-07, 1.27299501e-07],
 [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
  1.31705733e-07, 1.66422254e-07, 6.42566320e-08],
 [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
  6.95767071e-08, 2.43153302e-08, 3.38495454e-08]],

[[0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
  5.93234487e-02, 5.11498004e-02, 1.61629058e-02],
 [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
  3.40234004e-02, 2.79506147e-02, 5.32587338e-03],
 [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
  4.67488449e-03, 3.62476707e-03, 3.20405397e-03],
 ...,
 [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
  5.27295576e-08, 7.06176166e-08, 8.41809324e-08],
 [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
  8.38824121e-09, 9.52729007e-09, 4.97720665e-09],
 [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
  7.09532078e-09, 8.97911256e-09, 3.63916830e-09]]], dtype=float32)
```

In [16]:
```
rat
```

Out[16]:
```
array([[[0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         6.76671863e-02, 3.75292711e-02, 2.38517374e-02],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         3.97355147e-02, 2.19617300e-02, 3.04844771e-02],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         9.68677085e-03, 3.49362986e-03, 1.09125739e-02],
        ...,
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         6.60876083e-08, 1.32129330e-07, 1.93034012e-07],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         1.43331178e-08, 1.97977688e-08, 8.26928925e-09],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         3.61376351e-09, 4.81006834e-09, 1.51211232e-09]],

       [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         1.48089146e-02, 2.29577441e-02, 2.44681407e-02],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         1.45090669e-02, 1.72619764e-02, 1.79888830e-02],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         4.74095065e-03, 3.94128589e-03, 5.41834394e-03],
        ...,
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         2.51591583e-08, 5.50407329e-08, 7.20324920e-08],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         1.10143556e-08, 1.63978697e-08, 9.94420901e-09],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         8.32526137e-09, 8.01306399e-09, 6.04499606e-09]],

       [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         1.00214750e-01, 1.28310218e-01, 5.79744056e-02],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         5.57391196e-02, 8.36497322e-02, 6.35167584e-02],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         2.80091958e-03, 1.34705752e-02, 3.39664295e-02],
        ...,
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         3.94064500e-07, 2.53791796e-07, 1.97730785e-07],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         3.77265899e-08, 5.92475544e-08, 3.00246832e-08],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         1.76872046e-08, 2.96750766e-08, 7.77572673e-09]],

       ...,

       [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         7.01909289e-02, 3.30062881e-02, 4.37115654e-02],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         7.30663538e-02, 5.46729490e-02, 1.71476658e-02],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         5.39708696e-02, 3.93499881e-02, 3.36363502e-02],
        ...,
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         9.78753079e-08, 1.02546430e-07, 1.54251751e-07],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         9.15004179e-08, 9.08567159e-08, 9.38103994e-08],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         8.42596108e-08, 1.00980920e-07, 6.33511377e-08]],
```

```
       [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         2.67798956e-02, 9.63799283e-03, 3.53631563e-02],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         3.47845815e-02, 1.22454362e-02, 2.44195256e-02],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         2.05402337e-02, 8.03593360e-03, 7.15873763e-03],
        ...,
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         2.57046594e-07, 1.13859244e-07, 2.19989545e-07],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         1.93946477e-08, 3.72474673e-08, 5.78598041e-08],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         1.31067086e-08, 7.11485582e-09, 4.85730673e-08]],

       [[0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         1.00519527e-02, 4.89512607e-02, 2.25642491e-02],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         8.47600773e-03, 2.82154344e-02, 1.78734474e-02],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         3.16314981e-03, 1.41192169e-03, 7.55664520e-03],
        ...,
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         1.35946337e-07, 6.84414871e-08, 5.16779082e-08],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         8.79375062e-09, 1.63977294e-08, 5.38571587e-09],
        [0.00000000e+00, 0.00000000e+00, 0.00000000e+00, ...,
         9.45658840e-09, 2.23728609e-08, 2.51569983e-08]]], dtype=float3
2)
```

Step-3

In [17]:
```
class_labels=['cat','rat']
num_classes=len(class_labels)
```

In [18]:
```
all_stft_features=cat+rat
all_labels=[0]*len(cat)+[1]*len(rat)
```

In [19]:
```
indices=np.arange(len(all_stft_features))
np.random.shuffle(indices)
all_stft_features=[all_stft_features[i] for i in indices]
all_labels=[all_labels[i] for i in indices]
```

In [20]:
```
split_ratio=0.75
split_index=int(len(all_stft_features) * split_ratio)
x_train,y_train=all_stft_features[:split_index],all_labels[:split_index]
x_test,y_test=all_stft_features[split_index:],all_labels[split_index:]
```

In [21]:
```
x_train=np.array(x_train).astype(np.float32)
x_test=np.array(x_test).astype(np.float32)
y_train=np.array(y_train).astype(np.int32)
y_test=np.array(y_test).astype(np.int32)
```

In [23]:
```
x_train.shape
```

Out[23]: (7, 100, 129)

In [24]:
```
x_test.shape
```

Out[24]: (3, 100, 129)

In [25]:
```
y_train.shape
```

Out[25]: (7,)

In [26]:
```
y_test.shape
```

Out[26]: (3,)

Step-4

In [27]:
```
def create_model(input_shape,num_classes):
    model=Sequential()
    model.add(Flatten(input_shape=input_shape))
    model.add(Dense(128, activation='relu'))
    model.add(Dense(num_classes,activation='softmax'))
    return model
```

In [28]:
```
input_shape=x_train[0].shape
model=create_model(input_shape,num_classes)
model.compile(loss='sparse_categorical_crossentropy',optimizer='adam',met
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 flatten (Flatten)           (None, 12900)             0

 dense (Dense)               (None, 128)               1651328

 dense_1 (Dense)             (None, 2)                 258

=================================================================
Total params: 1,651,586
Trainable params: 1,651,586
Non-trainable params: 0
_________________________________________________________________
```

In [30]:
```
history=model.fit(x_train,y_train,batch_size=32,epochs=20,validation_spli
```

```
Epoch 1/20
1/1 [==============================] - 0s 139ms/step - loss: 0.3637 - ac
curacy: 1.0000 - val_loss: 0.2796 - val_accuracy: 1.0000
Epoch 2/20
1/1 [==============================] - 0s 90ms/step - loss: 0.3398 - acc
uracy: 1.0000 - val_loss: 0.2531 - val_accuracy: 1.0000
Epoch 3/20
1/1 [==============================] - 0s 99ms/step - loss: 0.3173 - acc
uracy: 1.0000 - val_loss: 0.2286 - val_accuracy: 1.0000
Epoch 4/20
1/1 [==============================] - 0s 101ms/step - loss: 0.2963 - ac
curacy: 1.0000 - val_loss: 0.2061 - val_accuracy: 1.0000
Epoch 5/20
1/1 [==============================] - 0s 73ms/step - loss: 0.2767 - acc
uracy: 1.0000 - val_loss: 0.1854 - val_accuracy: 1.0000
Epoch 6/20
1/1 [==============================] - 0s 85ms/step - loss: 0.2584 - acc
uracy: 1.0000 - val_loss: 0.1665 - val_accuracy: 1.0000
Epoch 7/20
1/1 [==============================] - 0s 98ms/step - loss: 0.2413 - acc
uracy: 1.0000 - val_loss: 0.1493 - val_accuracy: 1.0000
Epoch 8/20
1/1 [==============================] - 0s 85ms/step - loss: 0.2254 - acc
uracy: 1.0000 - val_loss: 0.1337 - val_accuracy: 1.0000
Epoch 9/20
1/1 [==============================] - 0s 97ms/step - loss: 0.2106 - acc
uracy: 1.0000 - val_loss: 0.1196 - val_accuracy: 1.0000
Epoch 10/20
1/1 [==============================] - 0s 67ms/step - loss: 0.1969 - acc
uracy: 1.0000 - val_loss: 0.1069 - val_accuracy: 1.0000
Epoch 11/20
1/1 [==============================] - 0s 60ms/step - loss: 0.1841 - acc
uracy: 1.0000 - val_loss: 0.0955 - val_accuracy: 1.0000
Epoch 12/20
1/1 [==============================] - 0s 55ms/step - loss: 0.1721 - acc
uracy: 1.0000 - val_loss: 0.0852 - val_accuracy: 1.0000
Epoch 13/20
1/1 [==============================] - 0s 51ms/step - loss: 0.1611 - acc
uracy: 1.0000 - val_loss: 0.0761 - val_accuracy: 1.0000
Epoch 14/20
1/1 [==============================] - 0s 71ms/step - loss: 0.1508 - acc
uracy: 1.0000 - val_loss: 0.0679 - val_accuracy: 1.0000
Epoch 15/20
1/1 [==============================] - 0s 56ms/step - loss: 0.1412 - acc
uracy: 1.0000 - val_loss: 0.0607 - val_accuracy: 1.0000
Epoch 16/20
1/1 [==============================] - 0s 56ms/step - loss: 0.1323 - acc
uracy: 1.0000 - val_loss: 0.0542 - val_accuracy: 1.0000
Epoch 17/20
1/1 [==============================] - 0s 57ms/step - loss: 0.1241 - acc
uracy: 1.0000 - val_loss: 0.0484 - val_accuracy: 1.0000
Epoch 18/20
1/1 [==============================] - 0s 55ms/step - loss: 0.1164 - acc
uracy: 1.0000 - val_loss: 0.0433 - val_accuracy: 1.0000
Epoch 19/20
1/1 [==============================] - 0s 73ms/step - loss: 0.1093 - acc
uracy: 1.0000 - val_loss: 0.0388 - val_accuracy: 1.0000
```

```
Epoch 20/20
1/1 [==============================] - 0s 54ms/step - loss: 0.1027 - acc
uracy: 1.0000 - val_loss: 0.0347 - val_accuracy: 1.0000
```

In [31]:
```
test_loss,test_accuracy=model.evaluate(x_test,y_test)
print("Test Loss:",test_loss)
print("Test Accuracy:",test_accuracy)
```

```
1/1 [==============================] - 0s 30ms/step - loss: 0.0217 - acc
uracy: 1.0000
Test Loss: 0.02172904647886753
Test Accuracy: 1.0
```

Step-5

In [32]:
```
def create_model_with_config(input_shape,num_classes,num_nodes,num_layers
    model = Sequential()
    model.add(Flatten(input_shape=input_shape))
    for _ in range(num_layers):
        model.add(Dense(num_nodes, activation='relu'))
    model.add(Dense(num_classes, activation='softmax'))
    return model
```

In [33]:
```
nodes_list=[8, 16, 32, 64, 128]
layers_list=[2, 3, 4]
```

In [34]:
```
num_params_list=[]
train_accuracy_list=[]
test_accuracy_list=[]
running_time_list=[]

for num_nodes in nodes_list:
    for num_layers in layers_list:
        model=create_model_with_config(input_shape,num_classes,num_nodes,
        model.compile(loss='sparse_categorical_crossentropy',optimizer='a
        start_time=time.time()
        history=model.fit(x_train,y_train,batch_size=32,epochs=20,validat
        end_time=time.time()
        running_time=end_time-start_time
        _,train_accuracy=model.evaluate(x_train,y_train,verbose=0)
        _,test_accuracy=model.evaluate(x_test, y_test,verbose=0)
        num_params=model.count_params()

        num_params_list.append(num_params)
        train_accuracy_list.append(train_accuracy)
        test_accuracy_list.append(test_accuracy)
        running_time_list.append(running_time)
```

In [35]:

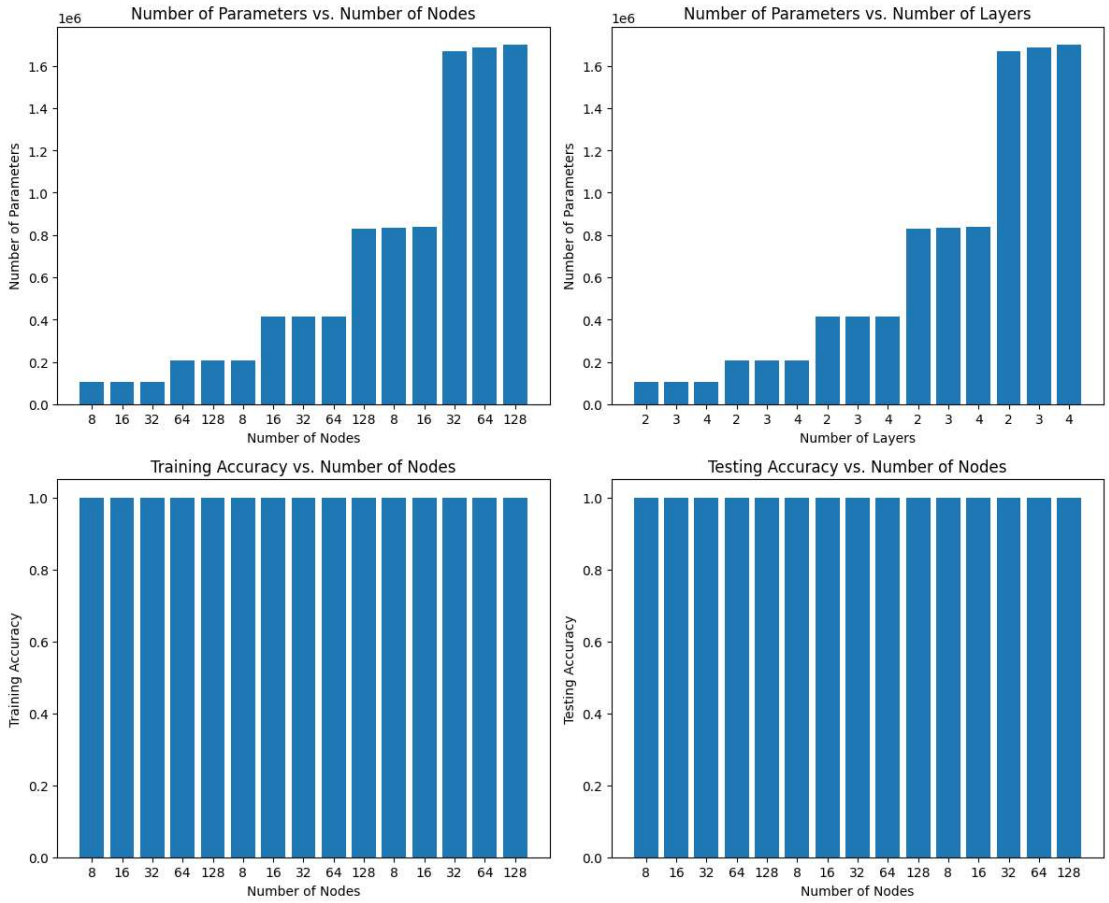
```
fig, axes = plt.subplots(2, 2, figsize=(12, 10))
axes[0, 0].bar(range(len(num_params_list)), num_params_list)
axes[0, 0].set_xticks(range(len(num_params_list)))
axes[0, 0].set_xticklabels(nodes_list * len(layers_list))
axes[0, 0].set_xlabel('Number of Nodes')
axes[0, 0].set_ylabel('Number of Parameters')
axes[0, 0].set_title('Number of Parameters vs. Number of Nodes')

axes[0, 1].bar(range(len(num_params_list)), num_params_list)
axes[0, 1].set_xticks(range(len(num_params_list)))
axes[0, 1].set_xticklabels(layers_list * len(nodes_list))
axes[0, 1].set_xlabel('Number of Layers')
axes[0, 1].set_ylabel('Number of Parameters')
axes[0, 1].set_title('Number of Parameters vs. Number of Layers')

axes[1, 0].bar(range(len(train_accuracy_list)), train_accuracy_list)
axes[1, 0].set_xticks(range(len(train_accuracy_list)))
axes[1, 0].set_xticklabels(nodes_list * len(layers_list))
axes[1, 0].set_xlabel('Number of Nodes')
axes[1, 0].set_ylabel('Training Accuracy')
axes[1, 0].set_title('Training Accuracy vs. Number of Nodes')

axes[1, 1].bar(range(len(test_accuracy_list)), test_accuracy_list)
axes[1, 1].set_xticks(range(len(test_accuracy_list)))
axes[1, 1].set_xticklabels(nodes_list * len(layers_list))
axes[1, 1].set_xlabel('Number of Nodes')
axes[1, 1].set_ylabel('Testing Accuracy')
axes[1, 1].set_title('Testing Accuracy vs. Number of Nodes')

plt.tight_layout()
plt.show()
```


Number of Parameters vs. Number of Nodes
1e6
Number of Parameters
Number of Nodes
Number of Parameters vs. Number of Layers
Number of Layers
Training Accuracy vs. Number of Nodes
Training Accuracy
Testing Accuracy vs. Number of Nodes
Testing Accuracy

In [ ]: