

Implementation of HPC for Music Genres Classification

for music recommendation systems

DSC 520-01: High Perform Scientific Compute

NIKHILESH NARKHEDE (02196141)

Problem Statement

Music genres: Build a DL model to classify music into the top 10 genres.

Audio Data  Time series data.

If we go with traditional time series architecture like RNN, the LSTM model turns out to be bulky and faces problems like vanishing gradient.

But we can convert Audio data in image format using a spectrogram with a library like Librosa.

Its representation of the spectrum of frequency in sound recording as they vary over time.

Dark Area: Low-intensity frequency.

Orange and yellow Areas: High intensities frequency.

<https://academo.org/demos/spectrum-analyzer/>

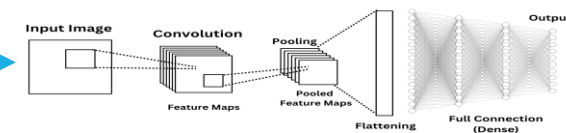


Architecture



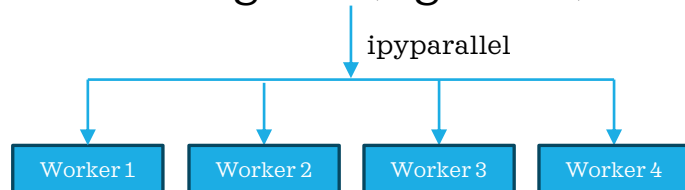
I have implemented HPC

- Preprocessing Data: using Librosa, ipyparallel.
- Deep Learning model training: Multithreading in TensorFlow.



Preprocessing Data :

- sub-genre(e.g. Rock) 100 songs



25 songs for each worker.

Deep Learning model training:

- OpenMP
- Inter-operation
- Intra-operation



ipyparallel for Distributed Computing

What is ipyparallel?

- A Python library that enables parallel task execution using multiple workers, ideal for distributed computing.

Key Components:

- **Controller:** Distributes tasks and maintains the state.
- **Engines:** Execute tasks independently.

Example Use Case:

- Parallel processing of audio files into spectrogram images to reduce computation time.

Benefits:

- Improves efficiency, scales to multi-core systems, and integrates seamlessly with Python and Jupyter.

Deep Learning Model Training Optimization

OpenMP

- OpenMP optimizes CPU-based computations by distributing work across multiple cores.
- Usage in this Model:
- Convolution layers (Conv2D) and matrix operations in fully connected layers (Dense) are parallelized.
- OpenMP ensures that operations like matrix multiplications for the filters in the convolution layers and the weight updates in the dense layers are executed across multiple CPU threads.

Inter-Operation Parallelism

- Manages parallel execution across different operations in the computational graph.
- If two Conv2D layers are computationally independent (e.g., during forward propagation), TensorFlow can process them concurrently.
- Pooling layers and dropout calculations can also be executed in parallel with certain independent layers.

Intra-Operation Parallelism

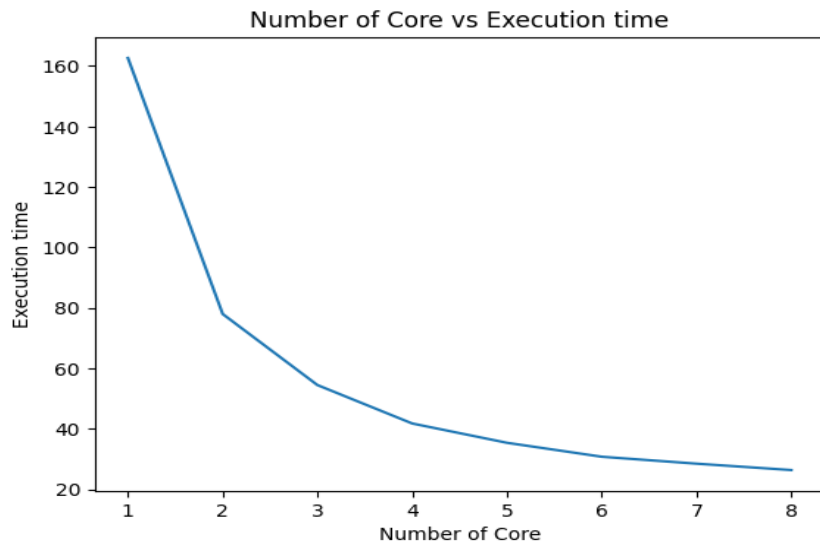
- Optimizes parallelism within a single operation.
- Each Conv2D operation (a matrix multiplication between input data and convolution filters) is broken into smaller tasks.
- Pooling and activation computations (like ReLU) for large feature maps are divided among threads.

Key Benefits

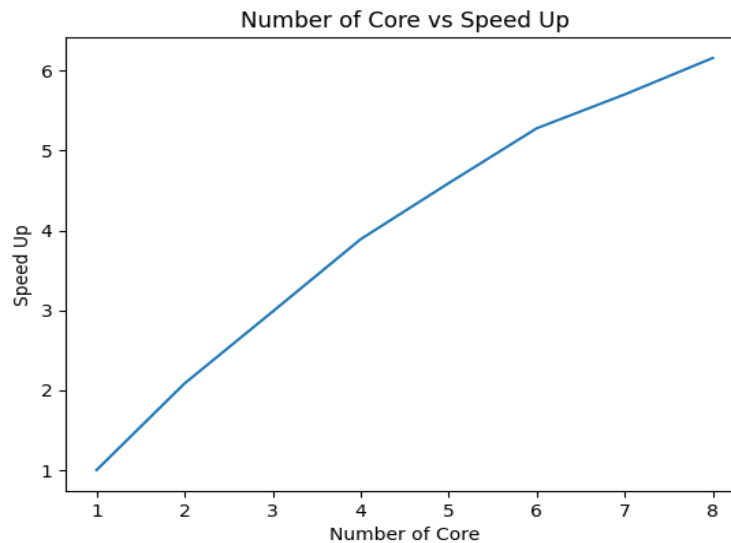
- Improved efficiency and reduced execution time.
- Scalability across multi-core systems.
- Better utilization of computational resources.

Result (Preprocessing)

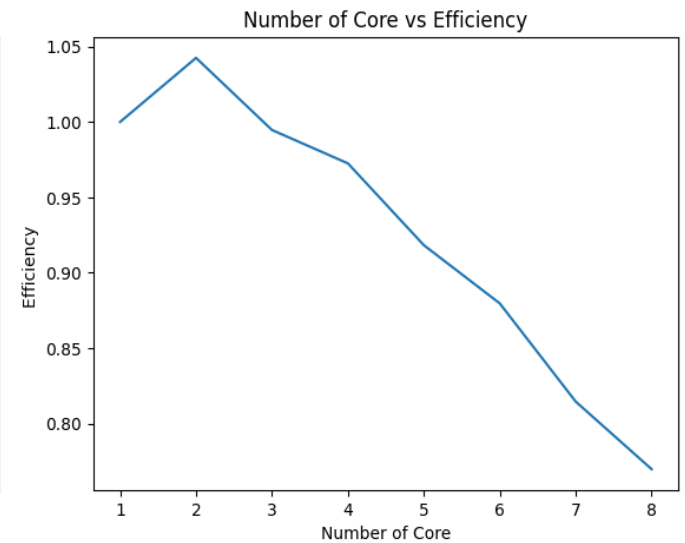
Number of core vs Execution time



Number of core vs Speed-up

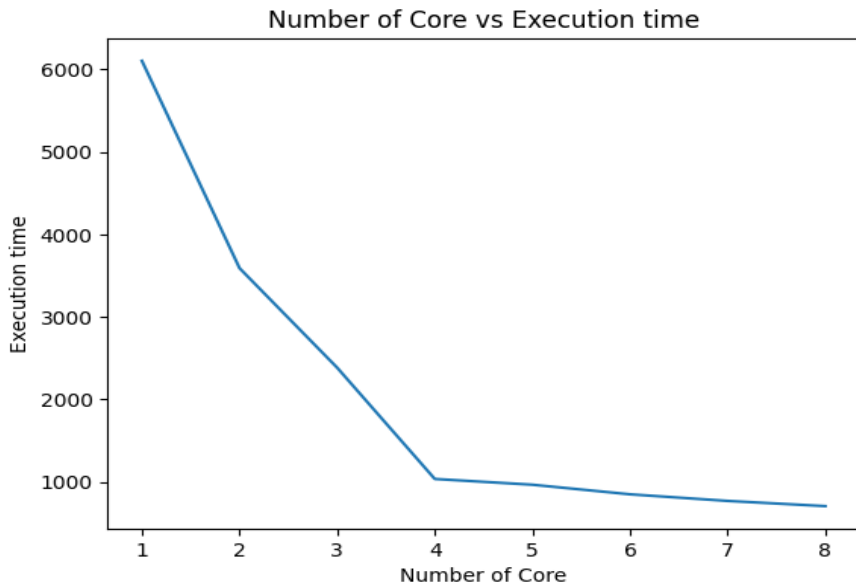


Number of core vs Efficiency

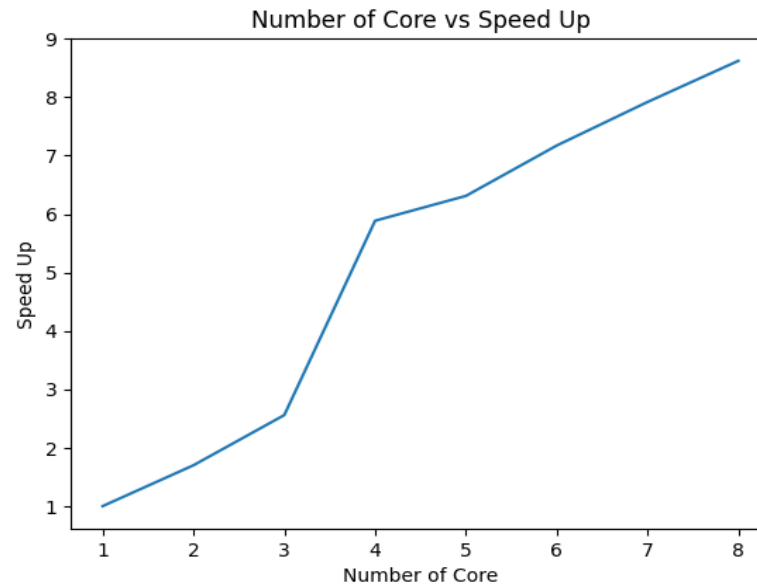


Result (DL Model Training)

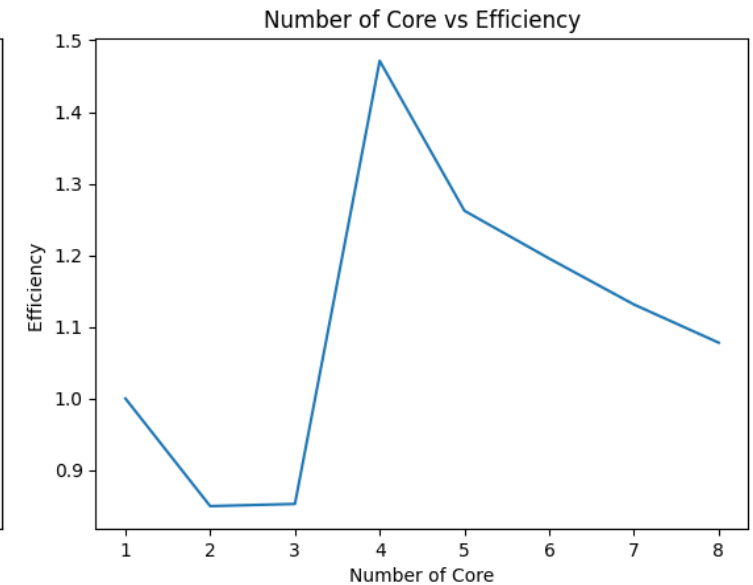
Number of core vs Execution time



Number of core vs Speed-up



Number of core vs Efficiency



Conclusion

Preprocessing Data:

- 2 workers are most preferable in this context if efficiency is given more weight
- 4 workers are most preferable in this context if execution time is given more weight

Deep Learning Model Training:

- 4 Multithreading (Worker) is most preferable in this context in terms of both execution time and efficiency.

Appendix

Import Lib

```
In [1]: import os
from glob import glob
import librosa
import librosa.display
import numpy as np
import matplotlib.pyplot as plt
import ipyparallel as ipp
import time
from warnings import filterwarnings
filterwarnings('ignore')
```

Function to create a spectrogram from an audio file

```
In [2]: def create_spectrogram(filename, name, store_path):
plt.interactive(False)
clip, sample_rate = librosa.load(filename, sr=None)
fig = plt.figure(figsize=[0.72, 0.72])
ax = fig.add_subplot(111)
ax.axes.get_xaxis().set_visible(False)
ax.axes.get_yaxis().set_visible(False)
ax.set_frame_on(False)
S = librosa.feature.melspectrogram(y=clip, sr=sample_rate)
librosa.display.specshow(librosa.power_to_db(S, ref=np.max))
filename = os.path.join(store_path, name + '.jpg')
plt.savefig(filename, dpi=400, bbox_inches='tight', pad_inches=0)
plt.close()
fig.clf()
plt.close(fig)
plt.close('all')
del filename, name, clip, sample_rate, fig, ax, S
```

Helper function to process a single audio file

```
In [3]: def process_audio_file(file, folder_path):
name = os.path.basename(file)
create_spectrogram(file, name, folder_path)
```

Convert Audio files to spectrogram JPG files with ipyparallel

```
In [4]: def Convert_Audio_File_to_jpg_file(filename, store_filepath, number_of_worker):
"""
Convert audio files to spectrogram JPG files using ipyparallel.

Parameters:
    filename (str): Path to the main directory containing audio files and number of worker.
"""
# Set up ipyparallel cluster
Execution_time = 0
rc = ipp.Cluster(n=number_of_worker).start_and_connect_sync() # Start and connect cluster with 4 workers
dview = rc[:] # Create a direct view to all workers

dview.push({"create_spectrogram": create_spectrogram})
dview.push({"process_audio_file": process_audio_file})
dview.execute("import os")
dview.execute("import librosa")
dview.execute("import librosa.display")
dview.execute("import numpy as np")
dview.execute("import matplotlib.pyplot as plt")
dview.execute("import time")

start_time = time.time()
# Make a dictionary for given class and their file path names
file_list = glob(os.path.join(filename, "*"))
file_dic = {}
for file in file_list:
    all_files = []
```

```

for root, dirs, files in os.walk(file):
    for file_ in files:
        # Join the root directory with the file name to get the full path
        all_files.append(os.path.join(root, file_))
file_dic[file] = all_files

# Create file directories to store the converted audio files into JPG
file_path = []
for folder in file_dic.keys():
    folder_rot = os.path.join(store_filepath, os.path.basename(folder))
    file_path.append(folder_rot)
    os.makedirs(folder_rot, exist_ok=True)

# Delegate work to workers using dview.map_sync()
for i, folder in enumerate(file_dic.keys()):
    music_files = file_dic[folder]
    folder_path = file_path[i]
    # Map the processing function across music files with ipyparallel
    dview.map_sync(lambda file: process_audio_file(file, folder_path), music_files)
end_time = time.time()
elapsed_time = end_time - start_time
Execution_time = elapsed_time
print(f"Execution time: {elapsed_time:.2f} seconds")
# Shut down the cluster after processing
rc.shutdown()

return Execution_time

```

Convert Audio File to jpg file

```

In [5]: Audio_Data_Path = r"C:\Users\nikhi\OneDrive\Desktop\Final Project\DATA\genres"
Spectro_jpg_Path = r"C:\Users\nikhi\OneDrive\Desktop\Final Project\DATA\Convert_Audio_File_to_jpg_file"
Convert_Audio_File_to_jpg_file(Audio_Data_Path, Spectro_jpg_Path, number_of_worker = 4)

```

```

Starting 4 engines with <class 'ipyparallel.cluster.launcher.LocalEngineSetLauncher'>
 0%|          | 0/4 [00:00<?, ?engine/s]
Execution time: 43.02 seconds

```

Out[5]: 43.01757502555847

```

engine set stopped 1733365864: {'engines': {'2': {'exit_code': 0, 'pid': 20512, 'identifier': '2'}, '0': {'exit_code': 0, 'pid': 21332, 'identifier': '0'}, '3': {'exit_code': 0, 'pid': 10848, 'identifier': '3'}, '1': {'exit_code': 0, 'pid': 18488, 'identifier': '1'}}, 'exit_code': 0}

```

Experiment with Core

```

In [6]: Execution_time = []
Cores = range(1,9)
for number_of_worker in Cores:
    Execution_time.append(Convert_Audio_File_to_jpg_file(Audio_Data_Path, Spectro_jpg_Path, number_of_worker = number_of_worker))

```

```

Starting 1 engines with <class 'ipyparallel.cluster.launcher.LocalEngineSetLauncher'>
 0%|          | 0/1 [00:00<?, ?engine/s]
Execution time: 162.66 seconds
engine set stopped 1733365915: {'engines': {'0': {'exit_code': 0, 'pid': 14416, 'identifier': '0'}}, 'exit_code': 0}
Starting 2 engines with <class 'ipyparallel.cluster.launcher.LocalEngineSetLauncher'>
 0%|          | 0/2 [00:00<?, ?engine/s]
Execution time: 78.02 seconds
Failed to remove C:\Users\nikhi\ipython\profile_default\log\ipengine-1733366084-vstu-1733366085-1.log: [WinError 5] Access is denied: 'C:\\Users\\nikhi\\ipython\\profile_default\\log\\ipengine-1733366084-vstu-1733366085-1.log'
engine set stopped 1733366085: {'engines': {'0': {'exit_code': 0, 'pid': 21460, 'identifier': '0'}, '1': {'exit_code': 0, 'pid': 27340, 'identifier': '1'}}, 'exit_code': 0}
Starting 3 engines with <class 'ipyparallel.cluster.launcher.LocalEngineSetLauncher'>
 0%|          | 0/3 [00:00<?, ?engine/s]
Execution time: 54.51 seconds
engine set stopped 1733366171: {'engines': {'2': {'exit_code': 0, 'pid': 25728, 'identifier': '2'}, '1': {'exit_code': 0, 'pid': 20336, 'identifier': '1'}, '0': {'exit_code': 0, 'pid': 25376, 'identifier': '0'}}, 'exit_code': 0}
Starting 4 engines with <class 'ipyparallel.cluster.launcher.LocalEngineSetLauncher'>
 0%|          | 0/4 [00:00<?, ?engine/s]
Execution time: 41.81 seconds
engine set stopped 1733366233: {'engines': {'0': {'exit_code': 0, 'pid': 18900, 'identifier': '0'}, '2': {'exit_code': 0, 'pid': 20572, 'identifier': '2'}, '3': {'exit_code': 0, 'pid': 22864, 'identifier': '3'}, '1': {'exit_code': 0, 'pid': 23844, 'identifier': '1'}}, 'exit_code': 0}
Starting 5 engines with <class 'ipyparallel.cluster.launcher.LocalEngineSetLauncher'>
 0%|          | 0/5 [00:00<?, ?engine/s]

```

```

Execution time: 35.43 seconds
engine set stopped 1733366282: {'engines': {'3': {'exit_code': 0, 'pid': 25924, 'identifier': '3'}, '0': {'exit_code': 0, 'pid': 24580, 'identifier': '0'}, '2': {'exit_code': 0, 'pid': 27620, 'identifier': '2'}, '1': {'exit_code': 0, 'pid': 21080, 'identifier': '1'}, '4': {'exit_code': 0, 'pid': 19184, 'identifier': '4'}}}, 'exit_code': 0}
Starting 6 engines with <class 'ipyparallel.cluster.launcher.LocalEngineSetLauncher'>
 0%|          | 0/6 [00:00<?, ?engine/s]
Execution time: 30.82 seconds
engine set stopped 1733366326: {'engines': {'2': {'exit_code': 0, 'pid': 20140, 'identifier': '2'}, '0': {'exit_code': 0, 'pid': 23164, 'identifier': '0'}, '1': {'exit_code': 0, 'pid': 11856, 'identifier': '1'}, '5': {'exit_code': 0, 'pid': 26344, 'identifier': '5'}, '4': {'exit_code': 0, 'pid': 2152, 'identifier': '4'}, '3': {'exit_code': 0, 'pid': 27152, 'identifier': '3'}}}, 'exit_code': 0}
Starting 7 engines with <class 'ipyparallel.cluster.launcher.LocalEngineSetLauncher'>
 0%|          | 0/7 [00:00<?, ?engine/s]
Execution time: 28.52 seconds
Failed to remove C:\Users\nikhi\ipython\profile_default\log\ipengine-1733366363-hjat-1733366364-2.log: [WinError 2] The system cannot find the file specified: 'C:\Users\nikhi\ipython\profile_default\log\ipengine-1733366363-hjat-1733366364-2.log'
Failed to remove C:\Users\nikhi\ipython\profile_default\log\ipengine-1733366363-hjat-1733366364-3.log: [WinError 32] The process cannot access the file because it is being used by another process: 'C:\Users\nikhi\ipython\profile_default\log\ipengine-1733366363-hjat-1733366364-3.log'
engine set stopped 1733366364: {'engines': {'1': {'exit_code': 0, 'pid': 23124, 'identifier': '1'}, '2': {'exit_code': 0, 'pid': 23284, 'identifier': '2'}, '3': {'exit_code': 0, 'pid': 10412, 'identifier': '3'}, '4': {'exit_code': 0, 'pid': 11464, 'identifier': '4'}, '5': {'exit_code': 0, 'pid': 1996, 'identifier': '5'}, '0': {'exit_code': 0, 'pid': 13308, 'identifier': '0'}, '6': {'exit_code': 0, 'pid': 26908, 'identifier': '6'}}}, 'exit_code': 0}
Starting 8 engines with <class 'ipyparallel.cluster.launcher.LocalEngineSetLauncher'>
 0%|          | 0/8 [00:00<?, ?engine/s]
Execution time: 26.41 seconds
Failed to remove C:\Users\nikhi\ipython\profile_default\log\ipengine-1733366400-7yh5-1733366401-1.log: [WinError 2] The system cannot find the file specified: 'C:\Users\nikhi\ipython\profile_default\log\ipengine-1733366400-7yh5-1733366401-1.log'
Failed to remove C:\Users\nikhi\ipython\profile_default\log\ipengine-1733366400-7yh5-1733366401-6.log: [WinError 32] The process cannot access the file because it is being used by another process: 'C:\Users\nikhi\ipython\profile_default\log\ipengine-1733366400-7yh5-1733366401-6.log'
engine set stopped 1733366401: {'engines': {'0': {'exit_code': 0, 'pid': 24628, 'identifier': '0'}, '1': {'exit_code': 0, 'pid': 14832, 'identifier': '1'}, '7': {'exit_code': 0, 'pid': 15012, 'identifier': '7'}, '5': {'exit_code': 0, 'pid': 27244, 'identifier': '5'}, '3': {'exit_code': 0, 'pid': 15744, 'identifier': '3'}, '6': {'exit_code': 0, 'pid': 11516, 'identifier': '6'}, '4': {'exit_code': 0, 'pid': 4368, 'identifier': '4'}, '2': {'exit_code': 0, 'pid': 18236, 'identifier': '2'}}}, 'exit_code': 0}

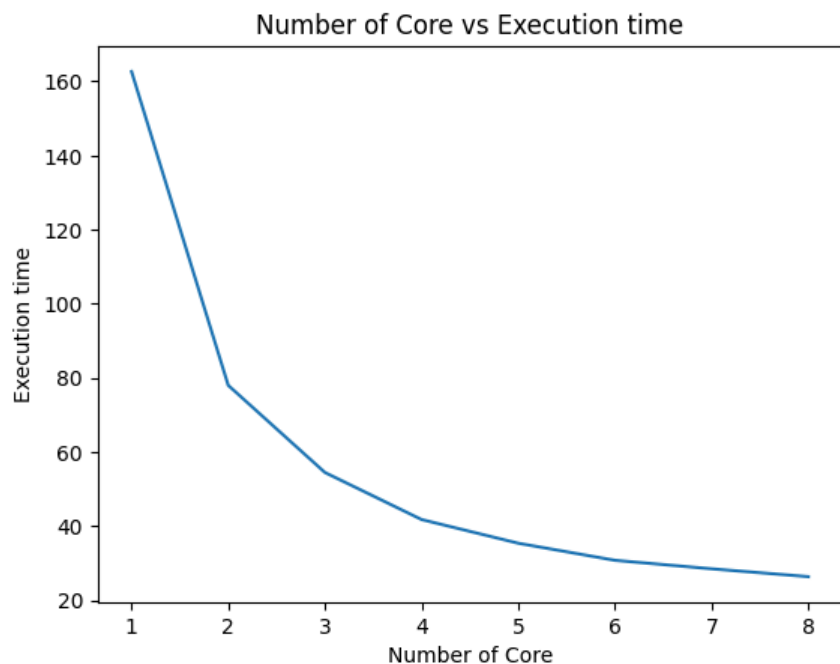
```

Number of Core vs Execution time

```

In [7]: plt.plot(Cores, Execution_time)
plt.xlabel('Number of Core')
plt.ylabel('Execution time')
plt.title('Number of Core vs Execution time')
plt.show()

```



Number of Core vs Speed Up

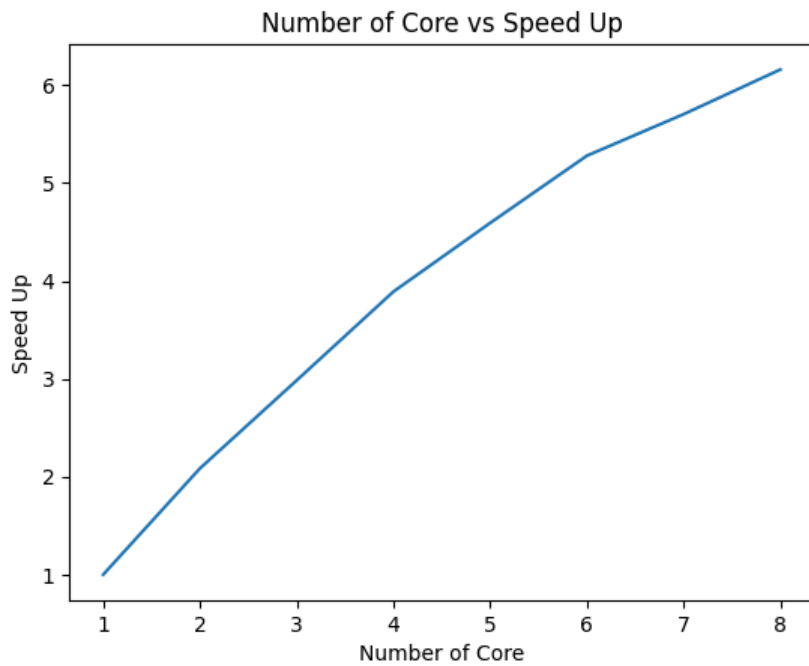
```

In [8]: speed_up = []
for i in Cores:
    speed_up.append(Execution_time[0] / Execution_time[i - 1])

plt.plot(Cores, speed_up)
plt.xlabel('Number of Core')
plt.ylabel('Speed Up')

```

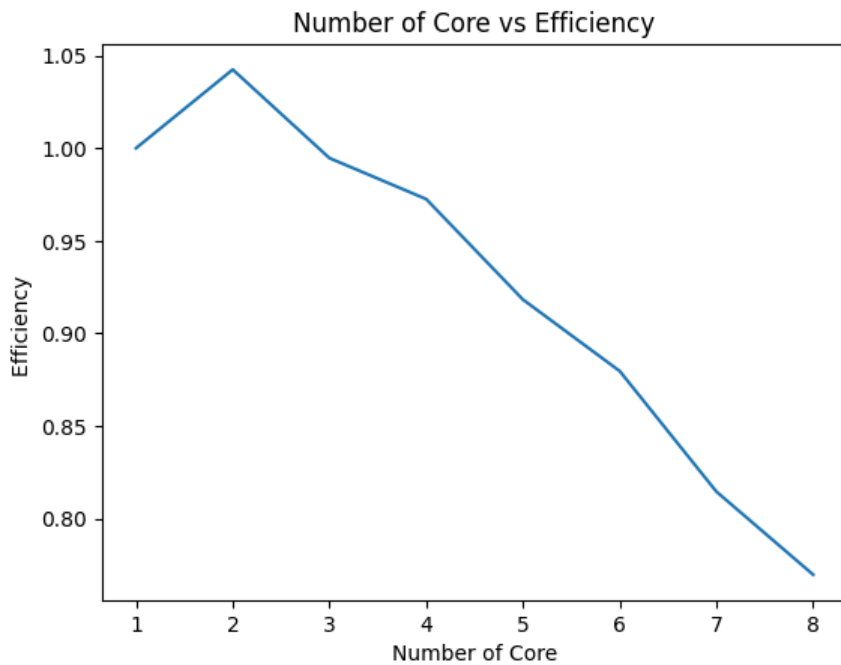
```
plt.title('Number of Core vs Speed Up')
plt.show()
```



Number of Core vs Efficiency

```
In [9]: Efficiency = []
for num_of_core in Cores:
    Efficiency.append(speed_up[num_of_core - 1] / num_of_core)

plt.plot(Cores, Efficiency)
plt.xlabel('Number of Core')
plt.ylabel('Efficiency ')
plt.title('Number of Core vs Efficiency')
plt.show()
```



```
In [ ]:
```

```
In [1]: import matplotlib.pyplot as plt
import pandas as pd
```

```
In [2]: data = pd.read_csv(r"C:\Users\nikhi\OneDrive\Desktop\Final Project\DEEP LEARNING WITH HPSC\core_data.txt")
```

```
In [3]: data.sort_values(by = 'Number of Cores', inplace = True, ignore_index= True)
```

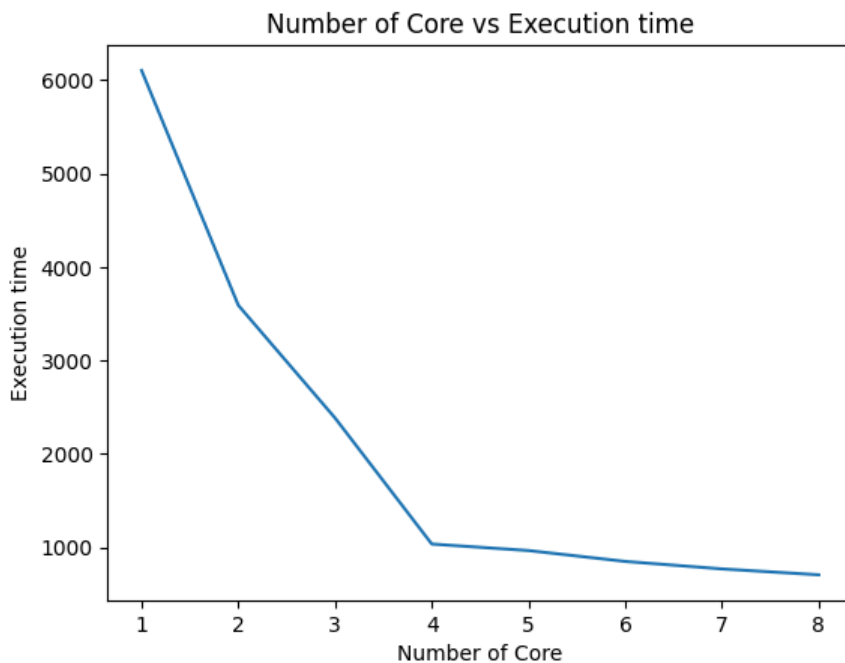
```
In [4]: data
```

```
Out[4]:
```

	Number of Cores	Elapsed Time
0	1	6102.521532
1	2	3590.608773
2	3	2385.136909
3	4	1036.823782
4	5	967.008510
5	6	850.766087
6	7	770.620165
7	8	707.659396

Number of Core vs Execution time

```
In [5]: plt.plot(data['Number of Cores'], data['Elapsed Time'])
plt.xlabel('Number of Core')
plt.ylabel('Execution time')
plt.title('Number of Core vs Execution time')
plt.show()
```



Number of Core vs Speed Up

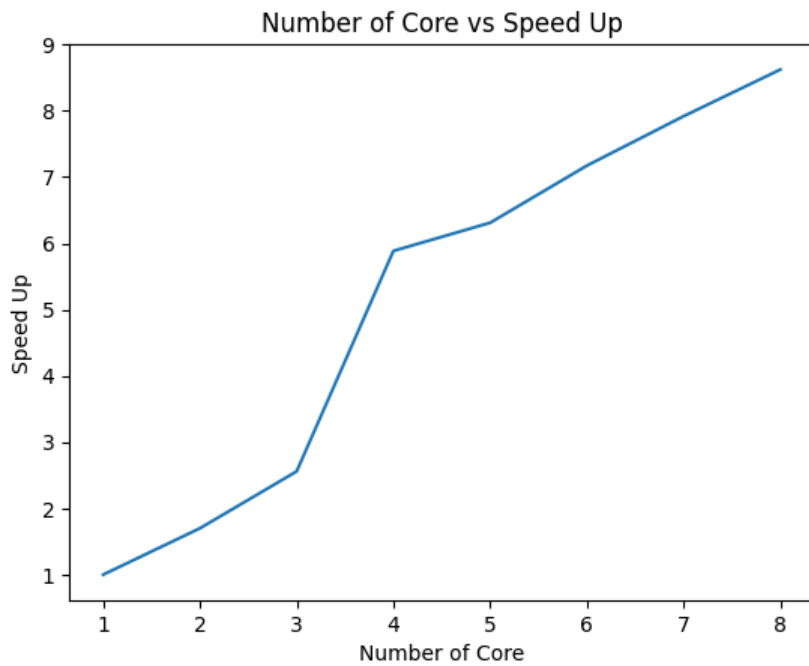
```
In [6]: data['Elapsed Time'][7]
```

```
Out[6]: 707.6593961715698
```

```
In [7]: speed_up = []
for i in data['Number of Cores']:
    speed_up.append(data['Elapsed Time'][0] / data['Elapsed Time'][i - 1])

plt.plot(data['Number of Cores'], speed_up)
plt.xlabel('Number of Core')
plt.ylabel('Speed Up')
```

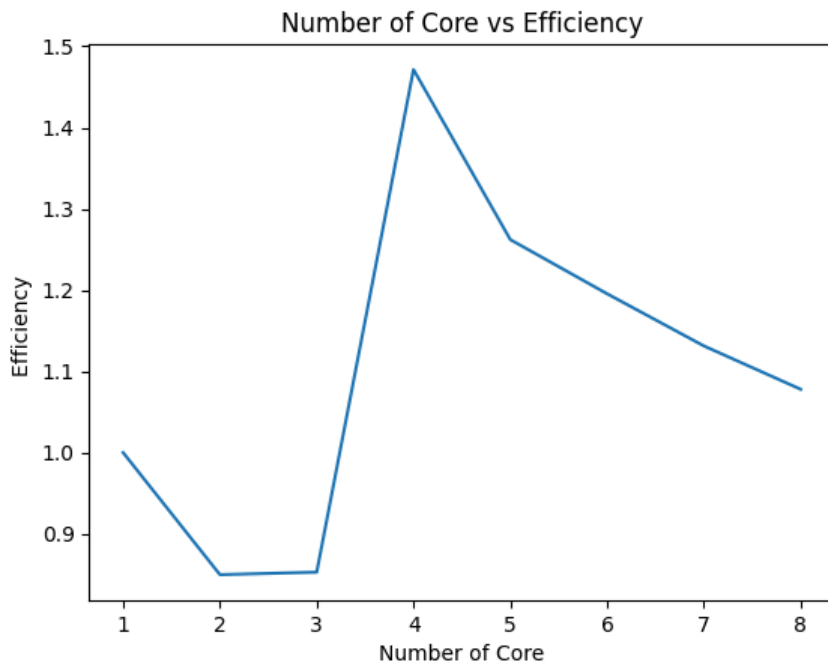
```
plt.title('Number of Core vs Speed Up')
plt.show()
```



Number of Core vs Efficiency

```
In [8]: Efficiency = []
for num_of_core in data['Number of Cores']:
    Efficiency.append(speed_up[num_of_core - 1] / num_of_core)

plt.plot(data['Number of Cores'], Efficiency)
plt.xlabel('Number of Core')
plt.ylabel('Efficiency ')
plt.title('Number of Core vs Efficiency')
plt.show()
```



```
In [ ]:
```

Import Req Lib

```
In [1]: %matplotlib inline

import shutil
import random
import numpy as np
from warnings import filterwarnings
filterwarnings('ignore')

from tensorflow.keras import layers, regularizers, optimizers
from tensorflow.keras import models
from tensorflow.keras.models import Sequential, Model
from tensorflow.keras.layers import LeakyReLU, Dense, Activation, Flatten, Dropout, BatchNormalization, Conv2D, MaxPooling2D
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import tensorflow as tf

import os
import time
import csv

from matplotlib import figure
```

Define 1 worker

```
In [2]: # Set the number of threads
number_of_worker = 1
os.environ['OMP_NUM_THREADS'] = '1' # OpenMP threads for parallelism
os.environ['TF_NUM_INTEROP_THREADS'] = '1' # Threads for inter-operation parallelism
os.environ['TF_NUM_INTRAOP_THREADS'] = '1' # Threads for intra-operation parallelism

# Confirm TensorFlow is using the specified number of threads
tf.config.threading.set_inter_op_parallelism_threads(number_of_worker)
tf.config.threading.set_intra_op_parallelism_threads(number_of_worker)
```

Train Val data Split

```
In [3]: source_dir = r"C:\Users\nikhi\OneDrive\Desktop\Final Project\DATA\Convert_Audio_File_to_jpg_file"
target_dir = r'genres_train_val_split_data'
split_ratio = 0.8

def Train_Test_Split(source_dir, target_dir, split_ratio):
    # Define source and target directories
    train_dir = os.path.join(target_dir, 'train')
    val_dir = os.path.join(target_dir, 'val')

    # Create target directories if they don't exist
    os.makedirs(train_dir, exist_ok=True)
    os.makedirs(val_dir, exist_ok=True)

    # Get the list of class directories
    classes = [d for d in os.listdir(source_dir) if os.path.isdir(os.path.join(source_dir, d))]

    for class_name in classes:
        # Create class directories in train and val folders
        os.makedirs(os.path.join(train_dir, class_name), exist_ok=True)
        os.makedirs(os.path.join(val_dir, class_name), exist_ok=True)

        # Get list of images in the class directory
        class_dir = os.path.join(source_dir, class_name)
        images = [f for f in os.listdir(class_dir) if os.path.isfile(os.path.join(class_dir, f))]

        # Shuffle the images
        random.shuffle(images)

        # Compute the split point
        split_point = int(len(images) * split_ratio)

        # Split the images into training and validation sets
        train_images = images[:split_point]
        val_images = images[split_point:]
```

```

# Move the images to the respective directories
for img in train_images:
    shutil.copy(os.path.join(class_dir, img), os.path.join(train_dir, class_name, img))

for img in val_images:
    shutil.copy(os.path.join(class_dir, img), os.path.join(val_dir, class_name, img))

print("Data split completed successfully!")

```

In [4]: `Train_Test_Split(source_dir,target_dir,split_ratio)`

Data split completed successfully!

Load the Data

```

In [5]: WIDTH = 64
HEIGHT = 64
BATCH_SIZE = 32
TRAIN_DIR=r'genres_train_val_split_data/train'
val_dir = r'genres_train_val_split_data/val'

# data prep
train_datagen = ImageDataGenerator(
    rescale=1./255.,validation_split=0.25)

train_generator = train_datagen.flow_from_directory(
    TRAIN_DIR,
    target_size=(HEIGHT, WIDTH),
    batch_size=BATCH_SIZE,
    class_mode='categorical')

validation_gen = train_datagen.flow_from_directory(
    val_dir,target_size = (HEIGHT,WIDTH),
    batch_size = BATCH_SIZE,
    class_mode = 'categorical'
)

```

Found 800 images belonging to 10 classes.

Found 200 images belonging to 10 classes.

Model Architecture

```

In [6]: model = Sequential()
model.add(Conv2D(32, (3, 3), padding='same',
    input_shape=(64,64,3)))
model.add(Activation('relu'))
model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Conv2D(64, (3, 3), padding='same'))
model.add(Activation('relu'))
model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.5))
model.add(Conv2D(128, (3, 3), padding='same'))
model.add(Activation('relu'))
model.add(Conv2D(128, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.5))
model.add(Flatten())
model.add(Dense(512))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(10, activation='softmax'))
model.compile(optimizer=RMSprop(learning_rate=0.0005, decay=1e-6),loss="categorical_crossentropy",metrics=["accuracy"])
model.summary()

```

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 64, 64, 32)	896
activation (Activation)	(None, 64, 64, 32)	0
conv2d_1 (Conv2D)	(None, 62, 62, 64)	18,496
activation_1 (Activation)	(None, 62, 62, 64)	0
max_pooling2d (MaxPooling2D)	(None, 31, 31, 64)	0
dropout (Dropout)	(None, 31, 31, 64)	0
conv2d_2 (Conv2D)	(None, 31, 31, 64)	36,928
activation_2 (Activation)	(None, 31, 31, 64)	0
conv2d_3 (Conv2D)	(None, 29, 29, 64)	36,928
activation_3 (Activation)	(None, 29, 29, 64)	0
max_pooling2d_1 (MaxPooling2D)	(None, 14, 14, 64)	0
dropout_1 (Dropout)	(None, 14, 14, 64)	0
conv2d_4 (Conv2D)	(None, 14, 14, 128)	73,856
activation_4 (Activation)	(None, 14, 14, 128)	0
conv2d_5 (Conv2D)	(None, 12, 12, 128)	147,584
activation_5 (Activation)	(None, 12, 12, 128)	0
max_pooling2d_2 (MaxPooling2D)	(None, 6, 6, 128)	0
dropout_2 (Dropout)	(None, 6, 6, 128)	0
flatten (Flatten)	(None, 4608)	0
dense (Dense)	(None, 512)	2,359,808
activation_6 (Activation)	(None, 512)	0
dropout_3 (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 10)	5,130

Total params: 2,679,626 (10.22 MB)

Trainable params: 2,679,626 (10.22 MB)

Non-trainable params: 0 (0.00 B)

```
In [7]: STEP_SIZE_TRAIN=train_generator.n//train_generator.batch_size
# Measure the execution time
start_time = time.time()

model.fit(train_generator,validation_data=validation_gen,epochs=200)

end_time = time.time()
elapsed_time = end_time - start_time
```

Epoch 1/200	25/25	32s	1s/step	- accuracy: 0.1023	- loss: 2.3116	- val_accuracy: 0.1750	- val_loss: 2.2917
Epoch 2/200	25/25	28s	1s/step	- accuracy: 0.1221	- loss: 2.2753	- val_accuracy: 0.2100	- val_loss: 2.1145
Epoch 3/200	25/25	28s	1s/step	- accuracy: 0.1855	- loss: 2.2138	- val_accuracy: 0.1950	- val_loss: 2.1203
Epoch 4/200	25/25	28s	1s/step	- accuracy: 0.1924	- loss: 2.1035	- val_accuracy: 0.2450	- val_loss: 1.9382
Epoch 5/200	25/25	27s	1s/step	- accuracy: 0.2253	- loss: 2.0658	- val_accuracy: 0.2900	- val_loss: 1.9680
Epoch 6/200	25/25	26s	1s/step	- accuracy: 0.2518	- loss: 2.0428	- val_accuracy: 0.3600	- val_loss: 2.0611
Epoch 7/200	25/25	26s	1s/step	- accuracy: 0.2844	- loss: 1.9486	- val_accuracy: 0.3550	- val_loss: 1.8568
Epoch 8/200	25/25	26s	1s/step	- accuracy: 0.3084	- loss: 1.8901	- val_accuracy: 0.2650	- val_loss: 2.1513
Epoch 9/200	25/25	26s	1s/step	- accuracy: 0.3282	- loss: 1.9096	- val_accuracy: 0.3850	- val_loss: 1.7727
Epoch 10/200	25/25	26s	1s/step	- accuracy: 0.3618	- loss: 1.7941	- val_accuracy: 0.4300	- val_loss: 1.6640
Epoch 11/200	25/25	27s	1s/step	- accuracy: 0.3581	- loss: 1.7205	- val_accuracy: 0.3650	- val_loss: 1.7061
Epoch 12/200	25/25	26s	1s/step	- accuracy: 0.3839	- loss: 1.7241	- val_accuracy: 0.4250	- val_loss: 1.6044
Epoch 13/200	25/25	25s	1s/step	- accuracy: 0.4243	- loss: 1.6182	- val_accuracy: 0.3600	- val_loss: 1.7587
Epoch 14/200	25/25	26s	1s/step	- accuracy: 0.4233	- loss: 1.6799	- val_accuracy: 0.4700	- val_loss: 1.5579
Epoch 15/200	25/25	25s	1s/step	- accuracy: 0.4338	- loss: 1.5815	- val_accuracy: 0.4800	- val_loss: 1.5056
Epoch 16/200	25/25	28s	1s/step	- accuracy: 0.4718	- loss: 1.4877	- val_accuracy: 0.3650	- val_loss: 1.6719
Epoch 17/200	25/25	47s	1s/step	- accuracy: 0.4637	- loss: 1.5132	- val_accuracy: 0.4750	- val_loss: 1.4645
Epoch 18/200	25/25	28s	1s/step	- accuracy: 0.4672	- loss: 1.4364	- val_accuracy: 0.5000	- val_loss: 1.4407
Epoch 19/200	25/25	27s	1s/step	- accuracy: 0.4664	- loss: 1.4338	- val_accuracy: 0.4600	- val_loss: 1.4912
Epoch 20/200	25/25	26s	1s/step	- accuracy: 0.4846	- loss: 1.3805	- val_accuracy: 0.4950	- val_loss: 1.4114
Epoch 21/200	25/25	28s	1s/step	- accuracy: 0.5301	- loss: 1.3195	- val_accuracy: 0.5650	- val_loss: 1.3299
Epoch 22/200	25/25	39s	1s/step	- accuracy: 0.5377	- loss: 1.3187	- val_accuracy: 0.5200	- val_loss: 1.4057
Epoch 23/200	25/25	26s	1s/step	- accuracy: 0.5209	- loss: 1.3272	- val_accuracy: 0.5250	- val_loss: 1.3737
Epoch 24/200	25/25	45s	1s/step	- accuracy: 0.5138	- loss: 1.3195	- val_accuracy: 0.4800	- val_loss: 1.3970
Epoch 25/200	25/25	27s	1s/step	- accuracy: 0.5613	- loss: 1.2026	- val_accuracy: 0.4850	- val_loss: 1.2982
Epoch 26/200	25/25	27s	1s/step	- accuracy: 0.5462	- loss: 1.2349	- val_accuracy: 0.5150	- val_loss: 1.4277
Epoch 27/200	25/25	28s	1s/step	- accuracy: 0.4924	- loss: 1.3835	- val_accuracy: 0.5600	- val_loss: 1.2649
Epoch 28/200	25/25	27s	1s/step	- accuracy: 0.5781	- loss: 1.2120	- val_accuracy: 0.5400	- val_loss: 1.3177
Epoch 29/200	25/25	28s	1s/step	- accuracy: 0.5310	- loss: 1.2935	- val_accuracy: 0.5350	- val_loss: 1.2755
Epoch 30/200	25/25	27s	1s/step	- accuracy: 0.5842	- loss: 1.1832	- val_accuracy: 0.5650	- val_loss: 1.2301
Epoch 31/200	25/25	27s	1s/step	- accuracy: 0.6170	- loss: 1.1483	- val_accuracy: 0.5300	- val_loss: 1.2360
Epoch 32/200	25/25	27s	1s/step	- accuracy: 0.5694	- loss: 1.1414	- val_accuracy: 0.5300	- val_loss: 1.1925
Epoch 33/200	25/25	28s	1s/step	- accuracy: 0.5970	- loss: 1.1031	- val_accuracy: 0.5650	- val_loss: 1.1584
Epoch 34/200	25/25	28s	1s/step	- accuracy: 0.6325	- loss: 1.0118	- val_accuracy: 0.6000	- val_loss: 1.1658
Epoch 35/200	25/25	28s	1s/step	- accuracy: 0.6461	- loss: 1.0089	- val_accuracy: 0.5200	- val_loss: 1.3458
Epoch 36/200	25/25	27s	1s/step	- accuracy: 0.6280	- loss: 1.0219	- val_accuracy: 0.5500	- val_loss: 1.2555
Epoch 37/200	25/25	29s	1s/step	- accuracy: 0.6563	- loss: 1.0223	- val_accuracy: 0.5600	- val_loss: 1.2584
Epoch 38/200	25/25	29s	1s/step	- accuracy: 0.6219	- loss: 1.0022	- val_accuracy: 0.5750	- val_loss: 1.1565
Epoch 39/200	25/25	28s	1s/step	- accuracy: 0.6673	- loss: 0.9150	- val_accuracy: 0.6000	- val_loss: 1.1377
Epoch 40/200	25/25	27s	1s/step	- accuracy: 0.6545	- loss: 0.9413	- val_accuracy: 0.5750	- val_loss: 1.1518
Epoch 41/200	25/25	28s	1s/step	- accuracy: 0.6994	- loss: 0.8533	- val_accuracy: 0.5050	- val_loss: 1.4275

Epoch 42/200	25/25	42s	1s/step	-	accuracy: 0.6991	-	loss: 0.8246	-	val_accuracy: 0.5700	-	val_loss: 1.1758
Epoch 43/200	25/25	27s	1s/step	-	accuracy: 0.7177	-	loss: 0.7888	-	val_accuracy: 0.6200	-	val_loss: 1.2086
Epoch 44/200	25/25	29s	1s/step	-	accuracy: 0.7449	-	loss: 0.7547	-	val_accuracy: 0.6150	-	val_loss: 1.1535
Epoch 45/200	25/25	42s	1s/step	-	accuracy: 0.7052	-	loss: 0.7935	-	val_accuracy: 0.5850	-	val_loss: 1.3092
Epoch 46/200	25/25	39s	1s/step	-	accuracy: 0.7307	-	loss: 0.7758	-	val_accuracy: 0.5550	-	val_loss: 1.1893
Epoch 47/200	25/25	41s	1s/step	-	accuracy: 0.7160	-	loss: 0.8295	-	val_accuracy: 0.5800	-	val_loss: 1.2159
Epoch 48/200	25/25	29s	1s/step	-	accuracy: 0.7674	-	loss: 0.6261	-	val_accuracy: 0.6150	-	val_loss: 1.2276
Epoch 49/200	25/25	27s	1s/step	-	accuracy: 0.7672	-	loss: 0.6559	-	val_accuracy: 0.6100	-	val_loss: 1.2374
Epoch 50/200	25/25	29s	1s/step	-	accuracy: 0.7832	-	loss: 0.6168	-	val_accuracy: 0.5850	-	val_loss: 1.3392
Epoch 51/200	25/25	30s	1s/step	-	accuracy: 0.7379	-	loss: 0.7095	-	val_accuracy: 0.5750	-	val_loss: 1.4176
Epoch 52/200	25/25	30s	1s/step	-	accuracy: 0.7792	-	loss: 0.5799	-	val_accuracy: 0.5300	-	val_loss: 1.4030
Epoch 53/200	25/25	30s	1s/step	-	accuracy: 0.7901	-	loss: 0.5600	-	val_accuracy: 0.6350	-	val_loss: 1.2566
Epoch 54/200	25/25	28s	1s/step	-	accuracy: 0.7973	-	loss: 0.5726	-	val_accuracy: 0.5500	-	val_loss: 1.3714
Epoch 55/200	25/25	27s	1s/step	-	accuracy: 0.7668	-	loss: 0.7013	-	val_accuracy: 0.6250	-	val_loss: 1.3406
Epoch 56/200	25/25	27s	1s/step	-	accuracy: 0.7882	-	loss: 0.5294	-	val_accuracy: 0.6150	-	val_loss: 1.3486
Epoch 57/200	25/25	30s	1s/step	-	accuracy: 0.8414	-	loss: 0.4948	-	val_accuracy: 0.5900	-	val_loss: 1.4015
Epoch 58/200	25/25	27s	1s/step	-	accuracy: 0.8212	-	loss: 0.4944	-	val_accuracy: 0.6150	-	val_loss: 1.4117
Epoch 59/200	25/25	43s	1s/step	-	accuracy: 0.8479	-	loss: 0.4028	-	val_accuracy: 0.6050	-	val_loss: 1.4449
Epoch 60/200	25/25	29s	1s/step	-	accuracy: 0.8089	-	loss: 0.5510	-	val_accuracy: 0.5700	-	val_loss: 1.5338
Epoch 61/200	25/25	28s	1s/step	-	accuracy: 0.8237	-	loss: 0.4647	-	val_accuracy: 0.5950	-	val_loss: 1.4833
Epoch 62/200	25/25	41s	1s/step	-	accuracy: 0.8668	-	loss: 0.3522	-	val_accuracy: 0.5650	-	val_loss: 1.4804
Epoch 63/200	25/25	28s	1s/step	-	accuracy: 0.8646	-	loss: 0.4137	-	val_accuracy: 0.6200	-	val_loss: 1.5929
Epoch 64/200	25/25	29s	1s/step	-	accuracy: 0.8611	-	loss: 0.4100	-	val_accuracy: 0.6150	-	val_loss: 1.5140
Epoch 65/200	25/25	30s	1s/step	-	accuracy: 0.8943	-	loss: 0.2995	-	val_accuracy: 0.5300	-	val_loss: 2.0294
Epoch 66/200	25/25	28s	1s/step	-	accuracy: 0.8659	-	loss: 0.3771	-	val_accuracy: 0.6100	-	val_loss: 1.3474
Epoch 67/200	25/25	28s	1s/step	-	accuracy: 0.8714	-	loss: 0.3636	-	val_accuracy: 0.6450	-	val_loss: 1.4715
Epoch 68/200	25/25	26s	1s/step	-	accuracy: 0.8731	-	loss: 0.3432	-	val_accuracy: 0.6350	-	val_loss: 1.5430
Epoch 69/200	25/25	28s	1s/step	-	accuracy: 0.9113	-	loss: 0.2766	-	val_accuracy: 0.6000	-	val_loss: 1.6362
Epoch 70/200	25/25	30s	1s/step	-	accuracy: 0.9115	-	loss: 0.2632	-	val_accuracy: 0.5900	-	val_loss: 1.7935
Epoch 71/200	25/25	30s	1s/step	-	accuracy: 0.8804	-	loss: 0.3234	-	val_accuracy: 0.5850	-	val_loss: 1.7517
Epoch 72/200	25/25	29s	1s/step	-	accuracy: 0.9097	-	loss: 0.2619	-	val_accuracy: 0.6050	-	val_loss: 1.7657
Epoch 73/200	25/25	31s	1s/step	-	accuracy: 0.8978	-	loss: 0.2883	-	val_accuracy: 0.5900	-	val_loss: 1.9864
Epoch 74/200	25/25	29s	1s/step	-	accuracy: 0.8944	-	loss: 0.2651	-	val_accuracy: 0.6050	-	val_loss: 1.7086
Epoch 75/200	25/25	28s	1s/step	-	accuracy: 0.9287	-	loss: 0.2225	-	val_accuracy: 0.6150	-	val_loss: 1.8275
Epoch 76/200	25/25	28s	1s/step	-	accuracy: 0.9313	-	loss: 0.2132	-	val_accuracy: 0.5600	-	val_loss: 1.9665
Epoch 77/200	25/25	30s	1s/step	-	accuracy: 0.9132	-	loss: 0.2315	-	val_accuracy: 0.6250	-	val_loss: 1.7940
Epoch 78/200	25/25	30s	1s/step	-	accuracy: 0.9298	-	loss: 0.2025	-	val_accuracy: 0.6100	-	val_loss: 1.7548
Epoch 79/200	25/25	28s	1s/step	-	accuracy: 0.8969	-	loss: 0.2864	-	val_accuracy: 0.5700	-	val_loss: 1.9362
Epoch 80/200	25/25	28s	1s/step	-	accuracy: 0.9037	-	loss: 0.2708	-	val_accuracy: 0.6050	-	val_loss: 1.7875
Epoch 81/200	25/25	30s	1s/step	-	accuracy: 0.9479	-	loss: 0.1578	-	val_accuracy: 0.6050	-	val_loss: 1.9154
Epoch 82/200	25/25	30s	1s/step	-	accuracy: 0.9278	-	loss: 0.2297	-	val_accuracy: 0.6100	-	val_loss: 2.0085

Epoch 83/200	25/25	28s	1s/step	- accuracy: 0.9603	- loss: 0.1253	- val_accuracy: 0.6050	- val_loss: 1.9944
Epoch 84/200	25/25	28s	1s/step	- accuracy: 0.9349	- loss: 0.2179	- val_accuracy: 0.5950	- val_loss: 1.7698
Epoch 85/200	25/25	29s	1s/step	- accuracy: 0.9260	- loss: 0.2480	- val_accuracy: 0.6000	- val_loss: 1.9084
Epoch 86/200	25/25	29s	1s/step	- accuracy: 0.9385	- loss: 0.1810	- val_accuracy: 0.6050	- val_loss: 1.9800
Epoch 87/200	25/25	40s	1s/step	- accuracy: 0.9315	- loss: 0.2115	- val_accuracy: 0.6000	- val_loss: 1.9982
Epoch 88/200	25/25	29s	1s/step	- accuracy: 0.9286	- loss: 0.1841	- val_accuracy: 0.6050	- val_loss: 1.8210
Epoch 89/200	25/25	42s	1s/step	- accuracy: 0.9449	- loss: 0.1455	- val_accuracy: 0.6050	- val_loss: 2.1485
Epoch 90/200	25/25	30s	1s/step	- accuracy: 0.9580	- loss: 0.1347	- val_accuracy: 0.6050	- val_loss: 2.0624
Epoch 91/200	25/25	29s	1s/step	- accuracy: 0.9658	- loss: 0.1287	- val_accuracy: 0.6050	- val_loss: 2.0623
Epoch 92/200	25/25	41s	1s/step	- accuracy: 0.9296	- loss: 0.2185	- val_accuracy: 0.5750	- val_loss: 2.1836
Epoch 93/200	25/25	31s	1s/step	- accuracy: 0.9534	- loss: 0.1375	- val_accuracy: 0.5700	- val_loss: 2.3075
Epoch 94/200	25/25	28s	1s/step	- accuracy: 0.9321	- loss: 0.1736	- val_accuracy: 0.5900	- val_loss: 2.0975
Epoch 95/200	25/25	28s	1s/step	- accuracy: 0.9544	- loss: 0.1090	- val_accuracy: 0.6050	- val_loss: 2.1837
Epoch 96/200	25/25	29s	1s/step	- accuracy: 0.9666	- loss: 0.1233	- val_accuracy: 0.5250	- val_loss: 2.7924
Epoch 97/200	25/25	29s	1s/step	- accuracy: 0.9305	- loss: 0.2006	- val_accuracy: 0.5950	- val_loss: 2.1621
Epoch 98/200	25/25	29s	1s/step	- accuracy: 0.9653	- loss: 0.1326	- val_accuracy: 0.5600	- val_loss: 2.3662
Epoch 99/200	25/25	40s	1s/step	- accuracy: 0.9604	- loss: 0.1206	- val_accuracy: 0.6150	- val_loss: 2.2831
Epoch 100/200	25/25	29s	1s/step	- accuracy: 0.9483	- loss: 0.1839	- val_accuracy: 0.6150	- val_loss: 1.8584
Epoch 101/200	25/25	30s	1s/step	- accuracy: 0.9725	- loss: 0.0949	- val_accuracy: 0.5950	- val_loss: 2.1361
Epoch 102/200	25/25	28s	1s/step	- accuracy: 0.9591	- loss: 0.1132	- val_accuracy: 0.6500	- val_loss: 2.1777
Epoch 103/200	25/25	28s	1s/step	- accuracy: 0.9600	- loss: 0.1332	- val_accuracy: 0.6000	- val_loss: 3.0784
Epoch 104/200	25/25	29s	1s/step	- accuracy: 0.9355	- loss: 0.1872	- val_accuracy: 0.5750	- val_loss: 2.4978
Epoch 105/200	25/25	30s	1s/step	- accuracy: 0.9633	- loss: 0.1094	- val_accuracy: 0.6200	- val_loss: 2.2200
Epoch 106/200	25/25	28s	1s/step	- accuracy: 0.9577	- loss: 0.1378	- val_accuracy: 0.5850	- val_loss: 2.5441
Epoch 107/200	25/25	41s	1s/step	- accuracy: 0.9277	- loss: 0.2046	- val_accuracy: 0.6250	- val_loss: 2.1615
Epoch 108/200	25/25	28s	1s/step	- accuracy: 0.9599	- loss: 0.0948	- val_accuracy: 0.5750	- val_loss: 2.4605
Epoch 109/200	25/25	43s	1s/step	- accuracy: 0.9670	- loss: 0.0811	- val_accuracy: 0.5950	- val_loss: 2.5483
Epoch 110/200	25/25	28s	1s/step	- accuracy: 0.9646	- loss: 0.1045	- val_accuracy: 0.5600	- val_loss: 3.1649
Epoch 111/200	25/25	29s	1s/step	- accuracy: 0.9542	- loss: 0.1843	- val_accuracy: 0.5850	- val_loss: 2.3699
Epoch 112/200	25/25	31s	1s/step	- accuracy: 0.9595	- loss: 0.0938	- val_accuracy: 0.6200	- val_loss: 2.5488
Epoch 113/200	25/25	29s	1s/step	- accuracy: 0.9756	- loss: 0.0709	- val_accuracy: 0.6450	- val_loss: 2.7340
Epoch 114/200	25/25	28s	1s/step	- accuracy: 0.9678	- loss: 0.0969	- val_accuracy: 0.5950	- val_loss: 2.5608
Epoch 115/200	25/25	28s	1s/step	- accuracy: 0.9566	- loss: 0.1611	- val_accuracy: 0.6350	- val_loss: 2.2199
Epoch 116/200	25/25	30s	1s/step	- accuracy: 0.9844	- loss: 0.0740	- val_accuracy: 0.5950	- val_loss: 2.5361
Epoch 117/200	25/25	30s	1s/step	- accuracy: 0.9755	- loss: 0.0631	- val_accuracy: 0.6150	- val_loss: 2.5037
Epoch 118/200	25/25	28s	1s/step	- accuracy: 0.9773	- loss: 0.0840	- val_accuracy: 0.5650	- val_loss: 2.7249
Epoch 119/200	25/25	28s	1s/step	- accuracy: 0.9873	- loss: 0.0544	- val_accuracy: 0.6150	- val_loss: 2.7848
Epoch 120/200	25/25	42s	1s/step	- accuracy: 0.9768	- loss: 0.0695	- val_accuracy: 0.6100	- val_loss: 2.9545
Epoch 121/200	25/25	29s	1s/step	- accuracy: 0.9731	- loss: 0.0767	- val_accuracy: 0.6150	- val_loss: 2.5158
Epoch 122/200	25/25	28s	1s/step	- accuracy: 0.9797	- loss: 0.0810	- val_accuracy: 0.6100	- val_loss: 2.5275
Epoch 123/200	25/25	28s	1s/step	- accuracy: 0.9807	- loss: 0.0617	- val_accuracy: 0.5100	- val_loss: 3.4139

Epoch 124/200
25/25 ————— 30s 1s/step - accuracy: 0.9686 - loss: 0.1716 - val_accuracy: 0.5950 - val_loss: 2.5259

Epoch 125/200
25/25 ————— 29s 1s/step - accuracy: 0.9579 - loss: 0.1311 - val_accuracy: 0.6000 - val_loss: 2.3598

Epoch 126/200
25/25 ————— 28s 1s/step - accuracy: 0.9614 - loss: 0.1006 - val_accuracy: 0.6250 - val_loss: 2.3710

Epoch 127/200
25/25 ————— 42s 1s/step - accuracy: 0.9831 - loss: 0.0494 - val_accuracy: 0.5900 - val_loss: 2.4309

Epoch 128/200
25/25 ————— 30s 1s/step - accuracy: 0.9865 - loss: 0.0413 - val_accuracy: 0.6150 - val_loss: 2.5843

Epoch 129/200
25/25 ————— 29s 1s/step - accuracy: 0.9723 - loss: 0.0938 - val_accuracy: 0.6200 - val_loss: 2.5107

Epoch 130/200
25/25 ————— 40s 1s/step - accuracy: 0.9890 - loss: 0.0352 - val_accuracy: 0.6250 - val_loss: 2.6177

Epoch 131/200
25/25 ————— 42s 1s/step - accuracy: 0.9732 - loss: 0.1001 - val_accuracy: 0.6350 - val_loss: 2.6362

Epoch 132/200
25/25 ————— 28s 1s/step - accuracy: 0.9685 - loss: 0.0753 - val_accuracy: 0.6450 - val_loss: 2.7245

Epoch 133/200
25/25 ————— 30s 1s/step - accuracy: 0.9689 - loss: 0.1208 - val_accuracy: 0.6350 - val_loss: 2.4855

Epoch 134/200
25/25 ————— 29s 1s/step - accuracy: 0.9775 - loss: 0.0830 - val_accuracy: 0.5800 - val_loss: 2.6929

Epoch 135/200
25/25 ————— 27s 1s/step - accuracy: 0.9776 - loss: 0.0684 - val_accuracy: 0.6250 - val_loss: 2.4147

Epoch 136/200
25/25 ————— 29s 1s/step - accuracy: 0.9900 - loss: 0.0449 - val_accuracy: 0.6200 - val_loss: 2.4069

Epoch 137/200
25/25 ————— 28s 1s/step - accuracy: 0.9814 - loss: 0.0603 - val_accuracy: 0.6400 - val_loss: 2.3076

Epoch 138/200
25/25 ————— 29s 1s/step - accuracy: 0.9831 - loss: 0.0480 - val_accuracy: 0.6300 - val_loss: 2.4884

Epoch 139/200
25/25 ————— 29s 1s/step - accuracy: 0.9798 - loss: 0.0811 - val_accuracy: 0.6100 - val_loss: 2.8976

Epoch 140/200
25/25 ————— 30s 1s/step - accuracy: 0.9540 - loss: 0.1233 - val_accuracy: 0.6000 - val_loss: 2.5558

Epoch 141/200
25/25 ————— 28s 1s/step - accuracy: 0.9726 - loss: 0.0607 - val_accuracy: 0.6150 - val_loss: 2.5155

Epoch 142/200
25/25 ————— 26s 1s/step - accuracy: 0.9861 - loss: 0.0612 - val_accuracy: 0.6150 - val_loss: 2.5438

Epoch 143/200
25/25 ————— 27s 1s/step - accuracy: 0.9839 - loss: 0.0469 - val_accuracy: 0.6150 - val_loss: 2.7714

Epoch 144/200
25/25 ————— 28s 1s/step - accuracy: 0.9804 - loss: 0.0644 - val_accuracy: 0.6050 - val_loss: 2.6876

Epoch 145/200
25/25 ————— 28s 1s/step - accuracy: 0.9745 - loss: 0.0825 - val_accuracy: 0.6150 - val_loss: 2.5074

Epoch 146/200
25/25 ————— 28s 1s/step - accuracy: 0.9850 - loss: 0.0500 - val_accuracy: 0.5700 - val_loss: 2.7695

Epoch 147/200
25/25 ————— 13s 534ms/step - accuracy: 0.9874 - loss: 0.0435 - val_accuracy: 0.6100 - val_loss: 2.9878

Epoch 148/200
25/25 ————— 13s 516ms/step - accuracy: 0.9890 - loss: 0.0413 - val_accuracy: 0.5550 - val_loss: 2.9865

Epoch 149/200
25/25 ————— 13s 506ms/step - accuracy: 0.9630 - loss: 0.1362 - val_accuracy: 0.5950 - val_loss: 2.6534

Epoch 150/200
25/25 ————— 13s 536ms/step - accuracy: 0.9646 - loss: 0.1046 - val_accuracy: 0.6100 - val_loss: 2.6097

Epoch 151/200
25/25 ————— 13s 516ms/step - accuracy: 0.9851 - loss: 0.0577 - val_accuracy: 0.5700 - val_loss: 3.1224

Epoch 152/200
25/25 ————— 12s 472ms/step - accuracy: 0.9864 - loss: 0.0592 - val_accuracy: 0.6250 - val_loss: 3.0793

Epoch 153/200
25/25 ————— 12s 493ms/step - accuracy: 0.9766 - loss: 0.1038 - val_accuracy: 0.5900 - val_loss: 3.0936

Epoch 154/200
25/25 ————— 14s 545ms/step - accuracy: 0.9854 - loss: 0.0362 - val_accuracy: 0.6250 - val_loss: 2.8501

Epoch 155/200
25/25 ————— 13s 512ms/step - accuracy: 0.9807 - loss: 0.0572 - val_accuracy: 0.6350 - val_loss: 2.8597

Epoch 156/200
25/25 ————— 14s 553ms/step - accuracy: 0.9818 - loss: 0.0439 - val_accuracy: 0.6250 - val_loss: 2.8805

Epoch 157/200
25/25 ————— 13s 523ms/step - accuracy: 0.9693 - loss: 0.1114 - val_accuracy: 0.6150 - val_loss: 2.8996

Epoch 158/200
25/25 ————— 35s 1s/step - accuracy: 0.9876 - loss: 0.0445 - val_accuracy: 0.6100 - val_loss: 2.5862

Epoch 159/200
25/25 ————— 49s 2s/step - accuracy: 0.9832 - loss: 0.0491 - val_accuracy: 0.6150 - val_loss: 2.7456

Epoch 160/200
25/25 ————— 45s 2s/step - accuracy: 0.9890 - loss: 0.0416 - val_accuracy: 0.6350 - val_loss: 2.6034

Epoch 161/200
25/25 ————— 45s 2s/step - accuracy: 0.9840 - loss: 0.0574 - val_accuracy: 0.6200 - val_loss: 2.5495

Epoch 162/200
25/25 ————— 36s 1s/step - accuracy: 0.9752 - loss: 0.0676 - val_accuracy: 0.6400 - val_loss: 2.4306

Epoch 163/200
25/25 ————— 31s 1s/step - accuracy: 0.9842 - loss: 0.0456 - val_accuracy: 0.5650 - val_loss: 3.2392

Epoch 164/200
25/25 ————— 44s 2s/step - accuracy: 0.9629 - loss: 0.1236 - val_accuracy: 0.6050 - val_loss: 2.5910

```

Epoch 165/200
25/25 ————— 39s 2s/step - accuracy: 0.9845 - loss: 0.0498 - val_accuracy: 0.5600 - val_loss: 3.4614
Epoch 166/200
25/25 ————— 41s 2s/step - accuracy: 0.9775 - loss: 0.0778 - val_accuracy: 0.5850 - val_loss: 3.4735
Epoch 167/200
25/25 ————— 39s 2s/step - accuracy: 0.9761 - loss: 0.1006 - val_accuracy: 0.6250 - val_loss: 2.8417
Epoch 168/200
25/25 ————— 36s 1s/step - accuracy: 0.9890 - loss: 0.0297 - val_accuracy: 0.6100 - val_loss: 3.5556
Epoch 169/200
25/25 ————— 42s 2s/step - accuracy: 0.9769 - loss: 0.0796 - val_accuracy: 0.6250 - val_loss: 2.8833
Epoch 170/200
25/25 ————— 76s 2s/step - accuracy: 0.9927 - loss: 0.0417 - val_accuracy: 0.5800 - val_loss: 3.4683
Epoch 171/200
25/25 ————— 38s 2s/step - accuracy: 0.9753 - loss: 0.1325 - val_accuracy: 0.5600 - val_loss: 3.0028
Epoch 172/200
25/25 ————— 41s 2s/step - accuracy: 0.9977 - loss: 0.0172 - val_accuracy: 0.5900 - val_loss: 3.1076
Epoch 173/200
25/25 ————— 36s 1s/step - accuracy: 0.9821 - loss: 0.0533 - val_accuracy: 0.6150 - val_loss: 2.7431
Epoch 174/200
25/25 ————— 32s 1s/step - accuracy: 0.9824 - loss: 0.0689 - val_accuracy: 0.6100 - val_loss: 2.9436
Epoch 175/200
25/25 ————— 45s 1s/step - accuracy: 0.9848 - loss: 0.0335 - val_accuracy: 0.6350 - val_loss: 3.0803
Epoch 176/200
25/25 ————— 42s 2s/step - accuracy: 0.9724 - loss: 0.0594 - val_accuracy: 0.6200 - val_loss: 2.8165
Epoch 177/200
25/25 ————— 38s 2s/step - accuracy: 0.9929 - loss: 0.0385 - val_accuracy: 0.5950 - val_loss: 2.7957
Epoch 178/200
25/25 ————— 39s 1s/step - accuracy: 0.9721 - loss: 0.0697 - val_accuracy: 0.5900 - val_loss: 3.0092
Epoch 179/200
25/25 ————— 37s 1s/step - accuracy: 0.9793 - loss: 0.0498 - val_accuracy: 0.6150 - val_loss: 3.1001
Epoch 180/200
25/25 ————— 40s 1s/step - accuracy: 0.9918 - loss: 0.0250 - val_accuracy: 0.6150 - val_loss: 3.1547
Epoch 181/200
25/25 ————— 46s 2s/step - accuracy: 0.9845 - loss: 0.0484 - val_accuracy: 0.6300 - val_loss: 3.0083
Epoch 182/200
25/25 ————— 41s 2s/step - accuracy: 0.9851 - loss: 0.0400 - val_accuracy: 0.6050 - val_loss: 3.2633
Epoch 183/200
25/25 ————— 41s 2s/step - accuracy: 0.9809 - loss: 0.0622 - val_accuracy: 0.5850 - val_loss: 2.9806
Epoch 184/200
25/25 ————— 78s 2s/step - accuracy: 0.9896 - loss: 0.0433 - val_accuracy: 0.5200 - val_loss: 4.4966
Epoch 185/200
25/25 ————— 36s 1s/step - accuracy: 0.9712 - loss: 0.0948 - val_accuracy: 0.6250 - val_loss: 3.3453
Epoch 186/200
25/25 ————— 48s 2s/step - accuracy: 0.9821 - loss: 0.0439 - val_accuracy: 0.6100 - val_loss: 3.2421
Epoch 187/200
25/25 ————— 43s 2s/step - accuracy: 0.9888 - loss: 0.0370 - val_accuracy: 0.6150 - val_loss: 3.3000
Epoch 188/200
25/25 ————— 52s 534ms/step - accuracy: 0.9968 - loss: 0.0204 - val_accuracy: 0.6100 - val_loss: 3.2120
Epoch 189/200
25/25 ————— 14s 568ms/step - accuracy: 0.9670 - loss: 0.1086 - val_accuracy: 0.6300 - val_loss: 2.9584
Epoch 190/200
25/25 ————— 14s 542ms/step - accuracy: 0.9895 - loss: 0.0179 - val_accuracy: 0.6400 - val_loss: 3.2522
Epoch 191/200
25/25 ————— 14s 539ms/step - accuracy: 0.9852 - loss: 0.0418 - val_accuracy: 0.6300 - val_loss: 2.9038
Epoch 192/200
25/25 ————— 13s 522ms/step - accuracy: 0.9963 - loss: 0.0141 - val_accuracy: 0.6100 - val_loss: 3.7893
Epoch 193/200
25/25 ————— 14s 549ms/step - accuracy: 0.9896 - loss: 0.0594 - val_accuracy: 0.6050 - val_loss: 3.6565
Epoch 194/200
25/25 ————— 13s 531ms/step - accuracy: 0.9762 - loss: 0.0803 - val_accuracy: 0.6000 - val_loss: 3.0845
Epoch 195/200
25/25 ————— 14s 553ms/step - accuracy: 0.9910 - loss: 0.0324 - val_accuracy: 0.5900 - val_loss: 3.1809
Epoch 196/200
25/25 ————— 13s 531ms/step - accuracy: 0.9763 - loss: 0.0641 - val_accuracy: 0.6000 - val_loss: 3.3820
Epoch 197/200
25/25 ————— 12s 490ms/step - accuracy: 0.9811 - loss: 0.0780 - val_accuracy: 0.6200 - val_loss: 3.3695
Epoch 198/200
25/25 ————— 39s 2s/step - accuracy: 0.9879 - loss: 0.0313 - val_accuracy: 0.6050 - val_loss: 3.4536
Epoch 199/200
25/25 ————— 37s 1s/step - accuracy: 0.9894 - loss: 0.0385 - val_accuracy: 0.6350 - val_loss: 3.3571
Epoch 200/200
25/25 ————— 42s 2s/step - accuracy: 0.9831 - loss: 0.0570 - val_accuracy: 0.6000 - val_loss: 3.3900

```

```
In [8]: print(f"Execution time: {elapsed_time:.2f} seconds")
```

Execution time: 6102.52 seconds

```
In [9]: def append_core_data(score_path, num_cores, elapsed_time):
        # Check if the file already exists
        file_exists = os.path.exists(score_path)

        # Open the file in append mode
        with open(score_path, mode='a', newline='') as file:
```

```
writer = csv.writer(file)

# If the file is new, write the header
if not file_exists:
    writer.writerow(["Number of Cores", "Elapsed Time"])

# Write the new data
writer.writerow([num_cores, elapsed_time])
```

```
In [10]: score_path = r"C:\Users\nikhi\OneDrive\Desktop\Final Project\DEEP LEARNING WITH HPSC\core_data.txt"
append_core_data(score_path, number_of_worker, elapsed_time)
```


Import Req Lib

```
In [1]: %matplotlib inline

import shutil
import random
import numpy as np
from warnings import filterwarnings
filterwarnings('ignore')

from tensorflow.keras import layers, regularizers, optimizers
from tensorflow.keras import models
from tensorflow.keras.models import Sequential, Model
from tensorflow.keras.layers import LeakyReLU, Dense, Activation, Flatten, Dropout, BatchNormalization, Conv2D, MaxPooling2D
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import tensorflow as tf

import os
import time
import csv

from matplotlib import figure
```

Define 2 worker

```
In [2]: # Set the number of threads
number_of_worker = 2
os.environ['OMP_NUM_THREADS'] = '2' # OpenMP threads for parallelism
os.environ['TF_NUM_INTEROP_THREADS'] = '2' # Threads for inter-operation parallelism
os.environ['TF_NUM_INTRAOP_THREADS'] = '2' # Threads for intra-operation parallelism

# Confirm TensorFlow is using the specified number of threads
tf.config.threading.set_inter_op_parallelism_threads(number_of_worker)
tf.config.threading.set_intra_op_parallelism_threads(number_of_worker)
```

Train Val data Split

```
In [3]: source_dir = r"C:\Users\nikhi\OneDrive\Desktop\Final Project\DATA\Convert_Audio_File_to_jpg_file"
target_dir = r'genres_train_val_split_data'
split_ratio = 0.8

def Train_Test_Split(source_dir, target_dir, split_ratio):
    # Define source and target directories
    train_dir = os.path.join(target_dir, 'train')
    val_dir = os.path.join(target_dir, 'val')

    # Create target directories if they don't exist
    os.makedirs(train_dir, exist_ok=True)
    os.makedirs(val_dir, exist_ok=True)

    # Get the list of class directories
    classes = [d for d in os.listdir(source_dir) if os.path.isdir(os.path.join(source_dir, d))]

    for class_name in classes:
        # Create class directories in train and val folders
        os.makedirs(os.path.join(train_dir, class_name), exist_ok=True)
        os.makedirs(os.path.join(val_dir, class_name), exist_ok=True)

        # Get list of images in the class directory
        class_dir = os.path.join(source_dir, class_name)
        images = [f for f in os.listdir(class_dir) if os.path.isfile(os.path.join(class_dir, f))]

        # Shuffle the images
        random.shuffle(images)

        # Compute the split point
        split_point = int(len(images) * split_ratio)

        # Split the images into training and validation sets
        train_images = images[:split_point]
        val_images = images[split_point:]
```



```

# Move the images to the respective directories
for img in train_images:
    shutil.copy(os.path.join(class_dir, img), os.path.join(train_dir, class_name, img))

for img in val_images:
    shutil.copy(os.path.join(class_dir, img), os.path.join(val_dir, class_name, img))

print("Data split completed successfully!")

```

In [4]: `Train_Test_Split(source_dir,target_dir,split_ratio)`

Data split completed successfully!

Load the Data

```

In [5]: WIDTH = 64
HEIGHT = 64
BATCH_SIZE = 32
TRAIN_DIR=r'genres_train_val_split_data/train'
val_dir = r'genres_train_val_split_data/val'

# data prep
train_datagen = ImageDataGenerator(
    rescale=1./255.,validation_split=0.25)

train_generator = train_datagen.flow_from_directory(
    TRAIN_DIR,
    target_size=(HEIGHT, WIDTH),
    batch_size=BATCH_SIZE,
    class_mode='categorical')

validation_gen = train_datagen.flow_from_directory(
    val_dir,target_size = (HEIGHT,WIDTH),
    batch_size = BATCH_SIZE,
    class_mode = 'categorical'
)

```

Found 800 images belonging to 10 classes.

Found 200 images belonging to 10 classes.

Model Architecture

```

In [6]: model = Sequential()
model.add(Conv2D(32, (3, 3), padding='same',
    input_shape=(64,64,3)))
model.add(Activation('relu'))
model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Conv2D(64, (3, 3), padding='same'))
model.add(Activation('relu'))
model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.5))
model.add(Conv2D(128, (3, 3), padding='same'))
model.add(Activation('relu'))
model.add(Conv2D(128, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.5))
model.add(Flatten())
model.add(Dense(512))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(10, activation='softmax'))
model.compile(optimizer=RMSprop(learning_rate=0.0005, decay=1e-6),loss="categorical_crossentropy",metrics=["accuracy"])
model.summary()

```

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 64, 64, 32)	896
activation (Activation)	(None, 64, 64, 32)	0
conv2d_1 (Conv2D)	(None, 62, 62, 64)	18,496
activation_1 (Activation)	(None, 62, 62, 64)	0
max_pooling2d (MaxPooling2D)	(None, 31, 31, 64)	0
dropout (Dropout)	(None, 31, 31, 64)	0
conv2d_2 (Conv2D)	(None, 31, 31, 64)	36,928
activation_2 (Activation)	(None, 31, 31, 64)	0
conv2d_3 (Conv2D)	(None, 29, 29, 64)	36,928
activation_3 (Activation)	(None, 29, 29, 64)	0
max_pooling2d_1 (MaxPooling2D)	(None, 14, 14, 64)	0
dropout_1 (Dropout)	(None, 14, 14, 64)	0
conv2d_4 (Conv2D)	(None, 14, 14, 128)	73,856
activation_4 (Activation)	(None, 14, 14, 128)	0
conv2d_5 (Conv2D)	(None, 12, 12, 128)	147,584
activation_5 (Activation)	(None, 12, 12, 128)	0
max_pooling2d_2 (MaxPooling2D)	(None, 6, 6, 128)	0
dropout_2 (Dropout)	(None, 6, 6, 128)	0
flatten (Flatten)	(None, 4608)	0
dense (Dense)	(None, 512)	2,359,808
activation_6 (Activation)	(None, 512)	0
dropout_3 (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 10)	5,130

Total params: 2,679,626 (10.22 MB)

Trainable params: 2,679,626 (10.22 MB)

Non-trainable params: 0 (0.00 B)

```
In [7]: STEP_SIZE_TRAIN=train_generator.n//train_generator.batch_size
# Measure the execution time
start_time = time.time()

model.fit(train_generator,validation_data=validation_gen,epochs=200)

end_time = time.time()
elapsed_time = end_time - start_time
```

Epoch 1/200	25/25	20s	675ms/step	- accuracy: 0.0658	- loss: 2.3332	- val_accuracy: 0.1000	- val_loss: 2.3021
Epoch 2/200	25/25	17s	693ms/step	- accuracy: 0.0986	- loss: 2.3021	- val_accuracy: 0.1000	- val_loss: 2.2934
Epoch 3/200	25/25	16s	658ms/step	- accuracy: 0.1090	- loss: 2.2960	- val_accuracy: 0.2100	- val_loss: 2.2949
Epoch 4/200	25/25	16s	652ms/step	- accuracy: 0.1633	- loss: 2.2699	- val_accuracy: 0.1900	- val_loss: 2.1670
Epoch 5/200	25/25	15s	607ms/step	- accuracy: 0.2279	- loss: 2.1263	- val_accuracy: 0.2100	- val_loss: 2.0336
Epoch 6/200	25/25	15s	598ms/step	- accuracy: 0.2639	- loss: 2.0327	- val_accuracy: 0.2800	- val_loss: 2.0367
Epoch 7/200	25/25	15s	595ms/step	- accuracy: 0.2955	- loss: 1.9982	- val_accuracy: 0.3250	- val_loss: 1.9601
Epoch 8/200	25/25	15s	594ms/step	- accuracy: 0.2485	- loss: 2.0284	- val_accuracy: 0.2950	- val_loss: 1.9328
Epoch 9/200	25/25	15s	589ms/step	- accuracy: 0.2926	- loss: 1.9276	- val_accuracy: 0.3500	- val_loss: 1.9102
Epoch 10/200	25/25	15s	602ms/step	- accuracy: 0.2897	- loss: 1.9524	- val_accuracy: 0.3400	- val_loss: 1.8158
Epoch 11/200	25/25	16s	650ms/step	- accuracy: 0.3268	- loss: 1.8249	- val_accuracy: 0.3350	- val_loss: 1.7215
Epoch 12/200	25/25	17s	687ms/step	- accuracy: 0.3834	- loss: 1.7091	- val_accuracy: 0.3750	- val_loss: 1.6773
Epoch 13/200	25/25	18s	700ms/step	- accuracy: 0.3712	- loss: 1.7670	- val_accuracy: 0.2500	- val_loss: 2.0661
Epoch 14/200	25/25	18s	699ms/step	- accuracy: 0.3958	- loss: 1.7096	- val_accuracy: 0.3350	- val_loss: 1.6078
Epoch 15/200	25/25	18s	714ms/step	- accuracy: 0.4427	- loss: 1.5865	- val_accuracy: 0.3950	- val_loss: 1.5535
Epoch 16/200	25/25	18s	733ms/step	- accuracy: 0.4159	- loss: 1.5572	- val_accuracy: 0.3500	- val_loss: 1.8125
Epoch 17/200	25/25	19s	741ms/step	- accuracy: 0.4032	- loss: 1.6410	- val_accuracy: 0.4050	- val_loss: 1.5667
Epoch 18/200	25/25	18s	733ms/step	- accuracy: 0.4755	- loss: 1.4892	- val_accuracy: 0.4050	- val_loss: 1.5534
Epoch 19/200	25/25	18s	721ms/step	- accuracy: 0.4703	- loss: 1.4501	- val_accuracy: 0.4250	- val_loss: 1.5390
Epoch 20/200	25/25	18s	715ms/step	- accuracy: 0.4298	- loss: 1.5520	- val_accuracy: 0.3700	- val_loss: 1.6879
Epoch 21/200	25/25	18s	727ms/step	- accuracy: 0.4590	- loss: 1.4803	- val_accuracy: 0.4050	- val_loss: 1.5830
Epoch 22/200	25/25	18s	713ms/step	- accuracy: 0.4832	- loss: 1.4678	- val_accuracy: 0.4850	- val_loss: 1.3881
Epoch 23/200	25/25	18s	721ms/step	- accuracy: 0.5379	- loss: 1.2982	- val_accuracy: 0.4600	- val_loss: 1.4611
Epoch 24/200	25/25	18s	734ms/step	- accuracy: 0.5154	- loss: 1.3152	- val_accuracy: 0.4250	- val_loss: 1.4089
Epoch 25/200	25/25	18s	724ms/step	- accuracy: 0.5432	- loss: 1.2727	- val_accuracy: 0.5100	- val_loss: 1.3474
Epoch 26/200	25/25	18s	719ms/step	- accuracy: 0.5513	- loss: 1.2137	- val_accuracy: 0.5150	- val_loss: 1.3525
Epoch 27/200	25/25	18s	722ms/step	- accuracy: 0.5891	- loss: 1.1408	- val_accuracy: 0.5200	- val_loss: 1.3607
Epoch 28/200	25/25	18s	713ms/step	- accuracy: 0.5416	- loss: 1.1971	- val_accuracy: 0.4800	- val_loss: 1.4404
Epoch 29/200	25/25	18s	738ms/step	- accuracy: 0.5836	- loss: 1.1399	- val_accuracy: 0.5000	- val_loss: 1.3671
Epoch 30/200	25/25	18s	735ms/step	- accuracy: 0.5671	- loss: 1.1366	- val_accuracy: 0.5200	- val_loss: 1.3093
Epoch 31/200	25/25	19s	751ms/step	- accuracy: 0.6065	- loss: 1.0926	- val_accuracy: 0.5100	- val_loss: 1.2397
Epoch 32/200	25/25	18s	737ms/step	- accuracy: 0.5776	- loss: 1.0719	- val_accuracy: 0.5250	- val_loss: 1.3024
Epoch 33/200	25/25	18s	707ms/step	- accuracy: 0.6042	- loss: 1.0833	- val_accuracy: 0.5250	- val_loss: 1.3458
Epoch 34/200	25/25	17s	690ms/step	- accuracy: 0.5982	- loss: 1.1189	- val_accuracy: 0.5650	- val_loss: 1.2984
Epoch 35/200	25/25	18s	715ms/step	- accuracy: 0.6693	- loss: 0.9085	- val_accuracy: 0.5400	- val_loss: 1.2657
Epoch 36/200	25/25	18s	735ms/step	- accuracy: 0.6650	- loss: 0.9373	- val_accuracy: 0.5600	- val_loss: 1.1906
Epoch 37/200	25/25	18s	730ms/step	- accuracy: 0.6666	- loss: 0.9179	- val_accuracy: 0.5850	- val_loss: 1.1938
Epoch 38/200	25/25	18s	735ms/step	- accuracy: 0.6960	- loss: 0.8488	- val_accuracy: 0.6050	- val_loss: 1.1426
Epoch 39/200	25/25	18s	713ms/step	- accuracy: 0.6970	- loss: 0.8214	- val_accuracy: 0.5650	- val_loss: 1.2680
Epoch 40/200	25/25	18s	712ms/step	- accuracy: 0.6785	- loss: 0.8729	- val_accuracy: 0.5750	- val_loss: 1.1458
Epoch 41/200	25/25	18s	717ms/step	- accuracy: 0.6635	- loss: 0.9008	- val_accuracy: 0.6000	- val_loss: 1.1763

Epoch 42/200	25/25	18s	707ms/step	- accuracy: 0.6824	- loss: 0.8254	- val_accuracy: 0.5600	- val_loss: 1.1993
Epoch 43/200	25/25	18s	734ms/step	- accuracy: 0.7287	- loss: 0.8137	- val_accuracy: 0.6000	- val_loss: 1.1503
Epoch 44/200	25/25	18s	725ms/step	- accuracy: 0.7484	- loss: 0.6776	- val_accuracy: 0.5950	- val_loss: 1.1720
Epoch 45/200	25/25	18s	730ms/step	- accuracy: 0.7591	- loss: 0.7487	- val_accuracy: 0.5900	- val_loss: 1.2234
Epoch 46/200	25/25	18s	708ms/step	- accuracy: 0.7634	- loss: 0.6756	- val_accuracy: 0.5950	- val_loss: 1.1382
Epoch 47/200	25/25	18s	721ms/step	- accuracy: 0.7567	- loss: 0.6095	- val_accuracy: 0.6150	- val_loss: 1.2420
Epoch 48/200	25/25	18s	710ms/step	- accuracy: 0.7541	- loss: 0.6675	- val_accuracy: 0.6150	- val_loss: 1.2987
Epoch 49/200	25/25	18s	735ms/step	- accuracy: 0.7979	- loss: 0.5888	- val_accuracy: 0.6400	- val_loss: 1.2228
Epoch 50/200	25/25	18s	731ms/step	- accuracy: 0.7945	- loss: 0.5587	- val_accuracy: 0.5650	- val_loss: 1.4477
Epoch 51/200	25/25	18s	728ms/step	- accuracy: 0.8001	- loss: 0.5551	- val_accuracy: 0.6050	- val_loss: 1.4301
Epoch 52/200	25/25	18s	728ms/step	- accuracy: 0.7869	- loss: 0.6716	- val_accuracy: 0.5750	- val_loss: 1.3105
Epoch 53/200	25/25	18s	718ms/step	- accuracy: 0.7919	- loss: 0.5782	- val_accuracy: 0.6300	- val_loss: 1.2621
Epoch 54/200	25/25	18s	719ms/step	- accuracy: 0.8451	- loss: 0.4805	- val_accuracy: 0.6350	- val_loss: 1.2906
Epoch 55/200	25/25	18s	720ms/step	- accuracy: 0.8179	- loss: 0.4926	- val_accuracy: 0.6200	- val_loss: 1.4216
Epoch 56/200	25/25	18s	725ms/step	- accuracy: 0.8320	- loss: 0.4691	- val_accuracy: 0.6300	- val_loss: 1.4858
Epoch 57/200	25/25	20s	718ms/step	- accuracy: 0.8099	- loss: 0.4724	- val_accuracy: 0.6100	- val_loss: 1.5430
Epoch 58/200	25/25	18s	728ms/step	- accuracy: 0.8482	- loss: 0.4111	- val_accuracy: 0.5950	- val_loss: 1.5319
Epoch 59/200	25/25	18s	708ms/step	- accuracy: 0.8596	- loss: 0.3476	- val_accuracy: 0.6100	- val_loss: 1.6917
Epoch 60/200	25/25	18s	709ms/step	- accuracy: 0.8134	- loss: 0.4908	- val_accuracy: 0.6200	- val_loss: 1.5236
Epoch 61/200	25/25	18s	715ms/step	- accuracy: 0.8909	- loss: 0.3588	- val_accuracy: 0.6200	- val_loss: 1.4370
Epoch 62/200	25/25	18s	713ms/step	- accuracy: 0.8414	- loss: 0.4690	- val_accuracy: 0.6300	- val_loss: 1.7034
Epoch 63/200	25/25	21s	716ms/step	- accuracy: 0.8408	- loss: 0.5160	- val_accuracy: 0.5900	- val_loss: 1.6704
Epoch 64/200	25/25	18s	722ms/step	- accuracy: 0.8854	- loss: 0.3459	- val_accuracy: 0.6050	- val_loss: 1.6405
Epoch 65/200	25/25	18s	723ms/step	- accuracy: 0.9229	- loss: 0.2495	- val_accuracy: 0.6100	- val_loss: 1.6316
Epoch 66/200	25/25	20s	720ms/step	- accuracy: 0.8997	- loss: 0.2830	- val_accuracy: 0.6150	- val_loss: 1.5437
Epoch 67/200	25/25	18s	713ms/step	- accuracy: 0.9092	- loss: 0.2726	- val_accuracy: 0.6000	- val_loss: 1.7793
Epoch 68/200	25/25	18s	715ms/step	- accuracy: 0.9390	- loss: 0.1926	- val_accuracy: 0.5800	- val_loss: 2.0801
Epoch 69/200	25/25	18s	723ms/step	- accuracy: 0.9058	- loss: 0.3264	- val_accuracy: 0.6200	- val_loss: 1.8446
Epoch 70/200	25/25	18s	733ms/step	- accuracy: 0.8686	- loss: 0.4121	- val_accuracy: 0.6000	- val_loss: 1.9129
Epoch 71/200	25/25	18s	730ms/step	- accuracy: 0.8806	- loss: 0.3671	- val_accuracy: 0.6050	- val_loss: 1.6417
Epoch 72/200	25/25	18s	705ms/step	- accuracy: 0.9386	- loss: 0.1723	- val_accuracy: 0.5900	- val_loss: 1.7384
Epoch 73/200	25/25	18s	715ms/step	- accuracy: 0.9273	- loss: 0.2003	- val_accuracy: 0.5850	- val_loss: 2.1487
Epoch 74/200	25/25	18s	714ms/step	- accuracy: 0.9156	- loss: 0.2440	- val_accuracy: 0.6000	- val_loss: 1.9785
Epoch 75/200	25/25	21s	729ms/step	- accuracy: 0.9193	- loss: 0.2718	- val_accuracy: 0.5850	- val_loss: 2.0705
Epoch 76/200	25/25	18s	724ms/step	- accuracy: 0.9459	- loss: 0.1662	- val_accuracy: 0.6100	- val_loss: 2.0769
Epoch 77/200	25/25	18s	726ms/step	- accuracy: 0.9097	- loss: 0.2411	- val_accuracy: 0.6250	- val_loss: 2.1611
Epoch 78/200	25/25	18s	733ms/step	- accuracy: 0.9575	- loss: 0.1431	- val_accuracy: 0.5850	- val_loss: 2.3373
Epoch 79/200	25/25	18s	722ms/step	- accuracy: 0.9352	- loss: 0.2195	- val_accuracy: 0.6050	- val_loss: 2.0010
Epoch 80/200	25/25	18s	707ms/step	- accuracy: 0.9505	- loss: 0.1423	- val_accuracy: 0.5950	- val_loss: 1.8676
Epoch 81/200	25/25	18s	708ms/step	- accuracy: 0.9476	- loss: 0.1579	- val_accuracy: 0.5950	- val_loss: 2.0369
Epoch 82/200	25/25	18s	723ms/step	- accuracy: 0.9688	- loss: 0.1092	- val_accuracy: 0.6150	- val_loss: 1.9845

Epoch 83/200	25/25	18s	731ms/step	- accuracy: 0.9479	- loss: 0.1606	- val_accuracy: 0.5800	- val_loss: 2.6820
Epoch 84/200	25/25	18s	726ms/step	- accuracy: 0.9545	- loss: 0.1623	- val_accuracy: 0.6100	- val_loss: 1.9176
Epoch 85/200	25/25	18s	706ms/step	- accuracy: 0.9567	- loss: 0.1448	- val_accuracy: 0.6100	- val_loss: 2.0825
Epoch 86/200	25/25	18s	710ms/step	- accuracy: 0.9462	- loss: 0.1407	- val_accuracy: 0.6050	- val_loss: 2.2098
Epoch 87/200	25/25	18s	721ms/step	- accuracy: 0.9378	- loss: 0.1713	- val_accuracy: 0.5900	- val_loss: 2.2017
Epoch 88/200	25/25	18s	716ms/step	- accuracy: 0.9610	- loss: 0.1053	- val_accuracy: 0.6200	- val_loss: 2.2934
Epoch 89/200	25/25	18s	728ms/step	- accuracy: 0.9620	- loss: 0.1244	- val_accuracy: 0.6000	- val_loss: 2.2978
Epoch 90/200	25/25	18s	718ms/step	- accuracy: 0.9526	- loss: 0.1334	- val_accuracy: 0.5850	- val_loss: 2.2852
Epoch 91/200	25/25	18s	730ms/step	- accuracy: 0.9629	- loss: 0.1154	- val_accuracy: 0.6000	- val_loss: 2.0158
Epoch 92/200	25/25	18s	708ms/step	- accuracy: 0.9730	- loss: 0.0986	- val_accuracy: 0.6100	- val_loss: 2.1760
Epoch 93/200	25/25	18s	716ms/step	- accuracy: 0.9814	- loss: 0.0740	- val_accuracy: 0.5800	- val_loss: 2.4375
Epoch 94/200	25/25	18s	703ms/step	- accuracy: 0.9482	- loss: 0.1256	- val_accuracy: 0.6200	- val_loss: 2.1586
Epoch 95/200	25/25	18s	715ms/step	- accuracy: 0.9786	- loss: 0.0733	- val_accuracy: 0.6100	- val_loss: 2.1589
Epoch 96/200	25/25	18s	731ms/step	- accuracy: 0.9680	- loss: 0.0907	- val_accuracy: 0.5900	- val_loss: 2.4064
Epoch 97/200	25/25	19s	754ms/step	- accuracy: 0.9696	- loss: 0.0912	- val_accuracy: 0.5600	- val_loss: 2.4183
Epoch 98/200	25/25	18s	733ms/step	- accuracy: 0.9788	- loss: 0.0655	- val_accuracy: 0.6050	- val_loss: 2.2126
Epoch 99/200	25/25	20s	722ms/step	- accuracy: 0.9470	- loss: 0.1776	- val_accuracy: 0.6300	- val_loss: 2.1396
Epoch 100/200	25/25	18s	709ms/step	- accuracy: 0.9711	- loss: 0.0763	- val_accuracy: 0.6050	- val_loss: 2.0356
Epoch 101/200	25/25	18s	724ms/step	- accuracy: 0.9650	- loss: 0.1307	- val_accuracy: 0.6050	- val_loss: 2.1148
Epoch 102/200	25/25	18s	724ms/step	- accuracy: 0.9790	- loss: 0.0682	- val_accuracy: 0.6000	- val_loss: 2.6045
Epoch 103/200	25/25	18s	718ms/step	- accuracy: 0.9743	- loss: 0.0784	- val_accuracy: 0.5750	- val_loss: 2.6396
Epoch 104/200	25/25	18s	711ms/step	- accuracy: 0.9579	- loss: 0.1212	- val_accuracy: 0.5750	- val_loss: 2.9462
Epoch 105/200	25/25	18s	719ms/step	- accuracy: 0.9851	- loss: 0.0671	- val_accuracy: 0.6000	- val_loss: 2.5684
Epoch 106/200	25/25	18s	726ms/step	- accuracy: 0.9438	- loss: 0.2108	- val_accuracy: 0.5700	- val_loss: 2.1696
Epoch 107/200	25/25	18s	720ms/step	- accuracy: 0.9562	- loss: 0.1139	- val_accuracy: 0.5850	- val_loss: 2.4554
Epoch 108/200	25/25	18s	724ms/step	- accuracy: 0.9572	- loss: 0.1481	- val_accuracy: 0.6150	- val_loss: 2.1096
Epoch 109/200	25/25	18s	713ms/step	- accuracy: 0.9787	- loss: 0.0589	- val_accuracy: 0.5750	- val_loss: 3.2846
Epoch 110/200	25/25	18s	714ms/step	- accuracy: 0.9564	- loss: 0.1865	- val_accuracy: 0.6050	- val_loss: 2.1813
Epoch 111/200	25/25	18s	717ms/step	- accuracy: 0.9763	- loss: 0.0739	- val_accuracy: 0.5700	- val_loss: 2.6695
Epoch 112/200	25/25	18s	712ms/step	- accuracy: 0.9603	- loss: 0.1191	- val_accuracy: 0.6050	- val_loss: 2.3246
Epoch 113/200	25/25	18s	713ms/step	- accuracy: 0.9768	- loss: 0.0602	- val_accuracy: 0.6150	- val_loss: 2.4997
Epoch 114/200	25/25	17s	696ms/step	- accuracy: 0.9840	- loss: 0.0772	- val_accuracy: 0.6000	- val_loss: 2.7861
Epoch 115/200	25/25	18s	716ms/step	- accuracy: 0.9773	- loss: 0.1082	- val_accuracy: 0.6150	- val_loss: 2.3787
Epoch 116/200	25/25	18s	720ms/step	- accuracy: 0.9836	- loss: 0.0524	- val_accuracy: 0.6000	- val_loss: 2.9112
Epoch 117/200	25/25	18s	721ms/step	- accuracy: 0.9603	- loss: 0.1101	- val_accuracy: 0.5900	- val_loss: 2.7486
Epoch 118/200	25/25	18s	710ms/step	- accuracy: 0.9809	- loss: 0.0785	- val_accuracy: 0.5950	- val_loss: 2.8456
Epoch 119/200	25/25	17s	702ms/step	- accuracy: 0.9673	- loss: 0.1098	- val_accuracy: 0.5950	- val_loss: 2.6416
Epoch 120/200	25/25	18s	701ms/step	- accuracy: 0.9882	- loss: 0.0370	- val_accuracy: 0.6100	- val_loss: 2.6769
Epoch 121/200	25/25	18s	712ms/step	- accuracy: 0.9638	- loss: 0.1044	- val_accuracy: 0.6200	- val_loss: 2.9083
Epoch 122/200	25/25	18s	725ms/step	- accuracy: 0.9782	- loss: 0.0979	- val_accuracy: 0.6200	- val_loss: 2.5686
Epoch 123/200	25/25	18s	729ms/step	- accuracy: 0.9812	- loss: 0.0578	- val_accuracy: 0.6250	- val_loss: 2.3114

[illegible]


```

Epoch 165/200
25/25 ----- 18s 726ms/step - accuracy: 0.9907 - loss: 0.0331 - val_accuracy: 0.6150 - val_loss: 3.0317
Epoch 166/200
25/25 ----- 17s 688ms/step - accuracy: 0.9721 - loss: 0.0709 - val_accuracy: 0.6100 - val_loss: 3.0954
Epoch 167/200
25/25 ----- 18s 713ms/step - accuracy: 0.9757 - loss: 0.0530 - val_accuracy: 0.6050 - val_loss: 3.2853
Epoch 168/200
25/25 ----- 18s 706ms/step - accuracy: 0.9767 - loss: 0.0868 - val_accuracy: 0.6100 - val_loss: 2.8212
Epoch 169/200
25/25 ----- 18s 721ms/step - accuracy: 0.9894 - loss: 0.0469 - val_accuracy: 0.6000 - val_loss: 3.0674
Epoch 170/200
25/25 ----- 18s 718ms/step - accuracy: 0.9883 - loss: 0.0524 - val_accuracy: 0.5900 - val_loss: 2.7765
Epoch 171/200
25/25 ----- 18s 714ms/step - accuracy: 0.9719 - loss: 0.0858 - val_accuracy: 0.5950 - val_loss: 2.9619
Epoch 172/200
25/25 ----- 18s 723ms/step - accuracy: 0.9967 - loss: 0.0159 - val_accuracy: 0.5950 - val_loss: 3.2934
Epoch 173/200
25/25 ----- 18s 702ms/step - accuracy: 0.9748 - loss: 0.0891 - val_accuracy: 0.5900 - val_loss: 2.9908
Epoch 174/200
25/25 ----- 17s 694ms/step - accuracy: 0.9945 - loss: 0.0353 - val_accuracy: 0.6050 - val_loss: 3.2003
Epoch 175/200
25/25 ----- 18s 710ms/step - accuracy: 0.9945 - loss: 0.0238 - val_accuracy: 0.5550 - val_loss: 5.2057
Epoch 176/200
25/25 ----- 18s 728ms/step - accuracy: 0.9718 - loss: 0.1247 - val_accuracy: 0.5800 - val_loss: 3.1216
Epoch 177/200
25/25 ----- 18s 718ms/step - accuracy: 0.9788 - loss: 0.0614 - val_accuracy: 0.6100 - val_loss: 3.3939
Epoch 178/200
25/25 ----- 18s 729ms/step - accuracy: 0.9917 - loss: 0.0176 - val_accuracy: 0.5800 - val_loss: 3.5900
Epoch 179/200
25/25 ----- 18s 720ms/step - accuracy: 0.9924 - loss: 0.0369 - val_accuracy: 0.6100 - val_loss: 3.5684
Epoch 180/200
25/25 ----- 18s 716ms/step - accuracy: 0.9919 - loss: 0.0242 - val_accuracy: 0.6100 - val_loss: 3.8176
Epoch 181/200
25/25 ----- 18s 703ms/step - accuracy: 0.9974 - loss: 0.0112 - val_accuracy: 0.6200 - val_loss: 3.5469
Epoch 182/200
25/25 ----- 18s 722ms/step - accuracy: 0.9864 - loss: 0.0510 - val_accuracy: 0.6000 - val_loss: 3.8091
Epoch 183/200
25/25 ----- 18s 723ms/step - accuracy: 0.9632 - loss: 0.1131 - val_accuracy: 0.5950 - val_loss: 3.4164
Epoch 184/200
25/25 ----- 18s 732ms/step - accuracy: 0.9814 - loss: 0.1048 - val_accuracy: 0.5850 - val_loss: 2.7835
Epoch 185/200
25/25 ----- 18s 718ms/step - accuracy: 0.9960 - loss: 0.0176 - val_accuracy: 0.5900 - val_loss: 3.4109
Epoch 186/200
25/25 ----- 18s 710ms/step - accuracy: 0.9937 - loss: 0.0333 - val_accuracy: 0.5950 - val_loss: 3.6571
Epoch 187/200
25/25 ----- 17s 681ms/step - accuracy: 0.9916 - loss: 0.0364 - val_accuracy: 0.6050 - val_loss: 3.6589
Epoch 188/200
25/25 ----- 18s 714ms/step - accuracy: 0.9962 - loss: 0.0138 - val_accuracy: 0.6000 - val_loss: 3.2003
Epoch 189/200
25/25 ----- 18s 723ms/step - accuracy: 0.9920 - loss: 0.0247 - val_accuracy: 0.5800 - val_loss: 3.8158
Epoch 190/200
25/25 ----- 18s 718ms/step - accuracy: 0.9831 - loss: 0.0737 - val_accuracy: 0.6250 - val_loss: 3.4735
Epoch 191/200
25/25 ----- 18s 719ms/step - accuracy: 0.9852 - loss: 0.0763 - val_accuracy: 0.6000 - val_loss: 3.5289
Epoch 192/200
25/25 ----- 18s 706ms/step - accuracy: 0.9861 - loss: 0.0350 - val_accuracy: 0.6000 - val_loss: 3.7007
Epoch 193/200
25/25 ----- 17s 697ms/step - accuracy: 0.9921 - loss: 0.0248 - val_accuracy: 0.6050 - val_loss: 3.6667
Epoch 194/200
25/25 ----- 18s 708ms/step - accuracy: 0.9772 - loss: 0.0782 - val_accuracy: 0.6200 - val_loss: 4.1763
Epoch 195/200
25/25 ----- 20s 703ms/step - accuracy: 0.9855 - loss: 0.0633 - val_accuracy: 0.6000 - val_loss: 3.3390
Epoch 196/200
25/25 ----- 18s 710ms/step - accuracy: 0.9920 - loss: 0.0385 - val_accuracy: 0.5800 - val_loss: 5.2956
Epoch 197/200
25/25 ----- 18s 703ms/step - accuracy: 0.9785 - loss: 0.0978 - val_accuracy: 0.5950 - val_loss: 4.0987
Epoch 198/200
25/25 ----- 18s 729ms/step - accuracy: 0.9750 - loss: 0.0530 - val_accuracy: 0.6100 - val_loss: 4.0447
Epoch 199/200
25/25 ----- 18s 705ms/step - accuracy: 0.9891 - loss: 0.0457 - val_accuracy: 0.5850 - val_loss: 3.7655
Epoch 200/200
25/25 ----- 17s 695ms/step - accuracy: 0.9887 - loss: 0.0432 - val_accuracy: 0.5750 - val_loss: 4.1484

```

```
In [8]: print(f"Execution time: {elapsed_time:.2f} seconds")
```

Execution time: 3590.61 seconds

```
In [9]: def append_core_data(score_path, num_cores, elapsed_time):
        # Check if the file already exists
        file_exists = os.path.exists(score_path)

        # Open the file in append mode
        with open(score_path, mode='a', newline='') as file:
```

```
writer = csv.writer(file)

# If the file is new, write the header
if not file_exists:
    writer.writerow(["Number of Cores", "Elapsed Time"])

# Write the new data
writer.writerow([num_cores, elapsed_time])
```

```
In [10]: score_path = r"C:\Users\nikhi\OneDrive\Desktop\Final Project\DEEP LEARNING WITH HPSC\core_data.txt"
append_core_data(score_path, number_of_worker, elapsed_time)
```


Import Req Lib

```
In [1]: %matplotlib inline

import shutil
import random
import numpy as np
from warnings import filterwarnings
filterwarnings('ignore')

from tensorflow.keras import layers, regularizers, optimizers
from tensorflow.keras import models
from tensorflow.keras.models import Sequential, Model
from tensorflow.keras.layers import LeakyReLU, Dense, Activation, Flatten, Dropout, BatchNormalization, Conv2D, MaxPooling2D
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import tensorflow as tf

import os
import time
import csv

from matplotlib import figure
```

Define 3 worker

```
In [2]: # Set the number of threads
number_of_worker = 3
os.environ['OMP_NUM_THREADS'] = '3' # OpenMP threads for parallelism
os.environ['TF_NUM_INTEROP_THREADS'] = '3' # Threads for inter-operation parallelism
os.environ['TF_NUM_INTRAOP_THREADS'] = '3' # Threads for intra-operation parallelism

# Confirm TensorFlow is using the specified number of threads
tf.config.threading.set_inter_op_parallelism_threads(number_of_worker)
tf.config.threading.set_intra_op_parallelism_threads(number_of_worker)
```

Train Val data Split

```
In [3]: source_dir = r"C:\Users\nikhi\OneDrive\Desktop\Final Project\DATA\Convert_Audio_File_to_jpg_file"
target_dir = r'genres_train_val_split_data'
split_ratio = 0.8

def Train_Test_Split(source_dir, target_dir, split_ratio):
    # Define source and target directories
    train_dir = os.path.join(target_dir, 'train')
    val_dir = os.path.join(target_dir, 'val')

    # Create target directories if they don't exist
    os.makedirs(train_dir, exist_ok=True)
    os.makedirs(val_dir, exist_ok=True)

    # Get the list of class directories
    classes = [d for d in os.listdir(source_dir) if os.path.isdir(os.path.join(source_dir, d))]

    for class_name in classes:
        # Create class directories in train and val folders
        os.makedirs(os.path.join(train_dir, class_name), exist_ok=True)
        os.makedirs(os.path.join(val_dir, class_name), exist_ok=True)

        # Get list of images in the class directory
        class_dir = os.path.join(source_dir, class_name)
        images = [f for f in os.listdir(class_dir) if os.path.isfile(os.path.join(class_dir, f))]

        # Shuffle the images
        random.shuffle(images)

        # Compute the split point
        split_point = int(len(images) * split_ratio)

        # Split the images into training and validation sets
        train_images = images[:split_point]
        val_images = images[split_point:]
```

```

# Move the images to the respective directories
for img in train_images:
    shutil.copy(os.path.join(class_dir, img), os.path.join(train_dir, class_name, img))

for img in val_images:
    shutil.copy(os.path.join(class_dir, img), os.path.join(val_dir, class_name, img))

print("Data split completed successfully!")

```

In [4]: `Train_Test_Split(source_dir,target_dir,split_ratio)`

Data split completed successfully!

Load the Data

```

In [5]: WIDTH = 64
HEIGHT = 64
BATCH_SIZE = 32
TRAIN_DIR=r'genres_train_val_split_data/train'
val_dir = r'genres_train_val_split_data/val'

# data prep
train_datagen = ImageDataGenerator(
    rescale=1./255.,validation_split=0.25)

train_generator = train_datagen.flow_from_directory(
    TRAIN_DIR,
    target_size=(HEIGHT, WIDTH),
    batch_size=BATCH_SIZE,
    class_mode='categorical')

validation_gen = train_datagen.flow_from_directory(
    val_dir,target_size = (HEIGHT,WIDTH),
    batch_size = BATCH_SIZE,
    class_mode = 'categorical'
)

```

Found 800 images belonging to 10 classes.

Found 200 images belonging to 10 classes.

Model Architecture

```

In [6]: model = Sequential()
model.add(Conv2D(32, (3, 3), padding='same',
    input_shape=(64,64,3)))
model.add(Activation('relu'))
model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Conv2D(64, (3, 3), padding='same'))
model.add(Activation('relu'))
model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.5))
model.add(Conv2D(128, (3, 3), padding='same'))
model.add(Activation('relu'))
model.add(Conv2D(128, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.5))
model.add(Flatten())
model.add(Dense(512))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(10, activation='softmax'))
model.compile(optimizer=RMSprop(learning_rate=0.0005, decay=1e-6),loss="categorical_crossentropy",metrics=["accuracy"])
model.summary()

```

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 64, 64, 32)	896
activation (Activation)	(None, 64, 64, 32)	0
conv2d_1 (Conv2D)	(None, 62, 62, 64)	18,496
activation_1 (Activation)	(None, 62, 62, 64)	0
max_pooling2d (MaxPooling2D)	(None, 31, 31, 64)	0
dropout (Dropout)	(None, 31, 31, 64)	0
conv2d_2 (Conv2D)	(None, 31, 31, 64)	36,928
activation_2 (Activation)	(None, 31, 31, 64)	0
conv2d_3 (Conv2D)	(None, 29, 29, 64)	36,928
activation_3 (Activation)	(None, 29, 29, 64)	0
max_pooling2d_1 (MaxPooling2D)	(None, 14, 14, 64)	0
dropout_1 (Dropout)	(None, 14, 14, 64)	0
conv2d_4 (Conv2D)	(None, 14, 14, 128)	73,856
activation_4 (Activation)	(None, 14, 14, 128)	0
conv2d_5 (Conv2D)	(None, 12, 12, 128)	147,584
activation_5 (Activation)	(None, 12, 12, 128)	0
max_pooling2d_2 (MaxPooling2D)	(None, 6, 6, 128)	0
dropout_2 (Dropout)	(None, 6, 6, 128)	0
flatten (Flatten)	(None, 4608)	0
dense (Dense)	(None, 512)	2,359,808
activation_6 (Activation)	(None, 512)	0
dropout_3 (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 10)	5,130

Total params: 2,679,626 (10.22 MB)

Trainable params: 2,679,626 (10.22 MB)

Non-trainable params: 0 (0.00 B)

```
In [7]: STEP_SIZE_TRAIN=train_generator.n//train_generator.batch_size
# Measure the execution time
start_time = time.time()

model.fit(train_generator,validation_data=validation_gen,epochs=200)

end_time = time.time()
elapsed_time = end_time - start_time
```

[illegible]

Epoch 42/200	25/25	11s	438ms/step	- accuracy: 0.7672	- loss: 0.7296	- val_accuracy: 0.4550	- val_loss: 1.7125
Epoch 43/200	25/25	11s	445ms/step	- accuracy: 0.7301	- loss: 0.7281	- val_accuracy: 0.4950	- val_loss: 1.5391
Epoch 44/200	25/25	11s	458ms/step	- accuracy: 0.7694	- loss: 0.6188	- val_accuracy: 0.5400	- val_loss: 1.4922
Epoch 45/200	25/25	11s	451ms/step	- accuracy: 0.7854	- loss: 0.6459	- val_accuracy: 0.5050	- val_loss: 1.6969
Epoch 46/200	25/25	13s	502ms/step	- accuracy: 0.7587	- loss: 0.7253	- val_accuracy: 0.5800	- val_loss: 1.4690
Epoch 47/200	25/25	12s	486ms/step	- accuracy: 0.8226	- loss: 0.5347	- val_accuracy: 0.5500	- val_loss: 1.4696
Epoch 48/200	25/25	12s	494ms/step	- accuracy: 0.8037	- loss: 0.5382	- val_accuracy: 0.5500	- val_loss: 1.4454
Epoch 49/200	25/25	12s	464ms/step	- accuracy: 0.8286	- loss: 0.4913	- val_accuracy: 0.5650	- val_loss: 1.3963
Epoch 50/200	25/25	10s	408ms/step	- accuracy: 0.8572	- loss: 0.4218	- val_accuracy: 0.5700	- val_loss: 1.5492
Epoch 51/200	25/25	11s	439ms/step	- accuracy: 0.8326	- loss: 0.4634	- val_accuracy: 0.5250	- val_loss: 1.8723
Epoch 52/200	25/25	12s	462ms/step	- accuracy: 0.8203	- loss: 0.5689	- val_accuracy: 0.5500	- val_loss: 1.6347
Epoch 53/200	25/25	13s	499ms/step	- accuracy: 0.8558	- loss: 0.4413	- val_accuracy: 0.5250	- val_loss: 1.7110
Epoch 54/200	25/25	13s	517ms/step	- accuracy: 0.8577	- loss: 0.4649	- val_accuracy: 0.5750	- val_loss: 1.5534
Epoch 55/200	25/25	13s	510ms/step	- accuracy: 0.8622	- loss: 0.3862	- val_accuracy: 0.5800	- val_loss: 1.4429
Epoch 56/200	25/25	13s	500ms/step	- accuracy: 0.8415	- loss: 0.4130	- val_accuracy: 0.5600	- val_loss: 1.6298
Epoch 57/200	25/25	12s	473ms/step	- accuracy: 0.8331	- loss: 0.4591	- val_accuracy: 0.5550	- val_loss: 1.7618
Epoch 58/200	25/25	13s	512ms/step	- accuracy: 0.8447	- loss: 0.4063	- val_accuracy: 0.5600	- val_loss: 1.7022
Epoch 59/200	25/25	13s	497ms/step	- accuracy: 0.8745	- loss: 0.3720	- val_accuracy: 0.5500	- val_loss: 1.7594
Epoch 60/200	25/25	12s	484ms/step	- accuracy: 0.8637	- loss: 0.3368	- val_accuracy: 0.5600	- val_loss: 1.4790
Epoch 61/200	25/25	12s	458ms/step	- accuracy: 0.9153	- loss: 0.2569	- val_accuracy: 0.5450	- val_loss: 1.6113
Epoch 62/200	25/25	12s	471ms/step	- accuracy: 0.9336	- loss: 0.2270	- val_accuracy: 0.5000	- val_loss: 2.4138
Epoch 63/200	25/25	11s	418ms/step	- accuracy: 0.9275	- loss: 0.2432	- val_accuracy: 0.5550	- val_loss: 2.1046
Epoch 64/200	25/25	11s	426ms/step	- accuracy: 0.9020	- loss: 0.2548	- val_accuracy: 0.5650	- val_loss: 1.9601
Epoch 65/200	25/25	11s	445ms/step	- accuracy: 0.9056	- loss: 0.2923	- val_accuracy: 0.6000	- val_loss: 1.7826
Epoch 66/200	25/25	11s	419ms/step	- accuracy: 0.9148	- loss: 0.2506	- val_accuracy: 0.5550	- val_loss: 1.8186
Epoch 67/200	25/25	11s	441ms/step	- accuracy: 0.9171	- loss: 0.2525	- val_accuracy: 0.5900	- val_loss: 1.8137
Epoch 68/200	25/25	11s	447ms/step	- accuracy: 0.9299	- loss: 0.2053	- val_accuracy: 0.5550	- val_loss: 2.3692
Epoch 69/200	25/25	11s	448ms/step	- accuracy: 0.9197	- loss: 0.2721	- val_accuracy: 0.5500	- val_loss: 2.1923
Epoch 70/200	25/25	12s	459ms/step	- accuracy: 0.9512	- loss: 0.1595	- val_accuracy: 0.5800	- val_loss: 2.3808
Epoch 71/200	25/25	12s	480ms/step	- accuracy: 0.9184	- loss: 0.2301	- val_accuracy: 0.5500	- val_loss: 2.0233
Epoch 72/200	25/25	13s	522ms/step	- accuracy: 0.9068	- loss: 0.2479	- val_accuracy: 0.5700	- val_loss: 2.1736
Epoch 73/200	25/25	12s	479ms/step	- accuracy: 0.9345	- loss: 0.1671	- val_accuracy: 0.5800	- val_loss: 1.8012
Epoch 74/200	25/25	12s	466ms/step	- accuracy: 0.9508	- loss: 0.1823	- val_accuracy: 0.5800	- val_loss: 1.9842
Epoch 75/200	25/25	11s	450ms/step	- accuracy: 0.9300	- loss: 0.2190	- val_accuracy: 0.5750	- val_loss: 2.5367
Epoch 76/200	25/25	12s	464ms/step	- accuracy: 0.9385	- loss: 0.1612	- val_accuracy: 0.5600	- val_loss: 2.3644
Epoch 77/200	25/25	12s	486ms/step	- accuracy: 0.9416	- loss: 0.1744	- val_accuracy: 0.6050	- val_loss: 2.2238
Epoch 78/200	25/25	12s	466ms/step	- accuracy: 0.9580	- loss: 0.1393	- val_accuracy: 0.5850	- val_loss: 2.2327
Epoch 79/200	25/25	12s	466ms/step	- accuracy: 0.9435	- loss: 0.1561	- val_accuracy: 0.6100	- val_loss: 2.2643
Epoch 80/200	25/25	12s	475ms/step	- accuracy: 0.9536	- loss: 0.1949	- val_accuracy: 0.5950	- val_loss: 2.2301
Epoch 81/200	25/25	12s	466ms/step	- accuracy: 0.9532	- loss: 0.1703	- val_accuracy: 0.5700	- val_loss: 2.2645
Epoch 82/200	25/25	11s	459ms/step	- accuracy: 0.9617	- loss: 0.0973	- val_accuracy: 0.5700	- val_loss: 2.4140

Epoch 83/200	25/25	11s	450ms/step	- accuracy: 0.9347	- loss: 0.1793	- val_accuracy: 0.5800	- val_loss: 2.2786
Epoch 84/200	25/25	11s	449ms/step	- accuracy: 0.9474	- loss: 0.1033	- val_accuracy: 0.5750	- val_loss: 2.3670
Epoch 85/200	25/25	11s	453ms/step	- accuracy: 0.9548	- loss: 0.1371	- val_accuracy: 0.5600	- val_loss: 2.4043
Epoch 86/200	25/25	11s	455ms/step	- accuracy: 0.9691	- loss: 0.0769	- val_accuracy: 0.5900	- val_loss: 2.4306
Epoch 87/200	25/25	13s	509ms/step	- accuracy: 0.9638	- loss: 0.1350	- val_accuracy: 0.5850	- val_loss: 2.1009
Epoch 88/200	25/25	13s	520ms/step	- accuracy: 0.9666	- loss: 0.0947	- val_accuracy: 0.5700	- val_loss: 2.6385
Epoch 89/200	25/25	12s	463ms/step	- accuracy: 0.9648	- loss: 0.1116	- val_accuracy: 0.5850	- val_loss: 2.4659
Epoch 90/200	25/25	12s	472ms/step	- accuracy: 0.9513	- loss: 0.1856	- val_accuracy: 0.6000	- val_loss: 2.4104
Epoch 91/200	25/25	12s	478ms/step	- accuracy: 0.9636	- loss: 0.1290	- val_accuracy: 0.6000	- val_loss: 2.7544
Epoch 92/200	25/25	12s	466ms/step	- accuracy: 0.9492	- loss: 0.1673	- val_accuracy: 0.5850	- val_loss: 2.5110
Epoch 93/200	25/25	12s	458ms/step	- accuracy: 0.9847	- loss: 0.0773	- val_accuracy: 0.5650	- val_loss: 2.4438
Epoch 94/200	25/25	11s	446ms/step	- accuracy: 0.9749	- loss: 0.0914	- val_accuracy: 0.5900	- val_loss: 2.8167
Epoch 95/200	25/25	12s	463ms/step	- accuracy: 0.9696	- loss: 0.0948	- val_accuracy: 0.5450	- val_loss: 2.9110
Epoch 96/200	25/25	12s	485ms/step	- accuracy: 0.9595	- loss: 0.1383	- val_accuracy: 0.5800	- val_loss: 2.2549
Epoch 97/200	25/25	20s	463ms/step	- accuracy: 0.9758	- loss: 0.0668	- val_accuracy: 0.5550	- val_loss: 2.1897
Epoch 98/200	25/25	11s	458ms/step	- accuracy: 0.9547	- loss: 0.1388	- val_accuracy: 0.5750	- val_loss: 2.3984
Epoch 99/200	25/25	11s	452ms/step	- accuracy: 0.9875	- loss: 0.0453	- val_accuracy: 0.5850	- val_loss: 2.7185
Epoch 100/200	25/25	12s	470ms/step	- accuracy: 0.9522	- loss: 0.1185	- val_accuracy: 0.5650	- val_loss: 2.8550
Epoch 101/200	25/25	12s	466ms/step	- accuracy: 0.9707	- loss: 0.0789	- val_accuracy: 0.5600	- val_loss: 2.9188
Epoch 102/200	25/25	11s	453ms/step	- accuracy: 0.9429	- loss: 0.1708	- val_accuracy: 0.5800	- val_loss: 2.5333
Epoch 103/200	25/25	12s	460ms/step	- accuracy: 0.9541	- loss: 0.1467	- val_accuracy: 0.5650	- val_loss: 2.8877
Epoch 104/200	25/25	12s	466ms/step	- accuracy: 0.9750	- loss: 0.0700	- val_accuracy: 0.5700	- val_loss: 2.9475
Epoch 105/200	25/25	11s	453ms/step	- accuracy: 0.9723	- loss: 0.0933	- val_accuracy: 0.5550	- val_loss: 2.5764
Epoch 106/200	25/25	11s	451ms/step	- accuracy: 0.9811	- loss: 0.0727	- val_accuracy: 0.5650	- val_loss: 2.3499
Epoch 107/200	25/25	12s	466ms/step	- accuracy: 0.9667	- loss: 0.0938	- val_accuracy: 0.5650	- val_loss: 3.0872
Epoch 108/200	25/25	11s	436ms/step	- accuracy: 0.9589	- loss: 0.1414	- val_accuracy: 0.5950	- val_loss: 3.2135
Epoch 109/200	25/25	11s	445ms/step	- accuracy: 0.9679	- loss: 0.1062	- val_accuracy: 0.5700	- val_loss: 2.9951
Epoch 110/200	25/25	11s	451ms/step	- accuracy: 0.9856	- loss: 0.0488	- val_accuracy: 0.5850	- val_loss: 3.3977
Epoch 111/200	25/25	11s	436ms/step	- accuracy: 0.9446	- loss: 0.1606	- val_accuracy: 0.5650	- val_loss: 3.1326
Epoch 112/200	25/25	12s	476ms/step	- accuracy: 0.9832	- loss: 0.0424	- val_accuracy: 0.5650	- val_loss: 2.7708
Epoch 113/200	25/25	11s	451ms/step	- accuracy: 0.9500	- loss: 0.1129	- val_accuracy: 0.5750	- val_loss: 2.9067
Epoch 114/200	25/25	12s	475ms/step	- accuracy: 0.9897	- loss: 0.0355	- val_accuracy: 0.5950	- val_loss: 2.9419
Epoch 115/200	25/25	11s	446ms/step	- accuracy: 0.9769	- loss: 0.0644	- val_accuracy: 0.5600	- val_loss: 3.7960
Epoch 116/200	25/25	11s	456ms/step	- accuracy: 0.9770	- loss: 0.0871	- val_accuracy: 0.5700	- val_loss: 3.1759
Epoch 117/200	25/25	12s	459ms/step	- accuracy: 0.9854	- loss: 0.0627	- val_accuracy: 0.5700	- val_loss: 3.0052
Epoch 118/200	25/25	12s	459ms/step	- accuracy: 0.9806	- loss: 0.0613	- val_accuracy: 0.5750	- val_loss: 3.2695
Epoch 119/200	25/25	11s	446ms/step	- accuracy: 0.9679	- loss: 0.1012	- val_accuracy: 0.5650	- val_loss: 2.8748
Epoch 120/200	25/25	11s	441ms/step	- accuracy: 0.9692	- loss: 0.0690	- val_accuracy: 0.6050	- val_loss: 2.5103
Epoch 121/200	25/25	11s	440ms/step	- accuracy: 0.9825	- loss: 0.0566	- val_accuracy: 0.6000	- val_loss: 2.9525
Epoch 122/200	25/25	21s	467ms/step	- accuracy: 0.9866	- loss: 0.0416	- val_accuracy: 0.5750	- val_loss: 3.5913
Epoch 123/200	25/25	13s	506ms/step	- accuracy: 0.9780	- loss: 0.1305	- val_accuracy: 0.5700	- val_loss: 2.7642

Epoch 124/200	25/25	13s	500ms/step	- accuracy: 0.9718	- loss: 0.0886	- val_accuracy: 0.5750	- val_loss: 3.1585
Epoch 125/200	25/25	12s	458ms/step	- accuracy: 0.9804	- loss: 0.0619	- val_accuracy: 0.5650	- val_loss: 2.9625
Epoch 126/200	25/25	12s	461ms/step	- accuracy: 0.9762	- loss: 0.0649	- val_accuracy: 0.5750	- val_loss: 3.0198
Epoch 127/200	25/25	11s	448ms/step	- accuracy: 0.9707	- loss: 0.1140	- val_accuracy: 0.5750	- val_loss: 2.7649
Epoch 128/200	25/25	11s	425ms/step	- accuracy: 0.9731	- loss: 0.0610	- val_accuracy: 0.6200	- val_loss: 2.5508
Epoch 129/200	25/25	11s	441ms/step	- accuracy: 0.9855	- loss: 0.0462	- val_accuracy: 0.5600	- val_loss: 3.7263
Epoch 130/200	25/25	11s	431ms/step	- accuracy: 0.9819	- loss: 0.0629	- val_accuracy: 0.5700	- val_loss: 3.2173
Epoch 131/200	25/25	11s	431ms/step	- accuracy: 0.9690	- loss: 0.0907	- val_accuracy: 0.5850	- val_loss: 3.3084
Epoch 132/200	25/25	11s	444ms/step	- accuracy: 0.9616	- loss: 0.0969	- val_accuracy: 0.5950	- val_loss: 2.6866
Epoch 133/200	25/25	12s	462ms/step	- accuracy: 0.9888	- loss: 0.0369	- val_accuracy: 0.5950	- val_loss: 2.8829
Epoch 134/200	25/25	11s	447ms/step	- accuracy: 0.9853	- loss: 0.0488	- val_accuracy: 0.5750	- val_loss: 2.7760
Epoch 135/200	25/25	12s	469ms/step	- accuracy: 0.9835	- loss: 0.0482	- val_accuracy: 0.5650	- val_loss: 3.7458
Epoch 136/200	25/25	12s	472ms/step	- accuracy: 0.9735	- loss: 0.0852	- val_accuracy: 0.5800	- val_loss: 2.7073
Epoch 137/200	25/25	12s	484ms/step	- accuracy: 0.9857	- loss: 0.0469	- val_accuracy: 0.5500	- val_loss: 3.1712
Epoch 138/200	25/25	12s	481ms/step	- accuracy: 0.9869	- loss: 0.0471	- val_accuracy: 0.5950	- val_loss: 2.9931
Epoch 139/200	25/25	12s	495ms/step	- accuracy: 0.9815	- loss: 0.0554	- val_accuracy: 0.5850	- val_loss: 2.8937
Epoch 140/200	25/25	12s	484ms/step	- accuracy: 0.9905	- loss: 0.0404	- val_accuracy: 0.5650	- val_loss: 3.5348
Epoch 141/200	25/25	12s	474ms/step	- accuracy: 0.9711	- loss: 0.1107	- val_accuracy: 0.5450	- val_loss: 2.9330
Epoch 142/200	25/25	11s	456ms/step	- accuracy: 0.9827	- loss: 0.0638	- val_accuracy: 0.5800	- val_loss: 2.7175
Epoch 143/200	25/25	11s	453ms/step	- accuracy: 0.9693	- loss: 0.1044	- val_accuracy: 0.5900	- val_loss: 2.5806
Epoch 144/200	25/25	12s	462ms/step	- accuracy: 0.9912	- loss: 0.0224	- val_accuracy: 0.5650	- val_loss: 3.2506
Epoch 145/200	25/25	12s	459ms/step	- accuracy: 0.9614	- loss: 0.1111	- val_accuracy: 0.5550	- val_loss: 3.4666
Epoch 146/200	25/25	11s	456ms/step	- accuracy: 0.9848	- loss: 0.0526	- val_accuracy: 0.5650	- val_loss: 3.5513
Epoch 147/200	25/25	12s	474ms/step	- accuracy: 0.9844	- loss: 0.0788	- val_accuracy: 0.5750	- val_loss: 2.8310
Epoch 148/200	25/25	11s	441ms/step	- accuracy: 0.9880	- loss: 0.0294	- val_accuracy: 0.6150	- val_loss: 3.0327
Epoch 149/200	25/25	12s	463ms/step	- accuracy: 0.9798	- loss: 0.0653	- val_accuracy: 0.6100	- val_loss: 2.5929
Epoch 150/200	25/25	12s	465ms/step	- accuracy: 0.9912	- loss: 0.0427	- val_accuracy: 0.5750	- val_loss: 3.6527
Epoch 151/200	25/25	12s	464ms/step	- accuracy: 0.9785	- loss: 0.1017	- val_accuracy: 0.5600	- val_loss: 3.2230
Epoch 152/200	25/25	12s	466ms/step	- accuracy: 0.9716	- loss: 0.0848	- val_accuracy: 0.5850	- val_loss: 3.4565
Epoch 153/200	25/25	11s	452ms/step	- accuracy: 0.9892	- loss: 0.0360	- val_accuracy: 0.5800	- val_loss: 3.7426
Epoch 154/200	25/25	12s	486ms/step	- accuracy: 0.9900	- loss: 0.0370	- val_accuracy: 0.5850	- val_loss: 3.6133
Epoch 155/200	25/25	12s	475ms/step	- accuracy: 0.9752	- loss: 0.0612	- val_accuracy: 0.5850	- val_loss: 3.4285
Epoch 156/200	25/25	12s	476ms/step	- accuracy: 0.9927	- loss: 0.0228	- val_accuracy: 0.5850	- val_loss: 3.9586
Epoch 157/200	25/25	12s	487ms/step	- accuracy: 0.9683	- loss: 0.1098	- val_accuracy: 0.5600	- val_loss: 3.3739
Epoch 158/200	25/25	11s	453ms/step	- accuracy: 0.9795	- loss: 0.0823	- val_accuracy: 0.5700	- val_loss: 3.5236
Epoch 159/200	25/25	12s	470ms/step	- accuracy: 0.9853	- loss: 0.0560	- val_accuracy: 0.5550	- val_loss: 3.9693
Epoch 160/200	25/25	13s	507ms/step	- accuracy: 0.9642	- loss: 0.1503	- val_accuracy: 0.5900	- val_loss: 3.8540
Epoch 161/200	25/25	12s	473ms/step	- accuracy: 0.9885	- loss: 0.0555	- val_accuracy: 0.5750	- val_loss: 3.8009
Epoch 162/200	25/25	12s	477ms/step	- accuracy: 0.9780	- loss: 0.0851	- val_accuracy: 0.6000	- val_loss: 3.0384
Epoch 163/200	25/25	12s	491ms/step	- accuracy: 0.9875	- loss: 0.0423	- val_accuracy: 0.6050	- val_loss: 3.1324
Epoch 164/200	25/25	12s	472ms/step	- accuracy: 0.9842	- loss: 0.0422	- val_accuracy: 0.5850	- val_loss: 3.5753


```

Epoch 165/200
25/25 ----- 13s 509ms/step - accuracy: 0.9946 - loss: 0.0276 - val_accuracy: 0.5800 - val_loss: 4.1496
Epoch 166/200
25/25 ----- 12s 475ms/step - accuracy: 0.9832 - loss: 0.0464 - val_accuracy: 0.5450 - val_loss: 3.8821
Epoch 167/200
25/25 ----- 13s 500ms/step - accuracy: 0.9846 - loss: 0.0371 - val_accuracy: 0.5700 - val_loss: 4.1487
Epoch 168/200
25/25 ----- 12s 498ms/step - accuracy: 0.9689 - loss: 0.1025 - val_accuracy: 0.5750 - val_loss: 3.4159
Epoch 169/200
25/25 ----- 12s 481ms/step - accuracy: 0.9792 - loss: 0.0696 - val_accuracy: 0.5800 - val_loss: 3.8318
Epoch 170/200
25/25 ----- 12s 474ms/step - accuracy: 0.9868 - loss: 0.0269 - val_accuracy: 0.5700 - val_loss: 3.6744
Epoch 171/200
25/25 ----- 20s 456ms/step - accuracy: 0.9849 - loss: 0.0440 - val_accuracy: 0.5550 - val_loss: 3.8352
Epoch 172/200
25/25 ----- 12s 463ms/step - accuracy: 0.9948 - loss: 0.0249 - val_accuracy: 0.5500 - val_loss: 4.1566
Epoch 173/200
25/25 ----- 13s 528ms/step - accuracy: 0.9784 - loss: 0.0598 - val_accuracy: 0.5800 - val_loss: 4.4257
Epoch 174/200
25/25 ----- 20s 489ms/step - accuracy: 0.9742 - loss: 0.0952 - val_accuracy: 0.5450 - val_loss: 4.0259
Epoch 175/200
25/25 ----- 12s 472ms/step - accuracy: 0.9933 - loss: 0.0179 - val_accuracy: 0.5750 - val_loss: 3.7908
Epoch 176/200
25/25 ----- 12s 475ms/step - accuracy: 0.9774 - loss: 0.1114 - val_accuracy: 0.5750 - val_loss: 3.6599
Epoch 177/200
25/25 ----- 12s 466ms/step - accuracy: 0.9853 - loss: 0.0346 - val_accuracy: 0.5600 - val_loss: 4.2077
Epoch 178/200
25/25 ----- 12s 475ms/step - accuracy: 0.9845 - loss: 0.0378 - val_accuracy: 0.5700 - val_loss: 4.1975
Epoch 179/200
25/25 ----- 12s 477ms/step - accuracy: 0.9792 - loss: 0.0861 - val_accuracy: 0.5750 - val_loss: 4.6767
Epoch 180/200
25/25 ----- 12s 481ms/step - accuracy: 0.9858 - loss: 0.0325 - val_accuracy: 0.5700 - val_loss: 4.3123
Epoch 181/200
25/25 ----- 12s 484ms/step - accuracy: 0.9795 - loss: 0.0905 - val_accuracy: 0.5850 - val_loss: 4.3368
Epoch 182/200
25/25 ----- 11s 446ms/step - accuracy: 0.9910 - loss: 0.0388 - val_accuracy: 0.6150 - val_loss: 4.0462
Epoch 183/200
25/25 ----- 11s 452ms/step - accuracy: 0.9808 - loss: 0.0731 - val_accuracy: 0.6000 - val_loss: 3.6067
Epoch 184/200
25/25 ----- 12s 465ms/step - accuracy: 0.9906 - loss: 0.0207 - val_accuracy: 0.5900 - val_loss: 4.4340
Epoch 185/200
25/25 ----- 11s 453ms/step - accuracy: 0.9866 - loss: 0.0524 - val_accuracy: 0.5950 - val_loss: 4.0456
Epoch 186/200
25/25 ----- 11s 435ms/step - accuracy: 0.9843 - loss: 0.0646 - val_accuracy: 0.6100 - val_loss: 3.4074
Epoch 187/200
25/25 ----- 11s 434ms/step - accuracy: 0.9754 - loss: 0.0781 - val_accuracy: 0.5750 - val_loss: 3.6424
Epoch 188/200
25/25 ----- 11s 435ms/step - accuracy: 0.9921 - loss: 0.0286 - val_accuracy: 0.5900 - val_loss: 3.4255
Epoch 189/200
25/25 ----- 11s 458ms/step - accuracy: 0.9945 - loss: 0.0156 - val_accuracy: 0.6150 - val_loss: 3.5043
Epoch 190/200
25/25 ----- 11s 457ms/step - accuracy: 0.9765 - loss: 0.0669 - val_accuracy: 0.5900 - val_loss: 3.0471
Epoch 191/200
25/25 ----- 11s 441ms/step - accuracy: 0.9822 - loss: 0.0497 - val_accuracy: 0.5900 - val_loss: 3.5785
Epoch 192/200
25/25 ----- 11s 432ms/step - accuracy: 0.9866 - loss: 0.0376 - val_accuracy: 0.6000 - val_loss: 3.5430
Epoch 193/200
25/25 ----- 11s 447ms/step - accuracy: 0.9905 - loss: 0.0331 - val_accuracy: 0.5700 - val_loss: 3.9939
Epoch 194/200
25/25 ----- 21s 459ms/step - accuracy: 0.9802 - loss: 0.0612 - val_accuracy: 0.5750 - val_loss: 4.2690
Epoch 195/200
25/25 ----- 12s 458ms/step - accuracy: 0.9828 - loss: 0.0629 - val_accuracy: 0.5700 - val_loss: 3.7037
Epoch 196/200
25/25 ----- 11s 457ms/step - accuracy: 0.9917 - loss: 0.0292 - val_accuracy: 0.6000 - val_loss: 3.6349
Epoch 197/200
25/25 ----- 11s 457ms/step - accuracy: 0.9881 - loss: 0.0378 - val_accuracy: 0.5800 - val_loss: 3.7407
Epoch 198/200
25/25 ----- 12s 460ms/step - accuracy: 0.9869 - loss: 0.0346 - val_accuracy: 0.5800 - val_loss: 3.9399
Epoch 199/200
25/25 ----- 12s 461ms/step - accuracy: 0.9959 - loss: 0.0230 - val_accuracy: 0.5700 - val_loss: 4.6127
Epoch 200/200
25/25 ----- 12s 467ms/step - accuracy: 0.9954 - loss: 0.0106 - val_accuracy: 0.5800 - val_loss: 4.2340

```

```
In [8]: print(f"Execution time: {elapsed_time:.2f} seconds")
```

Execution time: 2385.14 seconds

```
In [9]: def append_core_data(score_path, num_cores, elapsed_time):
        # Check if the file already exists
        file_exists = os.path.exists(score_path)

        # Open the file in append mode
        with open(score_path, mode='a', newline='') as file:
```



```
writer = csv.writer(file)

# If the file is new, write the header
if not file_exists:
    writer.writerow(["Number of Cores", "Elapsed Time"])

# Write the new data
writer.writerow([num_cores, elapsed_time])
```

```
In [10]: score_path = r"C:\Users\nikhi\OneDrive\Desktop\Final Project\DEEP LEARNING WITH HPSC\core_data.txt"
append_core_data(score_path, number_of_worker, elapsed_time)
```

Import Req Lib

```
In [1]: %matplotlib inline

import shutil
import random
import numpy as np
from warnings import filterwarnings
filterwarnings('ignore')

from tensorflow.keras import layers, regularizers, optimizers
from tensorflow.keras import models
from tensorflow.keras.models import Sequential, Model
from tensorflow.keras.layers import LeakyReLU, Dense, Activation, Flatten, Dropout, BatchNormalization, Conv2D, MaxPooling2D
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import tensorflow as tf

import os
import time
import csv

from matplotlib import figure
```

Define 4 worker

```
In [2]: # Set the number of threads
number_of_worker = 4
os.environ['OMP_NUM_THREADS'] = '4' # OpenMP threads for parallelism
os.environ['TF_NUM_INTEROP_THREADS'] = '4' # Threads for inter-operation parallelism
os.environ['TF_NUM_INTRAOP_THREADS'] = '4' # Threads for intra-operation parallelism

# Confirm TensorFlow is using the specified number of threads
tf.config.threading.set_inter_op_parallelism_threads(number_of_worker)
tf.config.threading.set_intra_op_parallelism_threads(number_of_worker)
```

Train Val data Split

```
In [3]: source_dir = r"C:\Users\nikhi\OneDrive\Desktop\Final Project\DATA\Convert_Audio_File_to_jpg_file"
target_dir = r'genres_train_val_split_data'
split_ratio = 0.8

def Train_Test_Split(source_dir, target_dir, split_ratio):
    # Define source and target directories
    train_dir = os.path.join(target_dir, 'train')
    val_dir = os.path.join(target_dir, 'val')

    # Create target directories if they don't exist
    os.makedirs(train_dir, exist_ok=True)
    os.makedirs(val_dir, exist_ok=True)

    # Get the list of class directories
    classes = [d for d in os.listdir(source_dir) if os.path.isdir(os.path.join(source_dir, d))]

    for class_name in classes:
        # Create class directories in train and val folders
        os.makedirs(os.path.join(train_dir, class_name), exist_ok=True)
        os.makedirs(os.path.join(val_dir, class_name), exist_ok=True)

        # Get list of images in the class directory
        class_dir = os.path.join(source_dir, class_name)
        images = [f for f in os.listdir(class_dir) if os.path.isfile(os.path.join(class_dir, f))]

        # Shuffle the images
        random.shuffle(images)

        # Compute the split point
        split_point = int(len(images) * split_ratio)

        # Split the images into training and validation sets
        train_images = images[:split_point]
        val_images = images[split_point:]
```

```

# Move the images to the respective directories
for img in train_images:
    shutil.copy(os.path.join(class_dir, img), os.path.join(train_dir, class_name, img))

for img in val_images:
    shutil.copy(os.path.join(class_dir, img), os.path.join(val_dir, class_name, img))

print("Data split completed successfully!")

```

In [4]: `Train_Test_Split(source_dir,target_dir,split_ratio)`

Data split completed successfully!

Load the Data

```

In [5]: WIDTH = 64
HEIGHT = 64
BATCH_SIZE = 32
TRAIN_DIR=r'genres_train_val_split_data/train'
val_dir = r'genres_train_val_split_data/val'

# data prep
train_datagen = ImageDataGenerator(
    rescale=1./255.,validation_split=0.25)

train_generator = train_datagen.flow_from_directory(
    TRAIN_DIR,
    target_size=(HEIGHT, WIDTH),
    batch_size=BATCH_SIZE,
    class_mode='categorical')

validation_gen = train_datagen.flow_from_directory(
    val_dir,target_size = (HEIGHT,WIDTH),
    batch_size = BATCH_SIZE,
    class_mode = 'categorical'
)

```

Found 800 images belonging to 10 classes.

Found 200 images belonging to 10 classes.

Model Architecture

```

In [6]: model = Sequential()
model.add(Conv2D(32, (3, 3), padding='same',
    input_shape=(64,64,3)))
model.add(Activation('relu'))
model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Conv2D(64, (3, 3), padding='same'))
model.add(Activation('relu'))
model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.5))
model.add(Conv2D(128, (3, 3), padding='same'))
model.add(Activation('relu'))
model.add(Conv2D(128, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.5))
model.add(Flatten())
model.add(Dense(512))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(10, activation='softmax'))
model.compile(optimizer=RMSprop(learning_rate=0.0005, decay=1e-6),loss="categorical_crossentropy",metrics=["accuracy"])
model.summary()

```

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 64, 64, 32)	896
activation (Activation)	(None, 64, 64, 32)	0
conv2d_1 (Conv2D)	(None, 62, 62, 64)	18,496
activation_1 (Activation)	(None, 62, 62, 64)	0
max_pooling2d (MaxPooling2D)	(None, 31, 31, 64)	0
dropout (Dropout)	(None, 31, 31, 64)	0
conv2d_2 (Conv2D)	(None, 31, 31, 64)	36,928
activation_2 (Activation)	(None, 31, 31, 64)	0
conv2d_3 (Conv2D)	(None, 29, 29, 64)	36,928
activation_3 (Activation)	(None, 29, 29, 64)	0
max_pooling2d_1 (MaxPooling2D)	(None, 14, 14, 64)	0
dropout_1 (Dropout)	(None, 14, 14, 64)	0
conv2d_4 (Conv2D)	(None, 14, 14, 128)	73,856
activation_4 (Activation)	(None, 14, 14, 128)	0
conv2d_5 (Conv2D)	(None, 12, 12, 128)	147,584
activation_5 (Activation)	(None, 12, 12, 128)	0
max_pooling2d_2 (MaxPooling2D)	(None, 6, 6, 128)	0
dropout_2 (Dropout)	(None, 6, 6, 128)	0
flatten (Flatten)	(None, 4608)	0
dense (Dense)	(None, 512)	2,359,808
activation_6 (Activation)	(None, 512)	0
dropout_3 (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 10)	5,130

Total params: 2,679,626 (10.22 MB)

Trainable params: 2,679,626 (10.22 MB)

Non-trainable params: 0 (0.00 B)

```
In [7]: STEP_SIZE_TRAIN=train_generator.n//train_generator.batch_size
# Measure the execution time
start_time = time.time()

model.fit(train_generator,validation_data=validation_gen,epochs=200)

end_time = time.time()
elapsed_time = end_time - start_time
```

[illegible]

Epoch 42/200
25/25 ————— 5s 209ms/step - accuracy: 0.7806 - loss: 0.6490 - val_accuracy: 0.5650 - val_loss: 1.2638

Epoch 43/200
25/25 ————— 6s 223ms/step - accuracy: 0.7568 - loss: 0.6466 - val_accuracy: 0.5850 - val_loss: 1.2449

Epoch 44/200
25/25 ————— 5s 207ms/step - accuracy: 0.7851 - loss: 0.5774 - val_accuracy: 0.5750 - val_loss: 1.3505

Epoch 45/200
25/25 ————— 5s 213ms/step - accuracy: 0.7652 - loss: 0.6773 - val_accuracy: 0.6100 - val_loss: 1.4523

Epoch 46/200
25/25 ————— 5s 204ms/step - accuracy: 0.7777 - loss: 0.5810 - val_accuracy: 0.5950 - val_loss: 1.4568

Epoch 47/200
25/25 ————— 5s 205ms/step - accuracy: 0.7944 - loss: 0.5911 - val_accuracy: 0.6000 - val_loss: 1.4199

Epoch 48/200
25/25 ————— 5s 202ms/step - accuracy: 0.8184 - loss: 0.5705 - val_accuracy: 0.5950 - val_loss: 1.4033

Epoch 49/200
25/25 ————— 5s 208ms/step - accuracy: 0.8032 - loss: 0.4969 - val_accuracy: 0.5350 - val_loss: 1.5725

Epoch 50/200
25/25 ————— 5s 203ms/step - accuracy: 0.8524 - loss: 0.4568 - val_accuracy: 0.5650 - val_loss: 1.5162

Epoch 51/200
25/25 ————— 5s 210ms/step - accuracy: 0.8675 - loss: 0.4323 - val_accuracy: 0.5550 - val_loss: 1.6451

Epoch 52/200
25/25 ————— 5s 204ms/step - accuracy: 0.8801 - loss: 0.3655 - val_accuracy: 0.6250 - val_loss: 1.4862

Epoch 53/200
25/25 ————— 5s 199ms/step - accuracy: 0.8359 - loss: 0.4234 - val_accuracy: 0.6050 - val_loss: 1.5294

Epoch 54/200
25/25 ————— 5s 215ms/step - accuracy: 0.8761 - loss: 0.3410 - val_accuracy: 0.5500 - val_loss: 1.8546

Epoch 55/200
25/25 ————— 5s 210ms/step - accuracy: 0.8498 - loss: 0.4090 - val_accuracy: 0.6050 - val_loss: 1.5737

Epoch 56/200
25/25 ————— 5s 216ms/step - accuracy: 0.8667 - loss: 0.3876 - val_accuracy: 0.5550 - val_loss: 2.0229

Epoch 57/200
25/25 ————— 5s 201ms/step - accuracy: 0.8925 - loss: 0.3106 - val_accuracy: 0.6200 - val_loss: 1.6577

Epoch 58/200
25/25 ————— 5s 207ms/step - accuracy: 0.8861 - loss: 0.3055 - val_accuracy: 0.6050 - val_loss: 1.7928

Epoch 59/200
25/25 ————— 5s 199ms/step - accuracy: 0.9239 - loss: 0.2821 - val_accuracy: 0.6300 - val_loss: 1.6734

Epoch 60/200
25/25 ————— 5s 211ms/step - accuracy: 0.9099 - loss: 0.2721 - val_accuracy: 0.6400 - val_loss: 1.8624

Epoch 61/200
25/25 ————— 5s 209ms/step - accuracy: 0.8851 - loss: 0.2915 - val_accuracy: 0.6300 - val_loss: 1.7194

Epoch 62/200
25/25 ————— 5s 205ms/step - accuracy: 0.9221 - loss: 0.2401 - val_accuracy: 0.5750 - val_loss: 2.4121

Epoch 63/200
25/25 ————— 5s 211ms/step - accuracy: 0.8771 - loss: 0.4255 - val_accuracy: 0.6400 - val_loss: 1.6150

Epoch 64/200
25/25 ————— 5s 208ms/step - accuracy: 0.9291 - loss: 0.2064 - val_accuracy: 0.6100 - val_loss: 2.1086

Epoch 65/200
25/25 ————— 5s 203ms/step - accuracy: 0.8845 - loss: 0.2733 - val_accuracy: 0.5550 - val_loss: 1.9794

Epoch 66/200
25/25 ————— 5s 208ms/step - accuracy: 0.9221 - loss: 0.2269 - val_accuracy: 0.6400 - val_loss: 1.6637

Epoch 67/200
25/25 ————— 5s 212ms/step - accuracy: 0.9380 - loss: 0.2449 - val_accuracy: 0.5950 - val_loss: 1.9453

Epoch 68/200
25/25 ————— 5s 196ms/step - accuracy: 0.9244 - loss: 0.1791 - val_accuracy: 0.6150 - val_loss: 1.8998

Epoch 69/200
25/25 ————— 5s 218ms/step - accuracy: 0.9393 - loss: 0.1946 - val_accuracy: 0.6400 - val_loss: 2.0027

Epoch 70/200
25/25 ————— 5s 212ms/step - accuracy: 0.9358 - loss: 0.1815 - val_accuracy: 0.6250 - val_loss: 2.1165

Epoch 71/200
25/25 ————— 5s 197ms/step - accuracy: 0.8922 - loss: 0.2702 - val_accuracy: 0.6250 - val_loss: 1.6935

Epoch 72/200
25/25 ————— 5s 211ms/step - accuracy: 0.9485 - loss: 0.1672 - val_accuracy: 0.6050 - val_loss: 1.8513

Epoch 73/200
25/25 ————— 5s 202ms/step - accuracy: 0.9588 - loss: 0.1375 - val_accuracy: 0.6150 - val_loss: 2.1926

Epoch 74/200
25/25 ————— 5s 204ms/step - accuracy: 0.9385 - loss: 0.1757 - val_accuracy: 0.6100 - val_loss: 2.0779

Epoch 75/200
25/25 ————— 5s 203ms/step - accuracy: 0.9415 - loss: 0.1850 - val_accuracy: 0.6000 - val_loss: 2.5533

Epoch 76/200
25/25 ————— 5s 202ms/step - accuracy: 0.9361 - loss: 0.1930 - val_accuracy: 0.6200 - val_loss: 2.0268

Epoch 77/200
25/25 ————— 5s 202ms/step - accuracy: 0.9443 - loss: 0.1485 - val_accuracy: 0.6150 - val_loss: 1.9755

Epoch 78/200
25/25 ————— 5s 203ms/step - accuracy: 0.9376 - loss: 0.1604 - val_accuracy: 0.6000 - val_loss: 2.1409

Epoch 79/200
25/25 ————— 5s 203ms/step - accuracy: 0.9639 - loss: 0.1241 - val_accuracy: 0.6350 - val_loss: 1.8937

Epoch 80/200
25/25 ————— 5s 199ms/step - accuracy: 0.9295 - loss: 0.1915 - val_accuracy: 0.6250 - val_loss: 2.3712

Epoch 81/200
25/25 ————— 5s 209ms/step - accuracy: 0.9673 - loss: 0.1002 - val_accuracy: 0.6200 - val_loss: 2.1481

Epoch 82/200
25/25 ————— 5s 215ms/step - accuracy: 0.9725 - loss: 0.1002 - val_accuracy: 0.6100 - val_loss: 2.2447

Epoch 83/200
25/25 ————— 5s 197ms/step - accuracy: 0.9341 - loss: 0.1945 - val_accuracy: 0.6350 - val_loss: 2.1486

Epoch 84/200
25/25 ————— 5s 210ms/step - accuracy: 0.9491 - loss: 0.1159 - val_accuracy: 0.6250 - val_loss: 2.4833

Epoch 85/200
25/25 ————— 5s 202ms/step - accuracy: 0.9479 - loss: 0.1705 - val_accuracy: 0.6550 - val_loss: 2.0087

Epoch 86/200
25/25 ————— 5s 207ms/step - accuracy: 0.9279 - loss: 0.2786 - val_accuracy: 0.6250 - val_loss: 1.9962

Epoch 87/200
25/25 ————— 5s 201ms/step - accuracy: 0.9769 - loss: 0.0652 - val_accuracy: 0.6300 - val_loss: 2.4628

Epoch 88/200
25/25 ————— 5s 199ms/step - accuracy: 0.9497 - loss: 0.1454 - val_accuracy: 0.6500 - val_loss: 2.3110

Epoch 89/200
25/25 ————— 5s 199ms/step - accuracy: 0.9618 - loss: 0.1068 - val_accuracy: 0.6050 - val_loss: 2.2710

Epoch 90/200
25/25 ————— 5s 205ms/step - accuracy: 0.9721 - loss: 0.0832 - val_accuracy: 0.5950 - val_loss: 2.7386

Epoch 91/200
25/25 ————— 5s 204ms/step - accuracy: 0.9349 - loss: 0.1676 - val_accuracy: 0.6250 - val_loss: 2.4609

Epoch 92/200
25/25 ————— 5s 205ms/step - accuracy: 0.9738 - loss: 0.0941 - val_accuracy: 0.6350 - val_loss: 2.3282

Epoch 93/200
25/25 ————— 5s 207ms/step - accuracy: 0.9720 - loss: 0.0798 - val_accuracy: 0.6100 - val_loss: 2.5202

Epoch 94/200
25/25 ————— 5s 213ms/step - accuracy: 0.9720 - loss: 0.0775 - val_accuracy: 0.6000 - val_loss: 2.6692

Epoch 95/200
25/25 ————— 5s 204ms/step - accuracy: 0.9602 - loss: 0.1247 - val_accuracy: 0.6050 - val_loss: 2.3386

Epoch 96/200
25/25 ————— 5s 211ms/step - accuracy: 0.9693 - loss: 0.1137 - val_accuracy: 0.5850 - val_loss: 2.4407

Epoch 97/200
25/25 ————— 5s 213ms/step - accuracy: 0.9746 - loss: 0.0910 - val_accuracy: 0.6400 - val_loss: 2.2343

Epoch 98/200
25/25 ————— 5s 201ms/step - accuracy: 0.9675 - loss: 0.0999 - val_accuracy: 0.6050 - val_loss: 2.4878

Epoch 99/200
25/25 ————— 5s 209ms/step - accuracy: 0.9729 - loss: 0.1077 - val_accuracy: 0.6300 - val_loss: 2.3810

Epoch 100/200
25/25 ————— 5s 207ms/step - accuracy: 0.9738 - loss: 0.0803 - val_accuracy: 0.6200 - val_loss: 2.4213

Epoch 101/200
25/25 ————— 5s 208ms/step - accuracy: 0.9728 - loss: 0.0843 - val_accuracy: 0.6150 - val_loss: 2.6465

Epoch 102/200
25/25 ————— 5s 206ms/step - accuracy: 0.9731 - loss: 0.0868 - val_accuracy: 0.6150 - val_loss: 2.2760

Epoch 103/200
25/25 ————— 5s 208ms/step - accuracy: 0.9727 - loss: 0.0999 - val_accuracy: 0.6450 - val_loss: 2.4515

Epoch 104/200
25/25 ————— 5s 202ms/step - accuracy: 0.9519 - loss: 0.1494 - val_accuracy: 0.6300 - val_loss: 2.3754

Epoch 105/200
25/25 ————— 5s 209ms/step - accuracy: 0.9746 - loss: 0.0898 - val_accuracy: 0.6300 - val_loss: 2.7129

Epoch 106/200
25/25 ————— 5s 196ms/step - accuracy: 0.9821 - loss: 0.0691 - val_accuracy: 0.6300 - val_loss: 2.8243

Epoch 107/200
25/25 ————— 5s 213ms/step - accuracy: 0.9801 - loss: 0.0752 - val_accuracy: 0.6200 - val_loss: 2.7709

Epoch 108/200
25/25 ————— 5s 216ms/step - accuracy: 0.9664 - loss: 0.1235 - val_accuracy: 0.6100 - val_loss: 2.7197

Epoch 109/200
25/25 ————— 5s 206ms/step - accuracy: 0.9812 - loss: 0.0641 - val_accuracy: 0.6400 - val_loss: 2.7450

Epoch 110/200
25/25 ————— 5s 200ms/step - accuracy: 0.9763 - loss: 0.0770 - val_accuracy: 0.6200 - val_loss: 2.6184

Epoch 111/200
25/25 ————— 5s 207ms/step - accuracy: 0.9752 - loss: 0.0604 - val_accuracy: 0.6100 - val_loss: 2.6203

Epoch 112/200
25/25 ————— 5s 204ms/step - accuracy: 0.9920 - loss: 0.0344 - val_accuracy: 0.6200 - val_loss: 2.7276

Epoch 113/200
25/25 ————— 5s 213ms/step - accuracy: 0.9461 - loss: 0.1494 - val_accuracy: 0.6650 - val_loss: 2.4975

Epoch 114/200
25/25 ————— 5s 212ms/step - accuracy: 0.9848 - loss: 0.0547 - val_accuracy: 0.6350 - val_loss: 2.7020

Epoch 115/200
25/25 ————— 5s 210ms/step - accuracy: 0.9735 - loss: 0.0877 - val_accuracy: 0.6250 - val_loss: 2.7517

Epoch 116/200
25/25 ————— 5s 211ms/step - accuracy: 0.9790 - loss: 0.0701 - val_accuracy: 0.6350 - val_loss: 2.5372

Epoch 117/200
25/25 ————— 5s 208ms/step - accuracy: 0.9793 - loss: 0.0651 - val_accuracy: 0.6350 - val_loss: 3.0050

Epoch 118/200
25/25 ————— 5s 207ms/step - accuracy: 0.9703 - loss: 0.0804 - val_accuracy: 0.6500 - val_loss: 2.2230

Epoch 119/200
25/25 ————— 5s 196ms/step - accuracy: 0.9836 - loss: 0.0570 - val_accuracy: 0.6300 - val_loss: 2.7662

Epoch 120/200
25/25 ————— 5s 216ms/step - accuracy: 0.9821 - loss: 0.0468 - val_accuracy: 0.6450 - val_loss: 2.6975

Epoch 121/200
25/25 ————— 5s 207ms/step - accuracy: 0.9702 - loss: 0.0829 - val_accuracy: 0.6350 - val_loss: 2.5993

Epoch 122/200
25/25 ————— 5s 207ms/step - accuracy: 0.9644 - loss: 0.1291 - val_accuracy: 0.6450 - val_loss: 2.7768

Epoch 123/200
25/25 ————— 5s 210ms/step - accuracy: 0.9766 - loss: 0.0689 - val_accuracy: 0.6200 - val_loss: 2.8439

Epoch 124/200
25/25 ————— 5s 210ms/step - accuracy: 0.9886 - loss: 0.0279 - val_accuracy: 0.6100 - val_loss: 3.0244

Epoch 125/200
25/25 ————— 5s 200ms/step - accuracy: 0.9813 - loss: 0.0663 - val_accuracy: 0.6500 - val_loss: 3.1208

Epoch 126/200
25/25 ————— 5s 215ms/step - accuracy: 0.9855 - loss: 0.0411 - val_accuracy: 0.6500 - val_loss: 2.7819

Epoch 127/200
25/25 ————— 5s 208ms/step - accuracy: 0.9888 - loss: 0.0385 - val_accuracy: 0.6600 - val_loss: 3.1580

Epoch 128/200
25/25 ————— 5s 203ms/step - accuracy: 0.9817 - loss: 0.0764 - val_accuracy: 0.6650 - val_loss: 2.7910

Epoch 129/200
25/25 ————— 5s 207ms/step - accuracy: 0.9767 - loss: 0.0670 - val_accuracy: 0.6450 - val_loss: 3.1073

Epoch 130/200
25/25 ————— 5s 204ms/step - accuracy: 0.9796 - loss: 0.0653 - val_accuracy: 0.6350 - val_loss: 2.9026

Epoch 131/200
25/25 ————— 5s 208ms/step - accuracy: 0.9897 - loss: 0.0606 - val_accuracy: 0.5650 - val_loss: 4.0656

Epoch 132/200
25/25 ————— 5s 213ms/step - accuracy: 0.9431 - loss: 0.2323 - val_accuracy: 0.6550 - val_loss: 2.6973

Epoch 133/200
25/25 ————— 6s 220ms/step - accuracy: 0.9872 - loss: 0.0522 - val_accuracy: 0.6400 - val_loss: 2.6923

Epoch 134/200
25/25 ————— 5s 201ms/step - accuracy: 0.9980 - loss: 0.0147 - val_accuracy: 0.6550 - val_loss: 2.8682

Epoch 135/200
25/25 ————— 5s 207ms/step - accuracy: 0.9789 - loss: 0.0659 - val_accuracy: 0.6550 - val_loss: 2.9029

Epoch 136/200
25/25 ————— 5s 213ms/step - accuracy: 0.9781 - loss: 0.0519 - val_accuracy: 0.6450 - val_loss: 3.3952

Epoch 137/200
25/25 ————— 5s 204ms/step - accuracy: 0.9841 - loss: 0.0550 - val_accuracy: 0.6950 - val_loss: 2.4342

Epoch 138/200
25/25 ————— 5s 213ms/step - accuracy: 0.9835 - loss: 0.0416 - val_accuracy: 0.5950 - val_loss: 3.0843

Epoch 139/200
25/25 ————— 5s 212ms/step - accuracy: 0.9796 - loss: 0.0624 - val_accuracy: 0.6200 - val_loss: 3.3547

Epoch 140/200
25/25 ————— 5s 201ms/step - accuracy: 0.9814 - loss: 0.0590 - val_accuracy: 0.6500 - val_loss: 2.9060

Epoch 141/200
25/25 ————— 5s 211ms/step - accuracy: 0.9790 - loss: 0.0728 - val_accuracy: 0.6400 - val_loss: 3.7736

Epoch 142/200
25/25 ————— 5s 218ms/step - accuracy: 0.9811 - loss: 0.0637 - val_accuracy: 0.6450 - val_loss: 3.2781

Epoch 143/200
25/25 ————— 5s 206ms/step - accuracy: 0.9746 - loss: 0.0844 - val_accuracy: 0.6050 - val_loss: 3.2517

Epoch 144/200
25/25 ————— 5s 201ms/step - accuracy: 0.9946 - loss: 0.0231 - val_accuracy: 0.6300 - val_loss: 3.2410

Epoch 145/200
25/25 ————— 5s 213ms/step - accuracy: 0.9646 - loss: 0.1369 - val_accuracy: 0.6450 - val_loss: 3.3306

Epoch 146/200
25/25 ————— 5s 196ms/step - accuracy: 0.9848 - loss: 0.0446 - val_accuracy: 0.6400 - val_loss: 2.9006

Epoch 147/200
25/25 ————— 5s 212ms/step - accuracy: 0.9890 - loss: 0.0328 - val_accuracy: 0.6350 - val_loss: 3.0408

Epoch 148/200
25/25 ————— 5s 206ms/step - accuracy: 0.9745 - loss: 0.0681 - val_accuracy: 0.6250 - val_loss: 2.8711

Epoch 149/200
25/25 ————— 5s 210ms/step - accuracy: 0.9827 - loss: 0.0773 - val_accuracy: 0.6700 - val_loss: 3.0453

Epoch 150/200
25/25 ————— 5s 203ms/step - accuracy: 0.9862 - loss: 0.0550 - val_accuracy: 0.6600 - val_loss: 2.8548

Epoch 151/200
25/25 ————— 5s 207ms/step - accuracy: 0.9790 - loss: 0.0703 - val_accuracy: 0.6650 - val_loss: 2.9448

Epoch 152/200
25/25 ————— 5s 206ms/step - accuracy: 0.9811 - loss: 0.0715 - val_accuracy: 0.6550 - val_loss: 3.0744

Epoch 153/200
25/25 ————— 5s 210ms/step - accuracy: 0.9831 - loss: 0.0430 - val_accuracy: 0.6500 - val_loss: 3.1276

Epoch 154/200
25/25 ————— 5s 208ms/step - accuracy: 0.9720 - loss: 0.0975 - val_accuracy: 0.6400 - val_loss: 2.9376

Epoch 155/200
25/25 ————— 5s 206ms/step - accuracy: 0.9923 - loss: 0.0249 - val_accuracy: 0.6200 - val_loss: 3.2989

Epoch 156/200
25/25 ————— 5s 207ms/step - accuracy: 0.9571 - loss: 0.0950 - val_accuracy: 0.6300 - val_loss: 3.2689

Epoch 157/200
25/25 ————— 5s 202ms/step - accuracy: 0.9922 - loss: 0.0247 - val_accuracy: 0.6300 - val_loss: 3.2124

Epoch 158/200
25/25 ————— 5s 199ms/step - accuracy: 0.9918 - loss: 0.0198 - val_accuracy: 0.6200 - val_loss: 3.5101

Epoch 159/200
25/25 ————— 5s 208ms/step - accuracy: 0.9756 - loss: 0.0976 - val_accuracy: 0.6450 - val_loss: 3.0890

Epoch 160/200
25/25 ————— 5s 200ms/step - accuracy: 0.9899 - loss: 0.0320 - val_accuracy: 0.5250 - val_loss: 4.3071

Epoch 161/200
25/25 ————— 5s 200ms/step - accuracy: 0.9500 - loss: 0.1923 - val_accuracy: 0.6150 - val_loss: 2.8679

Epoch 162/200
25/25 ————— 5s 206ms/step - accuracy: 0.9766 - loss: 0.0628 - val_accuracy: 0.6350 - val_loss: 2.5324

Epoch 163/200
25/25 ————— 5s 205ms/step - accuracy: 0.9859 - loss: 0.0392 - val_accuracy: 0.6400 - val_loss: 2.7800

Epoch 164/200
25/25 ————— 5s 205ms/step - accuracy: 0.9729 - loss: 0.0914 - val_accuracy: 0.6250 - val_loss: 3.1907


```

Epoch 165/200
25/25 ————— 5s 206ms/step - accuracy: 0.9707 - loss: 0.0861 - val_accuracy: 0.6700 - val_loss: 3.0659
Epoch 166/200
25/25 ————— 5s 201ms/step - accuracy: 0.9804 - loss: 0.0650 - val_accuracy: 0.6550 - val_loss: 2.8892
Epoch 167/200
25/25 ————— 5s 204ms/step - accuracy: 0.9777 - loss: 0.0900 - val_accuracy: 0.6450 - val_loss: 3.1240
Epoch 168/200
25/25 ————— 5s 207ms/step - accuracy: 0.9789 - loss: 0.0641 - val_accuracy: 0.6600 - val_loss: 2.8889
Epoch 169/200
25/25 ————— 5s 199ms/step - accuracy: 0.9611 - loss: 0.1326 - val_accuracy: 0.6800 - val_loss: 2.6419
Epoch 170/200
25/25 ————— 5s 211ms/step - accuracy: 0.9922 - loss: 0.0444 - val_accuracy: 0.6250 - val_loss: 3.5239
Epoch 171/200
25/25 ————— 6s 224ms/step - accuracy: 0.9777 - loss: 0.0627 - val_accuracy: 0.6350 - val_loss: 3.0843
Epoch 172/200
25/25 ————— 5s 209ms/step - accuracy: 0.9845 - loss: 0.0833 - val_accuracy: 0.6400 - val_loss: 3.2503
Epoch 173/200
25/25 ————— 5s 207ms/step - accuracy: 0.9847 - loss: 0.0406 - val_accuracy: 0.6300 - val_loss: 3.2927
Epoch 174/200
25/25 ————— 5s 205ms/step - accuracy: 0.9957 - loss: 0.0156 - val_accuracy: 0.6350 - val_loss: 3.5053
Epoch 175/200
25/25 ————— 5s 203ms/step - accuracy: 0.9836 - loss: 0.0420 - val_accuracy: 0.5950 - val_loss: 3.7164
Epoch 176/200
25/25 ————— 5s 204ms/step - accuracy: 0.9834 - loss: 0.0705 - val_accuracy: 0.6450 - val_loss: 3.0541
Epoch 177/200
25/25 ————— 5s 204ms/step - accuracy: 0.9733 - loss: 0.0997 - val_accuracy: 0.6600 - val_loss: 3.0176
Epoch 178/200
25/25 ————— 5s 195ms/step - accuracy: 0.9859 - loss: 0.0259 - val_accuracy: 0.6150 - val_loss: 3.4758
Epoch 179/200
25/25 ————— 5s 199ms/step - accuracy: 0.9645 - loss: 0.1496 - val_accuracy: 0.6200 - val_loss: 3.6725
Epoch 180/200
25/25 ————— 5s 203ms/step - accuracy: 0.9876 - loss: 0.0493 - val_accuracy: 0.6200 - val_loss: 3.2856
Epoch 181/200
25/25 ————— 5s 195ms/step - accuracy: 0.9886 - loss: 0.0565 - val_accuracy: 0.6250 - val_loss: 3.9449
Epoch 182/200
25/25 ————— 5s 202ms/step - accuracy: 0.9770 - loss: 0.0616 - val_accuracy: 0.6200 - val_loss: 3.2310
Epoch 183/200
25/25 ————— 5s 213ms/step - accuracy: 0.9864 - loss: 0.0475 - val_accuracy: 0.6450 - val_loss: 3.0371
Epoch 184/200
25/25 ————— 6s 228ms/step - accuracy: 0.9862 - loss: 0.0486 - val_accuracy: 0.6350 - val_loss: 3.6485
Epoch 185/200
25/25 ————— 5s 207ms/step - accuracy: 0.9795 - loss: 0.0680 - val_accuracy: 0.6100 - val_loss: 3.3821
Epoch 186/200
25/25 ————— 5s 197ms/step - accuracy: 0.9834 - loss: 0.0764 - val_accuracy: 0.6450 - val_loss: 3.2714
Epoch 187/200
25/25 ————— 5s 201ms/step - accuracy: 0.9888 - loss: 0.0797 - val_accuracy: 0.6800 - val_loss: 3.0651
Epoch 188/200
25/25 ————— 5s 198ms/step - accuracy: 0.9895 - loss: 0.0452 - val_accuracy: 0.6000 - val_loss: 3.8176
Epoch 189/200
25/25 ————— 5s 212ms/step - accuracy: 0.9764 - loss: 0.0732 - val_accuracy: 0.6300 - val_loss: 3.2673
Epoch 190/200
25/25 ————— 5s 208ms/step - accuracy: 0.9867 - loss: 0.0555 - val_accuracy: 0.6400 - val_loss: 3.1765
Epoch 191/200
25/25 ————— 5s 206ms/step - accuracy: 0.9903 - loss: 0.0231 - val_accuracy: 0.6750 - val_loss: 3.3168
Epoch 192/200
25/25 ————— 5s 204ms/step - accuracy: 0.9844 - loss: 0.0556 - val_accuracy: 0.6400 - val_loss: 2.8865
Epoch 193/200
25/25 ————— 5s 206ms/step - accuracy: 0.9779 - loss: 0.0447 - val_accuracy: 0.6750 - val_loss: 3.1290
Epoch 194/200
25/25 ————— 5s 210ms/step - accuracy: 0.9906 - loss: 0.0313 - val_accuracy: 0.5900 - val_loss: 3.4275
Epoch 195/200
25/25 ————— 5s 206ms/step - accuracy: 0.9902 - loss: 0.0367 - val_accuracy: 0.6250 - val_loss: 3.2306
Epoch 196/200
25/25 ————— 5s 215ms/step - accuracy: 0.9877 - loss: 0.0718 - val_accuracy: 0.5900 - val_loss: 3.6389
Epoch 197/200
25/25 ————— 5s 211ms/step - accuracy: 0.9846 - loss: 0.0554 - val_accuracy: 0.6650 - val_loss: 3.2400
Epoch 198/200
25/25 ————— 5s 209ms/step - accuracy: 0.9880 - loss: 0.0415 - val_accuracy: 0.6450 - val_loss: 3.2804
Epoch 199/200
25/25 ————— 5s 197ms/step - accuracy: 0.9821 - loss: 0.0445 - val_accuracy: 0.6550 - val_loss: 3.3964
Epoch 200/200
25/25 ————— 5s 212ms/step - accuracy: 0.9776 - loss: 0.0669 - val_accuracy: 0.6150 - val_loss: 3.7908

```

```
In [8]: print(f"Execution time: {elapsed_time:.2f} seconds")
```

Execution time: 1036.82 seconds

```
In [9]: def append_core_data(score_path, num_cores, elapsed_time):
# Check if the file already exists
file_exists = os.path.exists(score_path)

# Open the file in append mode
with open(score_path, mode='a', newline='') as file:
```

```
writer = csv.writer(file)

# If the file is new, write the header
if not file_exists:
    writer.writerow(["Number of Cores", "Elapsed Time"])

# Write the new data
writer.writerow([num_cores, elapsed_time])
```

```
In [10]: score_path = r"C:\Users\nikhi\OneDrive\Desktop\Final Project\DEEP LEARNING WITH HPSC\core_data.txt"
append_core_data(score_path, number_of_worker, elapsed_time)
```

Import Req Lib

```
In [1]: %matplotlib inline

import shutil
import random
import numpy as np
from warnings import filterwarnings
filterwarnings('ignore')

from tensorflow.keras import layers, regularizers, optimizers
from tensorflow.keras import models
from tensorflow.keras.models import Sequential, Model
from tensorflow.keras.layers import LeakyReLU, Dense, Activation, Flatten, Dropout, BatchNormalization, Conv2D, MaxPooling2D
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import tensorflow as tf

import os
import time
import csv

from matplotlib import figure
```

Define 5 worker

```
In [2]: # Set the number of threads
number_of_worker = 5
os.environ['OMP_NUM_THREADS'] = '5' # OpenMP threads for parallelism
os.environ['TF_NUM_INTEROP_THREADS'] = '5' # Threads for inter-operation parallelism
os.environ['TF_NUM_INTRAOP_THREADS'] = '5' # Threads for intra-operation parallelism

# Confirm TensorFlow is using the specified number of threads
tf.config.threading.set_inter_op_parallelism_threads(number_of_worker)
tf.config.threading.set_intra_op_parallelism_threads(number_of_worker)
```

Train Val data Split

```
In [3]: source_dir = r"C:\Users\nikhi\OneDrive\Desktop\Final Project\DATA\Convert_Audio_File_to_jpg_file"
target_dir = r'genres_train_val_split_data'
split_ratio = 0.8

def Train_Test_Split(source_dir, target_dir, split_ratio):
    # Define source and target directories
    train_dir = os.path.join(target_dir, 'train')
    val_dir = os.path.join(target_dir, 'val')

    # Create target directories if they don't exist
    os.makedirs(train_dir, exist_ok=True)
    os.makedirs(val_dir, exist_ok=True)

    # Get the list of class directories
    classes = [d for d in os.listdir(source_dir) if os.path.isdir(os.path.join(source_dir, d))]

    for class_name in classes:
        # Create class directories in train and val folders
        os.makedirs(os.path.join(train_dir, class_name), exist_ok=True)
        os.makedirs(os.path.join(val_dir, class_name), exist_ok=True)

        # Get list of images in the class directory
        class_dir = os.path.join(source_dir, class_name)
        images = [f for f in os.listdir(class_dir) if os.path.isfile(os.path.join(class_dir, f))]

        # Shuffle the images
        random.shuffle(images)

        # Compute the split point
        split_point = int(len(images) * split_ratio)

        # Split the images into training and validation sets
        train_images = images[:split_point]
        val_images = images[split_point:]
```

```

# Move the images to the respective directories
for img in train_images:
    shutil.copy(os.path.join(class_dir, img), os.path.join(train_dir, class_name, img))

for img in val_images:
    shutil.copy(os.path.join(class_dir, img), os.path.join(val_dir, class_name, img))

print("Data split completed successfully!")

```

In [4]: `Train_Test_Split(source_dir,target_dir,split_ratio)`

Data split completed successfully!

Load the Data

```

In [5]: WIDTH = 64
HEIGHT = 64
BATCH_SIZE = 32
TRAIN_DIR=r'genres_train_val_split_data/train'
val_dir = r'genres_train_val_split_data/val'

# data prep
train_datagen = ImageDataGenerator(
    rescale=1./255.,validation_split=0.25)

train_generator = train_datagen.flow_from_directory(
    TRAIN_DIR,
    target_size=(HEIGHT, WIDTH),
    batch_size=BATCH_SIZE,
    class_mode='categorical')

validation_gen = train_datagen.flow_from_directory(
    val_dir,target_size = (HEIGHT,WIDTH),
    batch_size = BATCH_SIZE,
    class_mode = 'categorical'
)

```

Found 800 images belonging to 10 classes.

Found 200 images belonging to 10 classes.

Model Architecture

```

In [6]: model = Sequential()
model.add(Conv2D(32, (3, 3), padding='same',
    input_shape=(64,64,3)))
model.add(Activation('relu'))
model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Conv2D(64, (3, 3), padding='same'))
model.add(Activation('relu'))
model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.5))
model.add(Conv2D(128, (3, 3), padding='same'))
model.add(Activation('relu'))
model.add(Conv2D(128, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.5))
model.add(Flatten())
model.add(Dense(512))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(10, activation='softmax'))
model.compile(optimizer=RMSprop(learning_rate=0.0005, decay=1e-6),loss="categorical_crossentropy",metrics=["accuracy"])
model.summary()

```

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 64, 64, 32)	896
activation (Activation)	(None, 64, 64, 32)	0
conv2d_1 (Conv2D)	(None, 62, 62, 64)	18,496
activation_1 (Activation)	(None, 62, 62, 64)	0
max_pooling2d (MaxPooling2D)	(None, 31, 31, 64)	0
dropout (Dropout)	(None, 31, 31, 64)	0
conv2d_2 (Conv2D)	(None, 31, 31, 64)	36,928
activation_2 (Activation)	(None, 31, 31, 64)	0
conv2d_3 (Conv2D)	(None, 29, 29, 64)	36,928
activation_3 (Activation)	(None, 29, 29, 64)	0
max_pooling2d_1 (MaxPooling2D)	(None, 14, 14, 64)	0
dropout_1 (Dropout)	(None, 14, 14, 64)	0
conv2d_4 (Conv2D)	(None, 14, 14, 128)	73,856
activation_4 (Activation)	(None, 14, 14, 128)	0
conv2d_5 (Conv2D)	(None, 12, 12, 128)	147,584
activation_5 (Activation)	(None, 12, 12, 128)	0
max_pooling2d_2 (MaxPooling2D)	(None, 6, 6, 128)	0
dropout_2 (Dropout)	(None, 6, 6, 128)	0
flatten (Flatten)	(None, 4608)	0
dense (Dense)	(None, 512)	2,359,808
activation_6 (Activation)	(None, 512)	0
dropout_3 (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 10)	5,130

Total params: 2,679,626 (10.22 MB)

Trainable params: 2,679,626 (10.22 MB)

Non-trainable params: 0 (0.00 B)

```
In [7]: STEP_SIZE_TRAIN=train_generator.n//train_generator.batch_size
# Measure the execution time
start_time = time.time()

model.fit(train_generator,validation_data=validation_gen,epochs=200)

end_time = time.time()
elapsed_time = end_time - start_time
```

Epoch 1/200	25/25	5s	153ms/step	- accuracy: 0.0947	- loss: 2.3395	- val_accuracy: 0.1800	- val_loss: 2.2968
Epoch 2/200	25/25	5s	189ms/step	- accuracy: 0.1267	- loss: 2.2874	- val_accuracy: 0.2250	- val_loss: 2.2684
Epoch 3/200	25/25	5s	194ms/step	- accuracy: 0.1525	- loss: 2.2579	- val_accuracy: 0.1450	- val_loss: 2.2890
Epoch 4/200	25/25	5s	191ms/step	- accuracy: 0.1746	- loss: 2.2096	- val_accuracy: 0.1300	- val_loss: 2.3036
Epoch 5/200	25/25	5s	192ms/step	- accuracy: 0.2015	- loss: 2.1897	- val_accuracy: 0.2700	- val_loss: 1.9919
Epoch 6/200	25/25	5s	193ms/step	- accuracy: 0.2600	- loss: 2.0449	- val_accuracy: 0.3150	- val_loss: 1.9102
Epoch 7/200	25/25	5s	193ms/step	- accuracy: 0.2192	- loss: 2.1385	- val_accuracy: 0.3200	- val_loss: 1.9702
Epoch 8/200	25/25	5s	193ms/step	- accuracy: 0.2778	- loss: 1.9687	- val_accuracy: 0.2950	- val_loss: 2.0048
Epoch 9/200	25/25	5s	194ms/step	- accuracy: 0.2959	- loss: 1.9138	- val_accuracy: 0.3400	- val_loss: 1.8138
Epoch 10/200	25/25	5s	195ms/step	- accuracy: 0.3270	- loss: 1.8565	- val_accuracy: 0.3700	- val_loss: 1.8105
Epoch 11/200	25/25	5s	195ms/step	- accuracy: 0.3472	- loss: 1.7873	- val_accuracy: 0.2900	- val_loss: 1.9588
Epoch 12/200	25/25	5s	197ms/step	- accuracy: 0.3980	- loss: 1.6698	- val_accuracy: 0.4450	- val_loss: 1.6245
Epoch 13/200	25/25	5s	194ms/step	- accuracy: 0.4249	- loss: 1.6145	- val_accuracy: 0.4050	- val_loss: 1.6579
Epoch 14/200	25/25	5s	195ms/step	- accuracy: 0.3869	- loss: 1.6297	- val_accuracy: 0.4300	- val_loss: 1.5762
Epoch 15/200	25/25	5s	197ms/step	- accuracy: 0.3926	- loss: 1.6391	- val_accuracy: 0.4050	- val_loss: 1.6126
Epoch 16/200	25/25	5s	186ms/step	- accuracy: 0.4444	- loss: 1.5492	- val_accuracy: 0.4450	- val_loss: 1.4956
Epoch 17/200	25/25	5s	197ms/step	- accuracy: 0.3875	- loss: 1.6314	- val_accuracy: 0.4400	- val_loss: 1.5098
Epoch 18/200	25/25	5s	192ms/step	- accuracy: 0.4552	- loss: 1.4967	- val_accuracy: 0.4350	- val_loss: 1.5309
Epoch 19/200	25/25	5s	191ms/step	- accuracy: 0.4820	- loss: 1.3829	- val_accuracy: 0.4600	- val_loss: 1.4706
Epoch 20/200	25/25	5s	194ms/step	- accuracy: 0.4636	- loss: 1.4114	- val_accuracy: 0.4750	- val_loss: 1.4325
Epoch 21/200	25/25	5s	192ms/step	- accuracy: 0.5140	- loss: 1.3883	- val_accuracy: 0.4900	- val_loss: 1.4102
Epoch 22/200	25/25	5s	185ms/step	- accuracy: 0.5524	- loss: 1.2626	- val_accuracy: 0.4800	- val_loss: 1.4580
Epoch 23/200	25/25	5s	195ms/step	- accuracy: 0.5119	- loss: 1.3195	- val_accuracy: 0.5000	- val_loss: 1.3673
Epoch 24/200	25/25	5s	186ms/step	- accuracy: 0.5571	- loss: 1.2439	- val_accuracy: 0.4900	- val_loss: 1.3707
Epoch 25/200	25/25	5s	190ms/step	- accuracy: 0.5772	- loss: 1.1844	- val_accuracy: 0.5100	- val_loss: 1.3961
Epoch 26/200	25/25	5s	187ms/step	- accuracy: 0.5738	- loss: 1.1956	- val_accuracy: 0.5150	- val_loss: 1.3461
Epoch 27/200	25/25	5s	188ms/step	- accuracy: 0.5962	- loss: 1.1263	- val_accuracy: 0.5350	- val_loss: 1.3342
Epoch 28/200	25/25	5s	191ms/step	- accuracy: 0.5756	- loss: 1.1709	- val_accuracy: 0.5850	- val_loss: 1.2785
Epoch 29/200	25/25	5s	194ms/step	- accuracy: 0.6157	- loss: 1.0428	- val_accuracy: 0.5050	- val_loss: 1.3166
Epoch 30/200	25/25	5s	188ms/step	- accuracy: 0.5870	- loss: 1.1603	- val_accuracy: 0.5200	- val_loss: 1.3354
Epoch 31/200	25/25	5s	190ms/step	- accuracy: 0.5897	- loss: 1.0747	- val_accuracy: 0.5750	- val_loss: 1.3224
Epoch 32/200	25/25	5s	187ms/step	- accuracy: 0.6535	- loss: 0.9584	- val_accuracy: 0.5150	- val_loss: 1.3154
Epoch 33/200	25/25	5s	190ms/step	- accuracy: 0.6701	- loss: 0.9027	- val_accuracy: 0.5900	- val_loss: 1.2023
Epoch 34/200	25/25	5s	187ms/step	- accuracy: 0.6354	- loss: 0.9593	- val_accuracy: 0.6050	- val_loss: 1.1918
Epoch 35/200	25/25	5s	192ms/step	- accuracy: 0.6474	- loss: 0.9642	- val_accuracy: 0.5800	- val_loss: 1.2895
Epoch 36/200	25/25	5s	200ms/step	- accuracy: 0.7099	- loss: 0.7981	- val_accuracy: 0.5900	- val_loss: 1.2998
Epoch 37/200	25/25	5s	182ms/step	- accuracy: 0.6821	- loss: 0.8373	- val_accuracy: 0.5450	- val_loss: 1.3157
Epoch 38/200	25/25	5s	198ms/step	- accuracy: 0.7395	- loss: 0.7698	- val_accuracy: 0.5550	- val_loss: 1.3241
Epoch 39/200	25/25	5s	194ms/step	- accuracy: 0.7437	- loss: 0.7579	- val_accuracy: 0.5700	- val_loss: 1.3051
Epoch 40/200	25/25	5s	200ms/step	- accuracy: 0.7123	- loss: 0.7496	- val_accuracy: 0.6300	- val_loss: 1.2596
Epoch 41/200	25/25	5s	193ms/step	- accuracy: 0.7576	- loss: 0.6787	- val_accuracy: 0.5650	- val_loss: 1.4545

Epoch 42/200	25/25	5s	204ms/step	- accuracy: 0.7592	- loss: 0.7017	- val_accuracy: 0.5650	- val_loss: 1.4580
Epoch 43/200	25/25	5s	196ms/step	- accuracy: 0.7450	- loss: 0.6825	- val_accuracy: 0.5950	- val_loss: 1.3154
Epoch 44/200	25/25	5s	200ms/step	- accuracy: 0.8020	- loss: 0.5688	- val_accuracy: 0.5600	- val_loss: 1.3848
Epoch 45/200	25/25	5s	197ms/step	- accuracy: 0.8079	- loss: 0.5317	- val_accuracy: 0.5700	- val_loss: 1.4590
Epoch 46/200	25/25	5s	193ms/step	- accuracy: 0.7713	- loss: 0.6108	- val_accuracy: 0.5650	- val_loss: 1.3786
Epoch 47/200	25/25	5s	192ms/step	- accuracy: 0.7668	- loss: 0.6010	- val_accuracy: 0.5900	- val_loss: 1.4130
Epoch 48/200	25/25	5s	197ms/step	- accuracy: 0.8221	- loss: 0.4975	- val_accuracy: 0.5700	- val_loss: 1.4629
Epoch 49/200	25/25	5s	197ms/step	- accuracy: 0.7979	- loss: 0.5379	- val_accuracy: 0.5650	- val_loss: 1.4927
Epoch 50/200	25/25	5s	191ms/step	- accuracy: 0.8335	- loss: 0.4722	- val_accuracy: 0.6000	- val_loss: 1.5870
Epoch 51/200	25/25	5s	191ms/step	- accuracy: 0.8321	- loss: 0.4891	- val_accuracy: 0.5650	- val_loss: 1.8292
Epoch 52/200	25/25	5s	194ms/step	- accuracy: 0.8426	- loss: 0.4643	- val_accuracy: 0.6250	- val_loss: 1.5597
Epoch 53/200	25/25	5s	191ms/step	- accuracy: 0.8810	- loss: 0.3997	- val_accuracy: 0.5900	- val_loss: 1.5873
Epoch 54/200	25/25	5s	188ms/step	- accuracy: 0.8325	- loss: 0.4602	- val_accuracy: 0.6000	- val_loss: 1.7301
Epoch 55/200	25/25	5s	196ms/step	- accuracy: 0.8486	- loss: 0.4165	- val_accuracy: 0.5950	- val_loss: 1.6603
Epoch 56/200	25/25	5s	194ms/step	- accuracy: 0.8677	- loss: 0.3930	- val_accuracy: 0.6050	- val_loss: 1.5699
Epoch 57/200	25/25	5s	189ms/step	- accuracy: 0.8873	- loss: 0.3440	- val_accuracy: 0.6050	- val_loss: 1.5525
Epoch 58/200	25/25	5s	199ms/step	- accuracy: 0.8883	- loss: 0.3339	- val_accuracy: 0.5850	- val_loss: 1.8909
Epoch 59/200	25/25	5s	185ms/step	- accuracy: 0.8920	- loss: 0.2942	- val_accuracy: 0.5700	- val_loss: 1.7802
Epoch 60/200	25/25	5s	200ms/step	- accuracy: 0.9011	- loss: 0.3193	- val_accuracy: 0.5950	- val_loss: 2.0626
Epoch 61/200	25/25	5s	190ms/step	- accuracy: 0.8948	- loss: 0.3518	- val_accuracy: 0.6150	- val_loss: 1.9260
Epoch 62/200	25/25	5s	191ms/step	- accuracy: 0.8986	- loss: 0.2625	- val_accuracy: 0.5950	- val_loss: 1.8665
Epoch 63/200	25/25	5s	190ms/step	- accuracy: 0.9059	- loss: 0.2910	- val_accuracy: 0.6100	- val_loss: 1.7793
Epoch 64/200	25/25	5s	191ms/step	- accuracy: 0.9151	- loss: 0.2510	- val_accuracy: 0.6300	- val_loss: 1.8918
Epoch 65/200	25/25	5s	187ms/step	- accuracy: 0.9331	- loss: 0.2084	- val_accuracy: 0.5800	- val_loss: 1.9621
Epoch 66/200	25/25	5s	194ms/step	- accuracy: 0.9136	- loss: 0.2767	- val_accuracy: 0.6150	- val_loss: 1.8002
Epoch 67/200	25/25	5s	195ms/step	- accuracy: 0.9290	- loss: 0.2592	- val_accuracy: 0.6050	- val_loss: 2.0399
Epoch 68/200	25/25	5s	195ms/step	- accuracy: 0.8988	- loss: 0.2698	- val_accuracy: 0.5950	- val_loss: 2.1498
Epoch 69/200	25/25	5s	190ms/step	- accuracy: 0.9527	- loss: 0.1518	- val_accuracy: 0.6100	- val_loss: 2.1151
Epoch 70/200	25/25	5s	188ms/step	- accuracy: 0.9299	- loss: 0.1859	- val_accuracy: 0.6050	- val_loss: 2.1399
Epoch 71/200	25/25	5s	186ms/step	- accuracy: 0.9354	- loss: 0.1745	- val_accuracy: 0.5850	- val_loss: 2.1911
Epoch 72/200	25/25	5s	196ms/step	- accuracy: 0.9467	- loss: 0.1526	- val_accuracy: 0.5800	- val_loss: 2.0856
Epoch 73/200	25/25	5s	191ms/step	- accuracy: 0.9430	- loss: 0.1523	- val_accuracy: 0.5550	- val_loss: 2.4861
Epoch 74/200	25/25	5s	188ms/step	- accuracy: 0.9512	- loss: 0.1692	- val_accuracy: 0.5850	- val_loss: 2.1487
Epoch 75/200	25/25	5s	194ms/step	- accuracy: 0.9661	- loss: 0.1040	- val_accuracy: 0.6400	- val_loss: 2.2880
Epoch 76/200	25/25	5s	192ms/step	- accuracy: 0.9370	- loss: 0.1945	- val_accuracy: 0.6150	- val_loss: 2.5149
Epoch 77/200	25/25	5s	188ms/step	- accuracy: 0.9633	- loss: 0.2527	- val_accuracy: 0.6050	- val_loss: 2.1542
Epoch 78/200	25/25	5s	194ms/step	- accuracy: 0.9281	- loss: 0.2165	- val_accuracy: 0.5850	- val_loss: 2.1198
Epoch 79/200	25/25	5s	190ms/step	- accuracy: 0.9671	- loss: 0.1049	- val_accuracy: 0.5950	- val_loss: 2.1969
Epoch 80/200	25/25	5s	192ms/step	- accuracy: 0.9430	- loss: 0.1505	- val_accuracy: 0.5700	- val_loss: 2.4307
Epoch 81/200	25/25	5s	185ms/step	- accuracy: 0.9560	- loss: 0.1293	- val_accuracy: 0.5650	- val_loss: 2.3232
Epoch 82/200	25/25	5s	186ms/step	- accuracy: 0.9472	- loss: 0.1393	- val_accuracy: 0.5800	- val_loss: 2.6178

Epoch 83/200
25/25 ————— 5s 199ms/step - accuracy: 0.9441 - loss: 0.1490 - val_accuracy: 0.5950 - val_loss: 2.2539
Epoch 84/200
25/25 ————— 5s 189ms/step - accuracy: 0.9621 - loss: 0.1124 - val_accuracy: 0.5850 - val_loss: 2.4427
Epoch 85/200
25/25 ————— 5s 195ms/step - accuracy: 0.9460 - loss: 0.1516 - val_accuracy: 0.6250 - val_loss: 2.1418
Epoch 86/200
25/25 ————— 5s 200ms/step - accuracy: 0.9625 - loss: 0.1522 - val_accuracy: 0.6050 - val_loss: 2.3322
Epoch 87/200
25/25 ————— 5s 189ms/step - accuracy: 0.9580 - loss: 0.1109 - val_accuracy: 0.6100 - val_loss: 2.1526
Epoch 88/200
25/25 ————— 5s 195ms/step - accuracy: 0.9683 - loss: 0.1126 - val_accuracy: 0.6200 - val_loss: 2.5570
Epoch 89/200
25/25 ————— 5s 189ms/step - accuracy: 0.9610 - loss: 0.1214 - val_accuracy: 0.5750 - val_loss: 2.4782
Epoch 90/200
25/25 ————— 5s 196ms/step - accuracy: 0.9599 - loss: 0.1022 - val_accuracy: 0.5550 - val_loss: 2.3816
Epoch 91/200
25/25 ————— 5s 196ms/step - accuracy: 0.9712 - loss: 0.1005 - val_accuracy: 0.5700 - val_loss: 2.6433
Epoch 92/200
25/25 ————— 5s 186ms/step - accuracy: 0.9472 - loss: 0.1302 - val_accuracy: 0.5750 - val_loss: 3.0499
Epoch 93/200
25/25 ————— 5s 193ms/step - accuracy: 0.9653 - loss: 0.1075 - val_accuracy: 0.5850 - val_loss: 2.9033
Epoch 94/200
25/25 ————— 5s 194ms/step - accuracy: 0.9574 - loss: 0.1204 - val_accuracy: 0.5850 - val_loss: 2.7043
Epoch 95/200
25/25 ————— 5s 190ms/step - accuracy: 0.9684 - loss: 0.1105 - val_accuracy: 0.6000 - val_loss: 2.6137
Epoch 96/200
25/25 ————— 5s 185ms/step - accuracy: 0.9573 - loss: 0.1077 - val_accuracy: 0.6000 - val_loss: 2.3739
Epoch 97/200
25/25 ————— 5s 193ms/step - accuracy: 0.9490 - loss: 0.1130 - val_accuracy: 0.6100 - val_loss: 2.3640
Epoch 98/200
25/25 ————— 5s 194ms/step - accuracy: 0.9790 - loss: 0.0706 - val_accuracy: 0.5750 - val_loss: 3.0800
Epoch 99/200
25/25 ————— 5s 194ms/step - accuracy: 0.9857 - loss: 0.0605 - val_accuracy: 0.6050 - val_loss: 2.4089
Epoch 100/200
25/25 ————— 5s 194ms/step - accuracy: 0.9794 - loss: 0.0811 - val_accuracy: 0.5750 - val_loss: 2.9083
Epoch 101/200
25/25 ————— 5s 194ms/step - accuracy: 0.9759 - loss: 0.0625 - val_accuracy: 0.6000 - val_loss: 3.0784
Epoch 102/200
25/25 ————— 5s 190ms/step - accuracy: 0.9781 - loss: 0.0984 - val_accuracy: 0.6300 - val_loss: 2.7971
Epoch 103/200
25/25 ————— 5s 190ms/step - accuracy: 0.9554 - loss: 0.1220 - val_accuracy: 0.6150 - val_loss: 2.3903
Epoch 104/200
25/25 ————— 5s 187ms/step - accuracy: 0.9743 - loss: 0.0762 - val_accuracy: 0.5700 - val_loss: 2.5217
Epoch 105/200
25/25 ————— 5s 193ms/step - accuracy: 0.9893 - loss: 0.0418 - val_accuracy: 0.5900 - val_loss: 2.6067
Epoch 106/200
25/25 ————— 5s 192ms/step - accuracy: 0.9804 - loss: 0.0651 - val_accuracy: 0.6000 - val_loss: 2.2410
Epoch 107/200
25/25 ————— 5s 187ms/step - accuracy: 0.9545 - loss: 0.0943 - val_accuracy: 0.5700 - val_loss: 2.6078
Epoch 108/200
25/25 ————— 5s 190ms/step - accuracy: 0.9713 - loss: 0.0938 - val_accuracy: 0.5700 - val_loss: 2.6523
Epoch 109/200
25/25 ————— 5s 194ms/step - accuracy: 0.9663 - loss: 0.1164 - val_accuracy: 0.6000 - val_loss: 2.6464
Epoch 110/200
25/25 ————— 5s 194ms/step - accuracy: 0.9702 - loss: 0.0860 - val_accuracy: 0.5700 - val_loss: 2.7814
Epoch 111/200
25/25 ————— 5s 196ms/step - accuracy: 0.9639 - loss: 0.1292 - val_accuracy: 0.6100 - val_loss: 2.4844
Epoch 112/200
25/25 ————— 5s 188ms/step - accuracy: 0.9848 - loss: 0.0520 - val_accuracy: 0.6100 - val_loss: 3.0841
Epoch 113/200
25/25 ————— 5s 196ms/step - accuracy: 0.9740 - loss: 0.0715 - val_accuracy: 0.5800 - val_loss: 2.8770
Epoch 114/200
25/25 ————— 5s 199ms/step - accuracy: 0.9860 - loss: 0.0509 - val_accuracy: 0.6000 - val_loss: 3.2372
Epoch 115/200
25/25 ————— 5s 192ms/step - accuracy: 0.9700 - loss: 0.0779 - val_accuracy: 0.5750 - val_loss: 3.0240
Epoch 116/200
25/25 ————— 5s 194ms/step - accuracy: 0.9712 - loss: 0.1071 - val_accuracy: 0.5850 - val_loss: 2.9968
Epoch 117/200
25/25 ————— 5s 191ms/step - accuracy: 0.9741 - loss: 0.0612 - val_accuracy: 0.6000 - val_loss: 3.3724
Epoch 118/200
25/25 ————— 5s 190ms/step - accuracy: 0.9812 - loss: 0.0701 - val_accuracy: 0.6150 - val_loss: 2.8990
Epoch 119/200
25/25 ————— 5s 190ms/step - accuracy: 0.9770 - loss: 0.0852 - val_accuracy: 0.5900 - val_loss: 2.3540
Epoch 120/200
25/25 ————— 5s 188ms/step - accuracy: 0.9712 - loss: 0.0952 - val_accuracy: 0.6200 - val_loss: 2.4252
Epoch 121/200
25/25 ————— 5s 189ms/step - accuracy: 0.9906 - loss: 0.0291 - val_accuracy: 0.5500 - val_loss: 3.8702
Epoch 122/200
25/25 ————— 5s 191ms/step - accuracy: 0.9757 - loss: 0.0829 - val_accuracy: 0.6100 - val_loss: 3.0957
Epoch 123/200
25/25 ————— 5s 193ms/step - accuracy: 0.9720 - loss: 0.0825 - val_accuracy: 0.5850 - val_loss: 3.0245

Epoch 124/200
25/25 ————— 5s 197ms/step - accuracy: 0.9676 - loss: 0.1079 - val_accuracy: 0.5900 - val_loss: 3.5802

Epoch 125/200
25/25 ————— 5s 191ms/step - accuracy: 0.9640 - loss: 0.1216 - val_accuracy: 0.5550 - val_loss: 2.6271

Epoch 126/200
25/25 ————— 5s 200ms/step - accuracy: 0.9681 - loss: 0.1053 - val_accuracy: 0.5950 - val_loss: 3.2023

Epoch 127/200
25/25 ————— 5s 195ms/step - accuracy: 0.9804 - loss: 0.0468 - val_accuracy: 0.5600 - val_loss: 2.7726

Epoch 128/200
25/25 ————— 5s 191ms/step - accuracy: 0.9893 - loss: 0.0385 - val_accuracy: 0.5950 - val_loss: 3.2134

Epoch 129/200
25/25 ————— 5s 192ms/step - accuracy: 0.9809 - loss: 0.0590 - val_accuracy: 0.5950 - val_loss: 2.9174

Epoch 130/200
25/25 ————— 5s 200ms/step - accuracy: 0.9780 - loss: 0.0603 - val_accuracy: 0.5900 - val_loss: 2.7604

Epoch 131/200
25/25 ————— 5s 194ms/step - accuracy: 0.9894 - loss: 0.0479 - val_accuracy: 0.5800 - val_loss: 3.1874

Epoch 132/200
25/25 ————— 5s 196ms/step - accuracy: 0.9859 - loss: 0.0606 - val_accuracy: 0.5750 - val_loss: 3.1136

Epoch 133/200
25/25 ————— 5s 194ms/step - accuracy: 0.9753 - loss: 0.0876 - val_accuracy: 0.6050 - val_loss: 2.9528

Epoch 134/200
25/25 ————— 5s 194ms/step - accuracy: 0.9896 - loss: 0.0464 - val_accuracy: 0.5900 - val_loss: 3.2710

Epoch 135/200
25/25 ————— 5s 192ms/step - accuracy: 0.9720 - loss: 0.1045 - val_accuracy: 0.5950 - val_loss: 2.7128

Epoch 136/200
25/25 ————— 5s 191ms/step - accuracy: 0.9933 - loss: 0.0282 - val_accuracy: 0.6050 - val_loss: 3.0566

Epoch 137/200
25/25 ————— 5s 195ms/step - accuracy: 0.9833 - loss: 0.0745 - val_accuracy: 0.5650 - val_loss: 3.8128

Epoch 138/200
25/25 ————— 5s 191ms/step - accuracy: 0.9689 - loss: 0.0789 - val_accuracy: 0.6200 - val_loss: 2.9367

Epoch 139/200
25/25 ————— 5s 195ms/step - accuracy: 0.9794 - loss: 0.0616 - val_accuracy: 0.5650 - val_loss: 2.7952

Epoch 140/200
25/25 ————— 5s 201ms/step - accuracy: 0.9961 - loss: 0.0180 - val_accuracy: 0.5850 - val_loss: 3.2651

Epoch 141/200
25/25 ————— 5s 195ms/step - accuracy: 0.9910 - loss: 0.0361 - val_accuracy: 0.5300 - val_loss: 3.2572

Epoch 142/200
25/25 ————— 5s 191ms/step - accuracy: 0.9853 - loss: 0.0725 - val_accuracy: 0.5800 - val_loss: 2.8074

Epoch 143/200
25/25 ————— 5s 189ms/step - accuracy: 0.9817 - loss: 0.0475 - val_accuracy: 0.5700 - val_loss: 3.0449

Epoch 144/200
25/25 ————— 5s 196ms/step - accuracy: 0.9891 - loss: 0.0474 - val_accuracy: 0.5900 - val_loss: 3.5814

Epoch 145/200
25/25 ————— 5s 186ms/step - accuracy: 0.9811 - loss: 0.0532 - val_accuracy: 0.5900 - val_loss: 3.5755

Epoch 146/200
25/25 ————— 5s 187ms/step - accuracy: 0.9628 - loss: 0.1155 - val_accuracy: 0.5650 - val_loss: 3.1388

Epoch 147/200
25/25 ————— 5s 198ms/step - accuracy: 0.9839 - loss: 0.0366 - val_accuracy: 0.6000 - val_loss: 3.2828

Epoch 148/200
25/25 ————— 5s 187ms/step - accuracy: 0.9905 - loss: 0.0346 - val_accuracy: 0.5800 - val_loss: 2.7950

Epoch 149/200
25/25 ————— 5s 192ms/step - accuracy: 0.9843 - loss: 0.0378 - val_accuracy: 0.5900 - val_loss: 3.3712

Epoch 150/200
25/25 ————— 5s 195ms/step - accuracy: 0.9826 - loss: 0.0814 - val_accuracy: 0.5600 - val_loss: 3.1906

Epoch 151/200
25/25 ————— 5s 196ms/step - accuracy: 0.9868 - loss: 0.0411 - val_accuracy: 0.6100 - val_loss: 3.3522

Epoch 152/200
25/25 ————— 5s 196ms/step - accuracy: 0.9826 - loss: 0.0764 - val_accuracy: 0.5850 - val_loss: 3.1451

Epoch 153/200
25/25 ————— 5s 192ms/step - accuracy: 0.9711 - loss: 0.0864 - val_accuracy: 0.5700 - val_loss: 3.3927

Epoch 154/200
25/25 ————— 5s 193ms/step - accuracy: 0.9854 - loss: 0.0434 - val_accuracy: 0.5650 - val_loss: 3.5499

Epoch 155/200
25/25 ————— 5s 202ms/step - accuracy: 0.9890 - loss: 0.0358 - val_accuracy: 0.5900 - val_loss: 3.3332

Epoch 156/200
25/25 ————— 5s 186ms/step - accuracy: 0.9874 - loss: 0.0431 - val_accuracy: 0.5850 - val_loss: 3.6944

Epoch 157/200
25/25 ————— 5s 192ms/step - accuracy: 0.9905 - loss: 0.0310 - val_accuracy: 0.5700 - val_loss: 3.6082

Epoch 158/200
25/25 ————— 5s 192ms/step - accuracy: 0.9814 - loss: 0.0555 - val_accuracy: 0.5950 - val_loss: 3.4962

Epoch 159/200
25/25 ————— 5s 189ms/step - accuracy: 0.9585 - loss: 0.0979 - val_accuracy: 0.6000 - val_loss: 3.2705

Epoch 160/200
25/25 ————— 5s 192ms/step - accuracy: 0.9963 - loss: 0.0190 - val_accuracy: 0.5800 - val_loss: 3.3270

Epoch 161/200
25/25 ————— 5s 190ms/step - accuracy: 0.9895 - loss: 0.0377 - val_accuracy: 0.5650 - val_loss: 2.9575

Epoch 162/200
25/25 ————— 5s 194ms/step - accuracy: 0.9772 - loss: 0.0976 - val_accuracy: 0.5950 - val_loss: 3.2801

Epoch 163/200
25/25 ————— 5s 189ms/step - accuracy: 0.9748 - loss: 0.0613 - val_accuracy: 0.5600 - val_loss: 2.9793

Epoch 164/200
25/25 ————— 5s 193ms/step - accuracy: 0.9830 - loss: 0.0523 - val_accuracy: 0.6050 - val_loss: 3.0790

```

Epoch 165/200
25/25 ————— 5s 192ms/step - accuracy: 0.9915 - loss: 0.0286 - val_accuracy: 0.6050 - val_loss: 3.7859
Epoch 166/200
25/25 ————— 5s 186ms/step - accuracy: 0.9891 - loss: 0.0436 - val_accuracy: 0.5500 - val_loss: 3.0983
Epoch 167/200
25/25 ————— 5s 207ms/step - accuracy: 0.9789 - loss: 0.0917 - val_accuracy: 0.5500 - val_loss: 3.3727
Epoch 168/200
25/25 ————— 5s 197ms/step - accuracy: 0.9814 - loss: 0.0524 - val_accuracy: 0.5650 - val_loss: 3.5163
Epoch 169/200
25/25 ————— 5s 191ms/step - accuracy: 0.9922 - loss: 0.0449 - val_accuracy: 0.5700 - val_loss: 3.3930
Epoch 170/200
25/25 ————— 5s 188ms/step - accuracy: 0.9858 - loss: 0.0606 - val_accuracy: 0.5400 - val_loss: 3.3500
Epoch 171/200
25/25 ————— 5s 187ms/step - accuracy: 0.9942 - loss: 0.0179 - val_accuracy: 0.5700 - val_loss: 3.6854
Epoch 172/200
25/25 ————— 5s 202ms/step - accuracy: 0.9836 - loss: 0.0725 - val_accuracy: 0.6050 - val_loss: 3.5931
Epoch 173/200
25/25 ————— 5s 194ms/step - accuracy: 0.9918 - loss: 0.0542 - val_accuracy: 0.5700 - val_loss: 3.5743
Epoch 174/200
25/25 ————— 5s 194ms/step - accuracy: 0.9908 - loss: 0.0405 - val_accuracy: 0.5850 - val_loss: 3.6747
Epoch 175/200
25/25 ————— 5s 193ms/step - accuracy: 0.9892 - loss: 0.0261 - val_accuracy: 0.6050 - val_loss: 3.7958
Epoch 176/200
25/25 ————— 5s 189ms/step - accuracy: 0.9804 - loss: 0.0539 - val_accuracy: 0.5300 - val_loss: 3.9636
Epoch 177/200
25/25 ————— 5s 195ms/step - accuracy: 0.9904 - loss: 0.0310 - val_accuracy: 0.6100 - val_loss: 3.5858
Epoch 178/200
25/25 ————— 5s 192ms/step - accuracy: 0.9821 - loss: 0.0586 - val_accuracy: 0.6000 - val_loss: 3.3208
Epoch 179/200
25/25 ————— 5s 190ms/step - accuracy: 0.9860 - loss: 0.0452 - val_accuracy: 0.5850 - val_loss: 3.4362
Epoch 180/200
25/25 ————— 5s 193ms/step - accuracy: 0.9893 - loss: 0.0315 - val_accuracy: 0.6050 - val_loss: 3.4990
Epoch 181/200
25/25 ————— 5s 192ms/step - accuracy: 0.9871 - loss: 0.0347 - val_accuracy: 0.5950 - val_loss: 3.3163
Epoch 182/200
25/25 ————— 5s 188ms/step - accuracy: 0.9882 - loss: 0.0513 - val_accuracy: 0.5800 - val_loss: 3.4056
Epoch 183/200
25/25 ————— 5s 185ms/step - accuracy: 0.9828 - loss: 0.0370 - val_accuracy: 0.5450 - val_loss: 3.7495
Epoch 184/200
25/25 ————— 5s 189ms/step - accuracy: 0.9824 - loss: 0.0682 - val_accuracy: 0.6000 - val_loss: 3.6786
Epoch 185/200
25/25 ————— 5s 192ms/step - accuracy: 0.9805 - loss: 0.0477 - val_accuracy: 0.5550 - val_loss: 3.7180
Epoch 186/200
25/25 ————— 5s 199ms/step - accuracy: 0.9865 - loss: 0.0487 - val_accuracy: 0.5350 - val_loss: 3.4560
Epoch 187/200
25/25 ————— 5s 200ms/step - accuracy: 0.9870 - loss: 0.0400 - val_accuracy: 0.5700 - val_loss: 3.4295
Epoch 188/200
25/25 ————— 5s 194ms/step - accuracy: 0.9863 - loss: 0.0447 - val_accuracy: 0.5650 - val_loss: 3.5240
Epoch 189/200
25/25 ————— 5s 195ms/step - accuracy: 0.9796 - loss: 0.0755 - val_accuracy: 0.5950 - val_loss: 3.6435
Epoch 190/200
25/25 ————— 5s 193ms/step - accuracy: 0.9626 - loss: 0.1290 - val_accuracy: 0.5600 - val_loss: 3.2255
Epoch 191/200
25/25 ————— 5s 189ms/step - accuracy: 0.9832 - loss: 0.0353 - val_accuracy: 0.5250 - val_loss: 3.6364
Epoch 192/200
25/25 ————— 5s 196ms/step - accuracy: 0.9832 - loss: 0.0420 - val_accuracy: 0.5600 - val_loss: 3.4615
Epoch 193/200
25/25 ————— 5s 192ms/step - accuracy: 0.9907 - loss: 0.0301 - val_accuracy: 0.5600 - val_loss: 4.0156
Epoch 194/200
25/25 ————— 5s 194ms/step - accuracy: 0.9744 - loss: 0.0831 - val_accuracy: 0.5800 - val_loss: 3.4824
Epoch 195/200
25/25 ————— 5s 193ms/step - accuracy: 0.9933 - loss: 0.0319 - val_accuracy: 0.5500 - val_loss: 4.6713
Epoch 196/200
25/25 ————— 5s 196ms/step - accuracy: 0.9767 - loss: 0.1003 - val_accuracy: 0.5400 - val_loss: 5.9740
Epoch 197/200
25/25 ————— 5s 193ms/step - accuracy: 0.9628 - loss: 0.2369 - val_accuracy: 0.5600 - val_loss: 3.9092
Epoch 198/200
25/25 ————— 5s 192ms/step - accuracy: 0.9807 - loss: 0.0682 - val_accuracy: 0.5750 - val_loss: 4.1215
Epoch 199/200
25/25 ————— 5s 190ms/step - accuracy: 0.9963 - loss: 0.0230 - val_accuracy: 0.5950 - val_loss: 3.8465
Epoch 200/200
25/25 ————— 5s 189ms/step - accuracy: 0.9853 - loss: 0.0466 - val_accuracy: 0.5800 - val_loss: 3.8862

```

```
In [8]: print(f"Execution time: {elapsed_time:.2f} seconds")
```

Execution time: 967.01 seconds

```
In [9]: def append_core_data(score_path, num_cores, elapsed_time):
        # Check if the file already exists
        file_exists = os.path.exists(score_path)

        # Open the file in append mode
        with open(score_path, mode='a', newline='') as file:
```

```
writer = csv.writer(file)

# If the file is new, write the header
if not file_exists:
    writer.writerow(["Number of Cores", "Elapsed Time"])

# Write the new data
writer.writerow([num_cores, elapsed_time])
```

```
In [10]: score_path = r"C:\Users\nikhi\OneDrive\Desktop\Final Project\DEEP LEARNING WITH HPSC\core_data.txt"
append_core_data(score_path, number_of_worker, elapsed_time)
```

Import Req Lib

```
In [1]: %matplotlib inline

import shutil
import random
import numpy as np
from warnings import filterwarnings
filterwarnings('ignore')

from tensorflow.keras import layers, regularizers, optimizers
from tensorflow.keras import models
from tensorflow.keras.models import Sequential, Model
from tensorflow.keras.layers import LeakyReLU, Dense, Activation, Flatten, Dropout, BatchNormalization, Conv2D, MaxPooling2D
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import tensorflow as tf

import os
import time
import csv

from matplotlib import figure
```

Define 6 worker

```
In [2]: # Set the number of threads
number_of_worker = 6
os.environ['OMP_NUM_THREADS'] = '6' # OpenMP threads for parallelism
os.environ['TF_NUM_INTEROP_THREADS'] = '6' # Threads for inter-operation parallelism
os.environ['TF_NUM_INTRAOP_THREADS'] = '6' # Threads for intra-operation parallelism

# Confirm TensorFlow is using the specified number of threads
tf.config.threading.set_inter_op_parallelism_threads(number_of_worker)
tf.config.threading.set_intra_op_parallelism_threads(number_of_worker)
```

Train Val data Split

```
In [3]: source_dir = r"C:\Users\nikhi\OneDrive\Desktop\Final Project\DATA\Convert_Audio_File_to_jpg_file"
target_dir = r'genres_train_val_split_data'
split_ratio = 0.8

def Train_Test_Split(source_dir, target_dir, split_ratio):
    # Define source and target directories
    train_dir = os.path.join(target_dir, 'train')
    val_dir = os.path.join(target_dir, 'val')

    # Create target directories if they don't exist
    os.makedirs(train_dir, exist_ok=True)
    os.makedirs(val_dir, exist_ok=True)

    # Get the list of class directories
    classes = [d for d in os.listdir(source_dir) if os.path.isdir(os.path.join(source_dir, d))]

    for class_name in classes:
        # Create class directories in train and val folders
        os.makedirs(os.path.join(train_dir, class_name), exist_ok=True)
        os.makedirs(os.path.join(val_dir, class_name), exist_ok=True)

        # Get list of images in the class directory
        class_dir = os.path.join(source_dir, class_name)
        images = [f for f in os.listdir(class_dir) if os.path.isfile(os.path.join(class_dir, f))]

        # Shuffle the images
        random.shuffle(images)

        # Compute the split point
        split_point = int(len(images) * split_ratio)

        # Split the images into training and validation sets
        train_images = images[:split_point]
        val_images = images[split_point:]
```

```

# Move the images to the respective directories
for img in train_images:
    shutil.copy(os.path.join(class_dir, img), os.path.join(train_dir, class_name, img))

for img in val_images:
    shutil.copy(os.path.join(class_dir, img), os.path.join(val_dir, class_name, img))

print("Data split completed successfully!")

```

In [4]: `Train_Test_Split(source_dir,target_dir,split_ratio)`

Data split completed successfully!

Load the Data

```

In [5]: WIDTH = 64
HEIGHT = 64
BATCH_SIZE = 32
TRAIN_DIR=r'genres_train_val_split_data/train'
val_dir = r'genres_train_val_split_data/val'

# data prep
train_datagen = ImageDataGenerator(
    rescale=1./255.,validation_split=0.25)

train_generator = train_datagen.flow_from_directory(
    TRAIN_DIR,
    target_size=(HEIGHT, WIDTH),
    batch_size=BATCH_SIZE,
    class_mode='categorical')

validation_gen = train_datagen.flow_from_directory(
    val_dir,target_size = (HEIGHT,WIDTH),
    batch_size = BATCH_SIZE,
    class_mode = 'categorical'
)

```

Found 800 images belonging to 10 classes.

Found 200 images belonging to 10 classes.

Model Architecture

```

In [6]: model = Sequential()
model.add(Conv2D(32, (3, 3), padding='same',
    input_shape=(64,64,3)))
model.add(Activation('relu'))
model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Conv2D(64, (3, 3), padding='same'))
model.add(Activation('relu'))
model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.5))
model.add(Conv2D(128, (3, 3), padding='same'))
model.add(Activation('relu'))
model.add(Conv2D(128, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.5))
model.add(Flatten())
model.add(Dense(512))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(10, activation='softmax'))
model.compile(optimizer=RMSprop(learning_rate=0.0005, decay=1e-6),loss="categorical_crossentropy",metrics=["accuracy"])
model.summary()

```

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 64, 64, 32)	896
activation (Activation)	(None, 64, 64, 32)	0
conv2d_1 (Conv2D)	(None, 62, 62, 64)	18,496
activation_1 (Activation)	(None, 62, 62, 64)	0
max_pooling2d (MaxPooling2D)	(None, 31, 31, 64)	0
dropout (Dropout)	(None, 31, 31, 64)	0
conv2d_2 (Conv2D)	(None, 31, 31, 64)	36,928
activation_2 (Activation)	(None, 31, 31, 64)	0
conv2d_3 (Conv2D)	(None, 29, 29, 64)	36,928
activation_3 (Activation)	(None, 29, 29, 64)	0
max_pooling2d_1 (MaxPooling2D)	(None, 14, 14, 64)	0
dropout_1 (Dropout)	(None, 14, 14, 64)	0
conv2d_4 (Conv2D)	(None, 14, 14, 128)	73,856
activation_4 (Activation)	(None, 14, 14, 128)	0
conv2d_5 (Conv2D)	(None, 12, 12, 128)	147,584
activation_5 (Activation)	(None, 12, 12, 128)	0
max_pooling2d_2 (MaxPooling2D)	(None, 6, 6, 128)	0
dropout_2 (Dropout)	(None, 6, 6, 128)	0
flatten (Flatten)	(None, 4608)	0
dense (Dense)	(None, 512)	2,359,808
activation_6 (Activation)	(None, 512)	0
dropout_3 (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 10)	5,130

Total params: 2,679,626 (10.22 MB)

Trainable params: 2,679,626 (10.22 MB)

Non-trainable params: 0 (0.00 B)

```
In [7]: STEP_SIZE_TRAIN=train_generator.n//train_generator.batch_size
# Measure the execution time
start_time = time.time()

model.fit(train_generator,validation_data=validation_gen,epochs=200)

end_time = time.time()
elapsed_time = end_time - start_time
```

Epoch 1/200
25/25 ————— 5s 157ms/step - accuracy: 0.0890 - loss: 2.3322 - val_accuracy: 0.1000 - val_loss: 2.3011

Epoch 2/200
25/25 ————— 4s 154ms/step - accuracy: 0.1340 - loss: 2.3001 - val_accuracy: 0.1550 - val_loss: 2.2372

Epoch 3/200
25/25 ————— 4s 162ms/step - accuracy: 0.1880 - loss: 2.2027 - val_accuracy: 0.2400 - val_loss: 2.0769

Epoch 4/200
25/25 ————— 4s 160ms/step - accuracy: 0.2299 - loss: 2.1084 - val_accuracy: 0.2600 - val_loss: 2.0161

Epoch 5/200
25/25 ————— 4s 167ms/step - accuracy: 0.2161 - loss: 2.0444 - val_accuracy: 0.2450 - val_loss: 1.9046

Epoch 6/200
25/25 ————— 4s 160ms/step - accuracy: 0.2236 - loss: 2.0755 - val_accuracy: 0.2700 - val_loss: 1.9300

Epoch 7/200
25/25 ————— 4s 157ms/step - accuracy: 0.2998 - loss: 1.9565 - val_accuracy: 0.2900 - val_loss: 1.9200

Epoch 8/200
25/25 ————— 4s 163ms/step - accuracy: 0.2854 - loss: 1.9678 - val_accuracy: 0.2200 - val_loss: 2.0452

Epoch 9/200
25/25 ————— 4s 161ms/step - accuracy: 0.3138 - loss: 1.9735 - val_accuracy: 0.3500 - val_loss: 1.8268

Epoch 10/200
25/25 ————— 4s 166ms/step - accuracy: 0.3174 - loss: 1.8635 - val_accuracy: 0.3950 - val_loss: 1.7900

Epoch 11/200
25/25 ————— 4s 166ms/step - accuracy: 0.3454 - loss: 1.8097 - val_accuracy: 0.4200 - val_loss: 1.6308

Epoch 12/200
25/25 ————— 4s 167ms/step - accuracy: 0.3527 - loss: 1.7176 - val_accuracy: 0.4450 - val_loss: 1.5843

Epoch 13/200
25/25 ————— 4s 164ms/step - accuracy: 0.3828 - loss: 1.6564 - val_accuracy: 0.4500 - val_loss: 1.5476

Epoch 14/200
25/25 ————— 4s 167ms/step - accuracy: 0.3882 - loss: 1.6838 - val_accuracy: 0.4000 - val_loss: 1.6897

Epoch 15/200
25/25 ————— 4s 168ms/step - accuracy: 0.3887 - loss: 1.6812 - val_accuracy: 0.4350 - val_loss: 1.5513

Epoch 16/200
25/25 ————— 4s 169ms/step - accuracy: 0.4168 - loss: 1.6188 - val_accuracy: 0.3950 - val_loss: 1.5964

Epoch 17/200
25/25 ————— 4s 164ms/step - accuracy: 0.3895 - loss: 1.6571 - val_accuracy: 0.4650 - val_loss: 1.5162

Epoch 18/200
25/25 ————— 4s 170ms/step - accuracy: 0.4429 - loss: 1.5384 - val_accuracy: 0.5150 - val_loss: 1.4634

Epoch 19/200
25/25 ————— 4s 165ms/step - accuracy: 0.4582 - loss: 1.4640 - val_accuracy: 0.4400 - val_loss: 1.4517

Epoch 20/200
25/25 ————— 4s 168ms/step - accuracy: 0.4804 - loss: 1.4429 - val_accuracy: 0.4650 - val_loss: 1.4854

Epoch 21/200
25/25 ————— 4s 170ms/step - accuracy: 0.4517 - loss: 1.4612 - val_accuracy: 0.5150 - val_loss: 1.4165

Epoch 22/200
25/25 ————— 4s 170ms/step - accuracy: 0.4987 - loss: 1.3627 - val_accuracy: 0.4500 - val_loss: 1.4091

Epoch 23/200
25/25 ————— 4s 165ms/step - accuracy: 0.4607 - loss: 1.4274 - val_accuracy: 0.5600 - val_loss: 1.2584

Epoch 24/200
25/25 ————— 4s 163ms/step - accuracy: 0.5193 - loss: 1.3213 - val_accuracy: 0.5650 - val_loss: 1.2483

Epoch 25/200
25/25 ————— 4s 163ms/step - accuracy: 0.4835 - loss: 1.3475 - val_accuracy: 0.5500 - val_loss: 1.2676

Epoch 26/200
25/25 ————— 4s 165ms/step - accuracy: 0.5028 - loss: 1.3596 - val_accuracy: 0.5300 - val_loss: 1.2933

Epoch 27/200
25/25 ————— 4s 168ms/step - accuracy: 0.5224 - loss: 1.3174 - val_accuracy: 0.5300 - val_loss: 1.2774

Epoch 28/200
25/25 ————— 4s 168ms/step - accuracy: 0.5612 - loss: 1.2203 - val_accuracy: 0.5550 - val_loss: 1.2307

Epoch 29/200
25/25 ————— 4s 166ms/step - accuracy: 0.5548 - loss: 1.1674 - val_accuracy: 0.5950 - val_loss: 1.2334

Epoch 30/200
25/25 ————— 4s 167ms/step - accuracy: 0.5841 - loss: 1.1709 - val_accuracy: 0.5500 - val_loss: 1.1717

Epoch 31/200
25/25 ————— 4s 165ms/step - accuracy: 0.6043 - loss: 1.0935 - val_accuracy: 0.5400 - val_loss: 1.2270

Epoch 32/200
25/25 ————— 4s 169ms/step - accuracy: 0.5900 - loss: 1.1565 - val_accuracy: 0.6050 - val_loss: 1.1707

Epoch 33/200
25/25 ————— 4s 166ms/step - accuracy: 0.6057 - loss: 1.0924 - val_accuracy: 0.5700 - val_loss: 1.1845

Epoch 34/200
25/25 ————— 4s 166ms/step - accuracy: 0.6117 - loss: 1.0508 - val_accuracy: 0.5650 - val_loss: 1.2611

Epoch 35/200
25/25 ————— 4s 162ms/step - accuracy: 0.6288 - loss: 1.0134 - val_accuracy: 0.6050 - val_loss: 1.1702

Epoch 36/200
25/25 ————— 4s 167ms/step - accuracy: 0.6574 - loss: 1.0158 - val_accuracy: 0.5050 - val_loss: 1.3790

Epoch 37/200
25/25 ————— 4s 173ms/step - accuracy: 0.6466 - loss: 0.9846 - val_accuracy: 0.4950 - val_loss: 1.3185

Epoch 38/200
25/25 ————— 4s 168ms/step - accuracy: 0.6530 - loss: 0.9448 - val_accuracy: 0.6300 - val_loss: 1.1059

Epoch 39/200
25/25 ————— 4s 169ms/step - accuracy: 0.6636 - loss: 0.9406 - val_accuracy: 0.5450 - val_loss: 1.3051

Epoch 40/200
25/25 ————— 4s 171ms/step - accuracy: 0.7135 - loss: 0.7819 - val_accuracy: 0.5100 - val_loss: 1.4909

Epoch 41/200
25/25 ————— 4s 171ms/step - accuracy: 0.6594 - loss: 0.9155 - val_accuracy: 0.6400 - val_loss: 1.0680

Epoch 42/200
25/25 ————— 4s 165ms/step - accuracy: 0.7064 - loss: 0.8548 - val_accuracy: 0.6250 - val_loss: 1.1288

Epoch 43/200
25/25 ————— 4s 166ms/step - accuracy: 0.7434 - loss: 0.7080 - val_accuracy: 0.6400 - val_loss: 1.0650

Epoch 44/200
25/25 ————— 4s 168ms/step - accuracy: 0.7365 - loss: 0.7392 - val_accuracy: 0.6400 - val_loss: 1.2224

Epoch 45/200
25/25 ————— 4s 165ms/step - accuracy: 0.7206 - loss: 0.7465 - val_accuracy: 0.6050 - val_loss: 1.1413

Epoch 46/200
25/25 ————— 4s 167ms/step - accuracy: 0.7372 - loss: 0.7719 - val_accuracy: 0.6600 - val_loss: 1.0741

Epoch 47/200
25/25 ————— 4s 166ms/step - accuracy: 0.7668 - loss: 0.6776 - val_accuracy: 0.6050 - val_loss: 1.2668

Epoch 48/200
25/25 ————— 4s 172ms/step - accuracy: 0.7406 - loss: 0.6342 - val_accuracy: 0.6450 - val_loss: 1.0855

Epoch 49/200
25/25 ————— 4s 167ms/step - accuracy: 0.7495 - loss: 0.6687 - val_accuracy: 0.6200 - val_loss: 1.1426

Epoch 50/200
25/25 ————— 4s 174ms/step - accuracy: 0.7787 - loss: 0.5902 - val_accuracy: 0.5900 - val_loss: 1.3110

Epoch 51/200
25/25 ————— 4s 162ms/step - accuracy: 0.7930 - loss: 0.6049 - val_accuracy: 0.6400 - val_loss: 1.2216

Epoch 52/200
25/25 ————— 4s 167ms/step - accuracy: 0.8118 - loss: 0.5545 - val_accuracy: 0.6900 - val_loss: 1.1773

Epoch 53/200
25/25 ————— 4s 162ms/step - accuracy: 0.7822 - loss: 0.5856 - val_accuracy: 0.6450 - val_loss: 1.1829

Epoch 54/200
25/25 ————— 4s 169ms/step - accuracy: 0.7902 - loss: 0.5876 - val_accuracy: 0.6500 - val_loss: 1.1793

Epoch 55/200
25/25 ————— 4s 172ms/step - accuracy: 0.7871 - loss: 0.5644 - val_accuracy: 0.5750 - val_loss: 1.3634

Epoch 56/200
25/25 ————— 4s 174ms/step - accuracy: 0.8249 - loss: 0.4743 - val_accuracy: 0.6400 - val_loss: 1.3440

Epoch 57/200
25/25 ————— 4s 167ms/step - accuracy: 0.7946 - loss: 0.5888 - val_accuracy: 0.6550 - val_loss: 1.2579

Epoch 58/200
25/25 ————— 4s 169ms/step - accuracy: 0.8377 - loss: 0.4342 - val_accuracy: 0.6600 - val_loss: 1.2937

Epoch 59/200
25/25 ————— 4s 167ms/step - accuracy: 0.8479 - loss: 0.4293 - val_accuracy: 0.6600 - val_loss: 1.2260

Epoch 60/200
25/25 ————— 4s 169ms/step - accuracy: 0.8441 - loss: 0.4288 - val_accuracy: 0.6800 - val_loss: 1.1514

Epoch 61/200
25/25 ————— 4s 172ms/step - accuracy: 0.8759 - loss: 0.3626 - val_accuracy: 0.6300 - val_loss: 1.5372

Epoch 62/200
25/25 ————— 4s 166ms/step - accuracy: 0.8100 - loss: 0.5374 - val_accuracy: 0.6400 - val_loss: 1.2935

Epoch 63/200
25/25 ————— 4s 170ms/step - accuracy: 0.8421 - loss: 0.4204 - val_accuracy: 0.6200 - val_loss: 1.5336

Epoch 64/200
25/25 ————— 4s 170ms/step - accuracy: 0.8930 - loss: 0.2955 - val_accuracy: 0.6700 - val_loss: 1.3578

Epoch 65/200
25/25 ————— 4s 167ms/step - accuracy: 0.8889 - loss: 0.3113 - val_accuracy: 0.6150 - val_loss: 1.3991

Epoch 66/200
25/25 ————— 4s 172ms/step - accuracy: 0.8777 - loss: 0.3702 - val_accuracy: 0.6750 - val_loss: 1.3092

Epoch 67/200
25/25 ————— 4s 170ms/step - accuracy: 0.8840 - loss: 0.3203 - val_accuracy: 0.6650 - val_loss: 1.3457

Epoch 68/200
25/25 ————— 4s 171ms/step - accuracy: 0.8963 - loss: 0.2672 - val_accuracy: 0.6700 - val_loss: 1.3466

Epoch 69/200
25/25 ————— 4s 174ms/step - accuracy: 0.8528 - loss: 0.4241 - val_accuracy: 0.6050 - val_loss: 1.6286

Epoch 70/200
25/25 ————— 4s 170ms/step - accuracy: 0.9103 - loss: 0.2695 - val_accuracy: 0.6650 - val_loss: 1.4309

Epoch 71/200
25/25 ————— 4s 170ms/step - accuracy: 0.9254 - loss: 0.2268 - val_accuracy: 0.6350 - val_loss: 1.4934

Epoch 72/200
25/25 ————— 4s 169ms/step - accuracy: 0.9097 - loss: 0.2662 - val_accuracy: 0.6550 - val_loss: 1.5303

Epoch 73/200
25/25 ————— 4s 162ms/step - accuracy: 0.8869 - loss: 0.2817 - val_accuracy: 0.6650 - val_loss: 1.4049

Epoch 74/200
25/25 ————— 4s 172ms/step - accuracy: 0.9100 - loss: 0.2291 - val_accuracy: 0.6550 - val_loss: 1.3215

Epoch 75/200
25/25 ————— 4s 169ms/step - accuracy: 0.9192 - loss: 0.2422 - val_accuracy: 0.6750 - val_loss: 1.2551

Epoch 76/200
25/25 ————— 4s 162ms/step - accuracy: 0.9105 - loss: 0.2636 - val_accuracy: 0.6300 - val_loss: 1.5186

Epoch 77/200
25/25 ————— 4s 166ms/step - accuracy: 0.9350 - loss: 0.1721 - val_accuracy: 0.5850 - val_loss: 1.7975

Epoch 78/200
25/25 ————— 4s 170ms/step - accuracy: 0.9183 - loss: 0.2491 - val_accuracy: 0.6500 - val_loss: 1.4510

Epoch 79/200
25/25 ————— 4s 159ms/step - accuracy: 0.9141 - loss: 0.2119 - val_accuracy: 0.6500 - val_loss: 1.5305

Epoch 80/200
25/25 ————— 4s 168ms/step - accuracy: 0.9163 - loss: 0.2393 - val_accuracy: 0.6850 - val_loss: 1.3357

Epoch 81/200
25/25 ————— 4s 159ms/step - accuracy: 0.9439 - loss: 0.1721 - val_accuracy: 0.6800 - val_loss: 1.5198

Epoch 82/200
25/25 ————— 4s 166ms/step - accuracy: 0.9051 - loss: 0.2558 - val_accuracy: 0.6450 - val_loss: 1.4418

Epoch 83/200
25/25 ————— 4s 172ms/step - accuracy: 0.9331 - loss: 0.1749 - val_accuracy: 0.6750 - val_loss: 1.4238

Epoch 84/200
25/25 ————— 4s 170ms/step - accuracy: 0.9438 - loss: 0.1506 - val_accuracy: 0.6850 - val_loss: 1.3142

Epoch 85/200
25/25 ————— 4s 171ms/step - accuracy: 0.9479 - loss: 0.1356 - val_accuracy: 0.6800 - val_loss: 1.5173

Epoch 86/200
25/25 ————— 4s 171ms/step - accuracy: 0.9248 - loss: 0.2362 - val_accuracy: 0.6600 - val_loss: 1.7096

Epoch 87/200
25/25 ————— 4s 172ms/step - accuracy: 0.9312 - loss: 0.1948 - val_accuracy: 0.6550 - val_loss: 1.7363

Epoch 88/200
25/25 ————— 4s 169ms/step - accuracy: 0.9397 - loss: 0.1930 - val_accuracy: 0.6350 - val_loss: 1.7357

Epoch 89/200
25/25 ————— 4s 171ms/step - accuracy: 0.9558 - loss: 0.1363 - val_accuracy: 0.6750 - val_loss: 1.4225

Epoch 90/200
25/25 ————— 4s 169ms/step - accuracy: 0.9226 - loss: 0.2255 - val_accuracy: 0.6750 - val_loss: 1.5259

Epoch 91/200
25/25 ————— 4s 173ms/step - accuracy: 0.9393 - loss: 0.1695 - val_accuracy: 0.6550 - val_loss: 1.8268

Epoch 92/200
25/25 ————— 4s 161ms/step - accuracy: 0.9457 - loss: 0.1703 - val_accuracy: 0.6550 - val_loss: 1.9608

Epoch 93/200
25/25 ————— 4s 167ms/step - accuracy: 0.9414 - loss: 0.1481 - val_accuracy: 0.6550 - val_loss: 1.5784

Epoch 94/200
25/25 ————— 4s 170ms/step - accuracy: 0.9607 - loss: 0.1032 - val_accuracy: 0.6550 - val_loss: 1.7475

Epoch 95/200
25/25 ————— 4s 165ms/step - accuracy: 0.9778 - loss: 0.1090 - val_accuracy: 0.6700 - val_loss: 1.4802

Epoch 96/200
25/25 ————— 4s 172ms/step - accuracy: 0.9754 - loss: 0.1131 - val_accuracy: 0.6850 - val_loss: 1.4780

Epoch 97/200
25/25 ————— 4s 171ms/step - accuracy: 0.9543 - loss: 0.1551 - val_accuracy: 0.4850 - val_loss: 3.1356

Epoch 98/200
25/25 ————— 4s 163ms/step - accuracy: 0.9005 - loss: 0.3302 - val_accuracy: 0.6600 - val_loss: 1.5196

Epoch 99/200
25/25 ————— 4s 164ms/step - accuracy: 0.9756 - loss: 0.0736 - val_accuracy: 0.6750 - val_loss: 1.6067

Epoch 100/200
25/25 ————— 4s 170ms/step - accuracy: 0.9593 - loss: 0.1224 - val_accuracy: 0.6150 - val_loss: 2.1983

Epoch 101/200
25/25 ————— 4s 173ms/step - accuracy: 0.9357 - loss: 0.2089 - val_accuracy: 0.6800 - val_loss: 1.7204

Epoch 102/200
25/25 ————— 4s 173ms/step - accuracy: 0.9600 - loss: 0.1106 - val_accuracy: 0.6600 - val_loss: 2.1862

Epoch 103/200
25/25 ————— 4s 173ms/step - accuracy: 0.9742 - loss: 0.1175 - val_accuracy: 0.6700 - val_loss: 1.7248

Epoch 104/200
25/25 ————— 4s 171ms/step - accuracy: 0.9673 - loss: 0.1144 - val_accuracy: 0.6800 - val_loss: 1.7310

Epoch 105/200
25/25 ————— 4s 170ms/step - accuracy: 0.9703 - loss: 0.0887 - val_accuracy: 0.6900 - val_loss: 1.7657

Epoch 106/200
25/25 ————— 4s 170ms/step - accuracy: 0.9544 - loss: 0.1503 - val_accuracy: 0.6500 - val_loss: 1.8447

Epoch 107/200
25/25 ————— 4s 171ms/step - accuracy: 0.9348 - loss: 0.1630 - val_accuracy: 0.6800 - val_loss: 1.7305

Epoch 108/200
25/25 ————— 4s 178ms/step - accuracy: 0.9695 - loss: 0.1021 - val_accuracy: 0.6550 - val_loss: 2.1379

Epoch 109/200
25/25 ————— 4s 161ms/step - accuracy: 0.9433 - loss: 0.1890 - val_accuracy: 0.6500 - val_loss: 1.6394

Epoch 110/200
25/25 ————— 4s 166ms/step - accuracy: 0.9777 - loss: 0.1011 - val_accuracy: 0.6550 - val_loss: 1.5713

Epoch 111/200
25/25 ————— 4s 171ms/step - accuracy: 0.9575 - loss: 0.1313 - val_accuracy: 0.6700 - val_loss: 1.6176

Epoch 112/200
25/25 ————— 4s 169ms/step - accuracy: 0.9760 - loss: 0.0796 - val_accuracy: 0.6400 - val_loss: 2.2218

Epoch 113/200
25/25 ————— 4s 155ms/step - accuracy: 0.9694 - loss: 0.0870 - val_accuracy: 0.6450 - val_loss: 2.2486

Epoch 114/200
25/25 ————— 4s 160ms/step - accuracy: 0.9743 - loss: 0.0628 - val_accuracy: 0.5700 - val_loss: 3.4683

Epoch 115/200
25/25 ————— 4s 169ms/step - accuracy: 0.9345 - loss: 0.2422 - val_accuracy: 0.6000 - val_loss: 2.5655

Epoch 116/200
25/25 ————— 4s 168ms/step - accuracy: 0.9648 - loss: 0.1994 - val_accuracy: 0.6300 - val_loss: 2.0663

Epoch 117/200
25/25 ————— 4s 174ms/step - accuracy: 0.9458 - loss: 0.1140 - val_accuracy: 0.6750 - val_loss: 1.9091

Epoch 118/200
25/25 ————— 4s 174ms/step - accuracy: 0.9762 - loss: 0.0969 - val_accuracy: 0.7050 - val_loss: 1.7098

Epoch 119/200
25/25 ————— 4s 169ms/step - accuracy: 0.9849 - loss: 0.0512 - val_accuracy: 0.6600 - val_loss: 2.0054

Epoch 120/200
25/25 ————— 4s 169ms/step - accuracy: 0.9551 - loss: 0.1360 - val_accuracy: 0.6650 - val_loss: 1.9079

Epoch 121/200
25/25 ————— 4s 170ms/step - accuracy: 0.9772 - loss: 0.0688 - val_accuracy: 0.6650 - val_loss: 1.8168

Epoch 122/200
25/25 ————— 4s 169ms/step - accuracy: 0.9620 - loss: 0.1115 - val_accuracy: 0.6750 - val_loss: 1.8947

Epoch 123/200
25/25 ————— 4s 179ms/step - accuracy: 0.9783 - loss: 0.0906 - val_accuracy: 0.6550 - val_loss: 1.8924

Epoch 124/200
25/25 ————— 4s 163ms/step - accuracy: 0.9669 - loss: 0.1133 - val_accuracy: 0.6750 - val_loss: 1.7190

Epoch 125/200
25/25 ————— 5s 180ms/step - accuracy: 0.9892 - loss: 0.0311 - val_accuracy: 0.6550 - val_loss: 2.1296

Epoch 126/200
25/25 ————— 4s 169ms/step - accuracy: 0.9659 - loss: 0.0754 - val_accuracy: 0.6700 - val_loss: 1.9735

Epoch 127/200
25/25 ————— 4s 177ms/step - accuracy: 0.9602 - loss: 0.1226 - val_accuracy: 0.6650 - val_loss: 2.0814

Epoch 128/200
25/25 ————— 4s 173ms/step - accuracy: 0.9858 - loss: 0.0539 - val_accuracy: 0.6800 - val_loss: 2.1406

Epoch 129/200
25/25 ————— 4s 169ms/step - accuracy: 0.9681 - loss: 0.1102 - val_accuracy: 0.6750 - val_loss: 1.9733

Epoch 130/200
25/25 ————— 4s 174ms/step - accuracy: 0.9774 - loss: 0.0680 - val_accuracy: 0.6700 - val_loss: 1.7611

Epoch 131/200
25/25 ————— 4s 174ms/step - accuracy: 0.9811 - loss: 0.0621 - val_accuracy: 0.6500 - val_loss: 1.8567

Epoch 132/200
25/25 ————— 4s 168ms/step - accuracy: 0.9886 - loss: 0.0347 - val_accuracy: 0.6050 - val_loss: 2.9189

Epoch 133/200
25/25 ————— 4s 171ms/step - accuracy: 0.9355 - loss: 0.3112 - val_accuracy: 0.6600 - val_loss: 1.9797

Epoch 134/200
25/25 ————— 4s 172ms/step - accuracy: 0.9788 - loss: 0.0517 - val_accuracy: 0.6500 - val_loss: 2.3874

Epoch 135/200
25/25 ————— 4s 173ms/step - accuracy: 0.9656 - loss: 0.0846 - val_accuracy: 0.6650 - val_loss: 2.0185

Epoch 136/200
25/25 ————— 4s 171ms/step - accuracy: 0.9800 - loss: 0.0580 - val_accuracy: 0.6900 - val_loss: 1.9009

Epoch 137/200
25/25 ————— 4s 159ms/step - accuracy: 0.9758 - loss: 0.0743 - val_accuracy: 0.6350 - val_loss: 2.5351

Epoch 138/200
25/25 ————— 4s 161ms/step - accuracy: 0.9412 - loss: 0.1937 - val_accuracy: 0.6750 - val_loss: 2.3021

Epoch 139/200
25/25 ————— 4s 172ms/step - accuracy: 0.9578 - loss: 0.0986 - val_accuracy: 0.6600 - val_loss: 2.4281

Epoch 140/200
25/25 ————— 4s 171ms/step - accuracy: 0.9918 - loss: 0.0322 - val_accuracy: 0.6400 - val_loss: 2.8026

Epoch 141/200
25/25 ————— 4s 167ms/step - accuracy: 0.9650 - loss: 0.0813 - val_accuracy: 0.6900 - val_loss: 2.1600

Epoch 142/200
25/25 ————— 4s 173ms/step - accuracy: 0.9815 - loss: 0.0500 - val_accuracy: 0.6800 - val_loss: 2.1603

Epoch 143/200
25/25 ————— 4s 173ms/step - accuracy: 0.9883 - loss: 0.0346 - val_accuracy: 0.6850 - val_loss: 2.0507

Epoch 144/200
25/25 ————— 4s 169ms/step - accuracy: 0.9832 - loss: 0.0488 - val_accuracy: 0.6600 - val_loss: 2.2622

Epoch 145/200
25/25 ————— 4s 171ms/step - accuracy: 0.9690 - loss: 0.1287 - val_accuracy: 0.6600 - val_loss: 2.0851

Epoch 146/200
25/25 ————— 4s 172ms/step - accuracy: 0.9703 - loss: 0.0971 - val_accuracy: 0.6600 - val_loss: 2.2161

Epoch 147/200
25/25 ————— 4s 172ms/step - accuracy: 0.9647 - loss: 0.1197 - val_accuracy: 0.6900 - val_loss: 1.9852

Epoch 148/200
25/25 ————— 4s 170ms/step - accuracy: 0.9848 - loss: 0.0470 - val_accuracy: 0.6800 - val_loss: 2.1950

Epoch 149/200
25/25 ————— 4s 168ms/step - accuracy: 0.9646 - loss: 0.1005 - val_accuracy: 0.6750 - val_loss: 2.2800

Epoch 150/200
25/25 ————— 4s 171ms/step - accuracy: 0.9792 - loss: 0.0724 - val_accuracy: 0.6750 - val_loss: 2.0512

Epoch 151/200
25/25 ————— 4s 170ms/step - accuracy: 0.9825 - loss: 0.0804 - val_accuracy: 0.6900 - val_loss: 2.4662

Epoch 152/200
25/25 ————— 4s 171ms/step - accuracy: 0.9782 - loss: 0.0690 - val_accuracy: 0.6450 - val_loss: 2.5868

Epoch 153/200
25/25 ————— 4s 171ms/step - accuracy: 0.9741 - loss: 0.0586 - val_accuracy: 0.6700 - val_loss: 2.2021

Epoch 154/200
25/25 ————— 4s 173ms/step - accuracy: 0.9772 - loss: 0.0831 - val_accuracy: 0.6850 - val_loss: 2.1732

Epoch 155/200
25/25 ————— 4s 174ms/step - accuracy: 0.9813 - loss: 0.0443 - val_accuracy: 0.6250 - val_loss: 2.8519

Epoch 156/200
25/25 ————— 4s 168ms/step - accuracy: 0.9704 - loss: 0.0755 - val_accuracy: 0.6450 - val_loss: 2.3332

Epoch 157/200
25/25 ————— 4s 173ms/step - accuracy: 0.9592 - loss: 0.1600 - val_accuracy: 0.6850 - val_loss: 2.0250

Epoch 158/200
25/25 ————— 4s 172ms/step - accuracy: 0.9815 - loss: 0.0503 - val_accuracy: 0.6800 - val_loss: 2.1698

Epoch 159/200
25/25 ————— 4s 169ms/step - accuracy: 0.9784 - loss: 0.0438 - val_accuracy: 0.7050 - val_loss: 2.1671

Epoch 160/200
25/25 ————— 4s 169ms/step - accuracy: 0.9854 - loss: 0.0685 - val_accuracy: 0.6750 - val_loss: 2.2213

Epoch 161/200
25/25 ————— 4s 175ms/step - accuracy: 0.9817 - loss: 0.0532 - val_accuracy: 0.6800 - val_loss: 2.1180

Epoch 162/200
25/25 ————— 5s 179ms/step - accuracy: 0.9793 - loss: 0.0800 - val_accuracy: 0.6650 - val_loss: 2.0562

Epoch 163/200
25/25 ————— 4s 168ms/step - accuracy: 0.9934 - loss: 0.0224 - val_accuracy: 0.6800 - val_loss: 2.3128

Epoch 164/200
25/25 ————— 4s 172ms/step - accuracy: 0.9781 - loss: 0.0955 - val_accuracy: 0.6850 - val_loss: 2.1782

```

Epoch 165/200
25/25 ————— 4s 161ms/step - accuracy: 0.9667 - loss: 0.1743 - val_accuracy: 0.6650 - val_loss: 2.3627
Epoch 166/200
25/25 ————— 4s 171ms/step - accuracy: 0.9924 - loss: 0.0248 - val_accuracy: 0.6650 - val_loss: 2.4867
Epoch 167/200
25/25 ————— 4s 171ms/step - accuracy: 0.9799 - loss: 0.0824 - val_accuracy: 0.6800 - val_loss: 2.2316
Epoch 168/200
25/25 ————— 4s 169ms/step - accuracy: 0.9718 - loss: 0.0875 - val_accuracy: 0.6750 - val_loss: 2.1804
Epoch 169/200
25/25 ————— 4s 174ms/step - accuracy: 0.9887 - loss: 0.0427 - val_accuracy: 0.6750 - val_loss: 2.0417
Epoch 170/200
25/25 ————— 4s 167ms/step - accuracy: 0.9892 - loss: 0.0395 - val_accuracy: 0.6750 - val_loss: 2.4242
Epoch 171/200
25/25 ————— 4s 168ms/step - accuracy: 0.9685 - loss: 0.1369 - val_accuracy: 0.6550 - val_loss: 2.5412
Epoch 172/200
25/25 ————— 4s 175ms/step - accuracy: 0.9850 - loss: 0.0492 - val_accuracy: 0.7000 - val_loss: 1.9487
Epoch 173/200
25/25 ————— 4s 170ms/step - accuracy: 0.9841 - loss: 0.0448 - val_accuracy: 0.6200 - val_loss: 2.5740
Epoch 174/200
25/25 ————— 4s 177ms/step - accuracy: 0.9827 - loss: 0.0601 - val_accuracy: 0.6850 - val_loss: 1.9880
Epoch 175/200
25/25 ————— 4s 172ms/step - accuracy: 0.9967 - loss: 0.0140 - val_accuracy: 0.6900 - val_loss: 2.1357
Epoch 176/200
25/25 ————— 4s 168ms/step - accuracy: 0.9829 - loss: 0.0802 - val_accuracy: 0.6800 - val_loss: 2.0864
Epoch 177/200
25/25 ————— 4s 167ms/step - accuracy: 0.9862 - loss: 0.0423 - val_accuracy: 0.6900 - val_loss: 2.1834
Epoch 178/200
25/25 ————— 4s 171ms/step - accuracy: 0.9574 - loss: 0.1224 - val_accuracy: 0.6650 - val_loss: 2.3492
Epoch 179/200
25/25 ————— 4s 172ms/step - accuracy: 0.9945 - loss: 0.0241 - val_accuracy: 0.6250 - val_loss: 3.3972
Epoch 180/200
25/25 ————— 4s 171ms/step - accuracy: 0.9831 - loss: 0.0915 - val_accuracy: 0.6700 - val_loss: 3.0905
Epoch 181/200
25/25 ————— 4s 162ms/step - accuracy: 0.9797 - loss: 0.0585 - val_accuracy: 0.6150 - val_loss: 3.0855
Epoch 182/200
25/25 ————— 4s 166ms/step - accuracy: 0.9878 - loss: 0.0432 - val_accuracy: 0.6750 - val_loss: 2.1840
Epoch 183/200
25/25 ————— 4s 173ms/step - accuracy: 0.9706 - loss: 0.1064 - val_accuracy: 0.6550 - val_loss: 2.7121
Epoch 184/200
25/25 ————— 4s 168ms/step - accuracy: 0.9745 - loss: 0.0734 - val_accuracy: 0.6900 - val_loss: 2.2102
Epoch 185/200
25/25 ————— 4s 172ms/step - accuracy: 0.9871 - loss: 0.0432 - val_accuracy: 0.6950 - val_loss: 2.0016
Epoch 186/200
25/25 ————— 4s 165ms/step - accuracy: 0.9841 - loss: 0.0458 - val_accuracy: 0.6650 - val_loss: 2.3818
Epoch 187/200
25/25 ————— 4s 171ms/step - accuracy: 0.9763 - loss: 0.0931 - val_accuracy: 0.6950 - val_loss: 2.1277
Epoch 188/200
25/25 ————— 4s 168ms/step - accuracy: 0.9905 - loss: 0.0339 - val_accuracy: 0.6950 - val_loss: 2.1814
Epoch 189/200
25/25 ————— 4s 172ms/step - accuracy: 0.9924 - loss: 0.0474 - val_accuracy: 0.6600 - val_loss: 2.8345
Epoch 190/200
25/25 ————— 4s 169ms/step - accuracy: 0.9727 - loss: 0.0834 - val_accuracy: 0.6600 - val_loss: 2.1733
Epoch 191/200
25/25 ————— 4s 171ms/step - accuracy: 0.9894 - loss: 0.0371 - val_accuracy: 0.6700 - val_loss: 2.3174
Epoch 192/200
25/25 ————— 4s 170ms/step - accuracy: 0.9936 - loss: 0.0223 - val_accuracy: 0.6850 - val_loss: 2.5772
Epoch 193/200
25/25 ————— 4s 160ms/step - accuracy: 0.9917 - loss: 0.0226 - val_accuracy: 0.6650 - val_loss: 2.3985
Epoch 194/200
25/25 ————— 4s 167ms/step - accuracy: 0.9880 - loss: 0.0382 - val_accuracy: 0.6750 - val_loss: 2.5574
Epoch 195/200
25/25 ————— 4s 170ms/step - accuracy: 0.9776 - loss: 0.1042 - val_accuracy: 0.6600 - val_loss: 2.7016
Epoch 196/200
25/25 ————— 4s 175ms/step - accuracy: 0.9847 - loss: 0.0479 - val_accuracy: 0.6950 - val_loss: 2.6320
Epoch 197/200
25/25 ————— 4s 170ms/step - accuracy: 0.9786 - loss: 0.0357 - val_accuracy: 0.6550 - val_loss: 2.3906
Epoch 198/200
25/25 ————— 4s 174ms/step - accuracy: 0.9901 - loss: 0.0448 - val_accuracy: 0.6600 - val_loss: 2.7241
Epoch 199/200
25/25 ————— 4s 167ms/step - accuracy: 0.9870 - loss: 0.0350 - val_accuracy: 0.6250 - val_loss: 2.9793
Epoch 200/200
25/25 ————— 4s 171ms/step - accuracy: 0.9906 - loss: 0.0226 - val_accuracy: 0.6500 - val_loss: 2.4609

```

```
In [8]: print(f"Execution time: {elapsed_time:.2f} seconds")
```

Execution time: 850.77 seconds

```
In [9]: def append_core_data(score_path, num_cores, elapsed_time):
        # Check if the file already exists
        file_exists = os.path.exists(score_path)

        # Open the file in append mode
        with open(score_path, mode='a', newline='') as file:
```

```
writer = csv.writer(file)

# If the file is new, write the header
if not file_exists:
    writer.writerow(["Number of Cores", "Elapsed Time"])

# Write the new data
writer.writerow([num_cores, elapsed_time])
```

```
In [10]: score_path = r"C:\Users\nikhi\OneDrive\Desktop\Final Project\DEEP LEARNING WITH HPSC\core_data.txt"
append_core_data(score_path, number_of_worker, elapsed_time)
```

Import Req Lib

```
In [1]: %matplotlib inline

import shutil
import random
import numpy as np
from warnings import filterwarnings
filterwarnings('ignore')

from tensorflow.keras import layers, regularizers, optimizers
from tensorflow.keras import models
from tensorflow.keras.models import Sequential, Model
from tensorflow.keras.layers import LeakyReLU, Dense, Activation, Flatten, Dropout, BatchNormalization, Conv2D, MaxPooling2D
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import tensorflow as tf

import os
import time
import csv

from matplotlib import figure
```

Define 7 worker

```
In [2]: # Set the number of threads
number_of_worker = 7
os.environ['OMP_NUM_THREADS'] = '7' # OpenMP threads for parallelism
os.environ['TF_NUM_INTEROP_THREADS'] = '7' # Threads for inter-operation parallelism
os.environ['TF_NUM_INTRAOP_THREADS'] = '7' # Threads for intra-operation parallelism

# Confirm TensorFlow is using the specified number of threads
tf.config.threading.set_inter_op_parallelism_threads(number_of_worker)
tf.config.threading.set_intra_op_parallelism_threads(number_of_worker)
```

Train Val data Split

```
In [3]: source_dir = r"C:\Users\nikhi\OneDrive\Desktop\Final Project\DATA\Convert_Audio_File_to_jpg_file"
target_dir = r'genres_train_val_split_data'
split_ratio = 0.8

def Train_Test_Split(source_dir, target_dir, split_ratio):
    # Define source and target directories
    train_dir = os.path.join(target_dir, 'train')
    val_dir = os.path.join(target_dir, 'val')

    # Create target directories if they don't exist
    os.makedirs(train_dir, exist_ok=True)
    os.makedirs(val_dir, exist_ok=True)

    # Get the list of class directories
    classes = [d for d in os.listdir(source_dir) if os.path.isdir(os.path.join(source_dir, d))]

    for class_name in classes:
        # Create class directories in train and val folders
        os.makedirs(os.path.join(train_dir, class_name), exist_ok=True)
        os.makedirs(os.path.join(val_dir, class_name), exist_ok=True)

        # Get list of images in the class directory
        class_dir = os.path.join(source_dir, class_name)
        images = [f for f in os.listdir(class_dir) if os.path.isfile(os.path.join(class_dir, f))]

        # Shuffle the images
        random.shuffle(images)

        # Compute the split point
        split_point = int(len(images) * split_ratio)

        # Split the images into training and validation sets
        train_images = images[:split_point]
        val_images = images[split_point:]
```

```

# Move the images to the respective directories
for img in train_images:
    shutil.copy(os.path.join(class_dir, img), os.path.join(train_dir, class_name, img))

for img in val_images:
    shutil.copy(os.path.join(class_dir, img), os.path.join(val_dir, class_name, img))

print("Data split completed successfully!")

```

In [4]: `Train_Test_Split(source_dir,target_dir,split_ratio)`

Data split completed successfully!

Load the Data

```

In [5]: WIDTH = 64
HEIGHT = 64
BATCH_SIZE = 32
TRAIN_DIR=r'genres_train_val_split_data/train'
val_dir = r'genres_train_val_split_data/val'

# data prep
train_datagen = ImageDataGenerator(
    rescale=1./255.,validation_split=0.25)

train_generator = train_datagen.flow_from_directory(
    TRAIN_DIR,
    target_size=(HEIGHT, WIDTH),
    batch_size=BATCH_SIZE,
    class_mode='categorical')

validation_gen = train_datagen.flow_from_directory(
    val_dir,target_size = (HEIGHT,WIDTH),
    batch_size = BATCH_SIZE,
    class_mode = 'categorical'
)

```

Found 800 images belonging to 10 classes.

Found 200 images belonging to 10 classes.

Model Architecture

```

In [6]: model = Sequential()
model.add(Conv2D(32, (3, 3), padding='same',
    input_shape=(64,64,3)))
model.add(Activation('relu'))
model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Conv2D(64, (3, 3), padding='same'))
model.add(Activation('relu'))
model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.5))
model.add(Conv2D(128, (3, 3), padding='same'))
model.add(Activation('relu'))
model.add(Conv2D(128, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.5))
model.add(Flatten())
model.add(Dense(512))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(10, activation='softmax'))
model.compile(optimizer=RMSprop(learning_rate=0.0005, decay=1e-6),loss="categorical_crossentropy",metrics=["accuracy"])
model.summary()

```

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 64, 64, 32)	896
activation (Activation)	(None, 64, 64, 32)	0
conv2d_1 (Conv2D)	(None, 62, 62, 64)	18,496
activation_1 (Activation)	(None, 62, 62, 64)	0
max_pooling2d (MaxPooling2D)	(None, 31, 31, 64)	0
dropout (Dropout)	(None, 31, 31, 64)	0
conv2d_2 (Conv2D)	(None, 31, 31, 64)	36,928
activation_2 (Activation)	(None, 31, 31, 64)	0
conv2d_3 (Conv2D)	(None, 29, 29, 64)	36,928
activation_3 (Activation)	(None, 29, 29, 64)	0
max_pooling2d_1 (MaxPooling2D)	(None, 14, 14, 64)	0
dropout_1 (Dropout)	(None, 14, 14, 64)	0
conv2d_4 (Conv2D)	(None, 14, 14, 128)	73,856
activation_4 (Activation)	(None, 14, 14, 128)	0
conv2d_5 (Conv2D)	(None, 12, 12, 128)	147,584
activation_5 (Activation)	(None, 12, 12, 128)	0
max_pooling2d_2 (MaxPooling2D)	(None, 6, 6, 128)	0
dropout_2 (Dropout)	(None, 6, 6, 128)	0
flatten (Flatten)	(None, 4608)	0
dense (Dense)	(None, 512)	2,359,808
activation_6 (Activation)	(None, 512)	0
dropout_3 (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 10)	5,130

Total params: 2,679,626 (10.22 MB)

Trainable params: 2,679,626 (10.22 MB)

Non-trainable params: 0 (0.00 B)

```
In [7]: STEP_SIZE_TRAIN=train_generator.n//train_generator.batch_size
# Measure the execution time
start_time = time.time()

model.fit(train_generator,validation_data=validation_gen,epochs=200)

end_time = time.time()
elapsed_time = end_time - start_time
```

Epoch 1/200
25/25 ————— 4s 123ms/step - accuracy: 0.0818 - loss: 2.3262 - val_accuracy: 0.1050 - val_loss: 2.2680
Epoch 2/200
25/25 ————— 4s 150ms/step - accuracy: 0.1573 - loss: 2.2889 - val_accuracy: 0.2100 - val_loss: 2.0738
Epoch 3/200
25/25 ————— 4s 149ms/step - accuracy: 0.1911 - loss: 2.1370 - val_accuracy: 0.2250 - val_loss: 2.1721
Epoch 4/200
25/25 ————— 4s 145ms/step - accuracy: 0.2274 - loss: 2.1387 - val_accuracy: 0.2350 - val_loss: 2.0582
Epoch 5/200
25/25 ————— 4s 149ms/step - accuracy: 0.2377 - loss: 2.0395 - val_accuracy: 0.3100 - val_loss: 1.9587
Epoch 6/200
25/25 ————— 4s 150ms/step - accuracy: 0.2534 - loss: 1.9749 - val_accuracy: 0.3600 - val_loss: 2.0144
Epoch 7/200
25/25 ————— 4s 153ms/step - accuracy: 0.2613 - loss: 1.9955 - val_accuracy: 0.3100 - val_loss: 1.8442
Epoch 8/200
25/25 ————— 4s 149ms/step - accuracy: 0.2949 - loss: 1.8974 - val_accuracy: 0.3950 - val_loss: 1.8752
Epoch 9/200
25/25 ————— 4s 152ms/step - accuracy: 0.3669 - loss: 1.8716 - val_accuracy: 0.4100 - val_loss: 1.8567
Epoch 10/200
25/25 ————— 4s 154ms/step - accuracy: 0.3370 - loss: 1.8430 - val_accuracy: 0.4150 - val_loss: 1.7267
Epoch 11/200
25/25 ————— 4s 148ms/step - accuracy: 0.3315 - loss: 1.8672 - val_accuracy: 0.3900 - val_loss: 1.6877
Epoch 12/200
25/25 ————— 4s 154ms/step - accuracy: 0.4074 - loss: 1.7262 - val_accuracy: 0.4200 - val_loss: 1.7082
Epoch 13/200
25/25 ————— 4s 155ms/step - accuracy: 0.3605 - loss: 1.7425 - val_accuracy: 0.4300 - val_loss: 1.6156
Epoch 14/200
25/25 ————— 4s 155ms/step - accuracy: 0.3964 - loss: 1.6583 - val_accuracy: 0.4500 - val_loss: 1.5646
Epoch 15/200
25/25 ————— 4s 150ms/step - accuracy: 0.3550 - loss: 1.7073 - val_accuracy: 0.4550 - val_loss: 1.5227
Epoch 16/200
25/25 ————— 4s 153ms/step - accuracy: 0.4600 - loss: 1.5362 - val_accuracy: 0.4300 - val_loss: 1.5923
Epoch 17/200
25/25 ————— 4s 151ms/step - accuracy: 0.4731 - loss: 1.5131 - val_accuracy: 0.4800 - val_loss: 1.4602
Epoch 18/200
25/25 ————— 4s 153ms/step - accuracy: 0.4707 - loss: 1.4784 - val_accuracy: 0.4700 - val_loss: 1.4989
Epoch 19/200
25/25 ————— 4s 147ms/step - accuracy: 0.5048 - loss: 1.4154 - val_accuracy: 0.4900 - val_loss: 1.4474
Epoch 20/200
25/25 ————— 4s 152ms/step - accuracy: 0.5050 - loss: 1.4192 - val_accuracy: 0.4300 - val_loss: 1.5239
Epoch 21/200
25/25 ————— 4s 154ms/step - accuracy: 0.4976 - loss: 1.4264 - val_accuracy: 0.5150 - val_loss: 1.3873
Epoch 22/200
25/25 ————— 4s 151ms/step - accuracy: 0.5021 - loss: 1.3814 - val_accuracy: 0.5700 - val_loss: 1.3150
Epoch 23/200
25/25 ————— 4s 152ms/step - accuracy: 0.5420 - loss: 1.3249 - val_accuracy: 0.5050 - val_loss: 1.3644
Epoch 24/200
25/25 ————— 4s 153ms/step - accuracy: 0.5389 - loss: 1.2833 - val_accuracy: 0.5650 - val_loss: 1.2784
Epoch 25/200
25/25 ————— 4s 150ms/step - accuracy: 0.5266 - loss: 1.2446 - val_accuracy: 0.5300 - val_loss: 1.3330
Epoch 26/200
25/25 ————— 4s 155ms/step - accuracy: 0.5608 - loss: 1.2418 - val_accuracy: 0.5250 - val_loss: 1.3601
Epoch 27/200
25/25 ————— 4s 154ms/step - accuracy: 0.6144 - loss: 1.1044 - val_accuracy: 0.5150 - val_loss: 1.3303
Epoch 28/200
25/25 ————— 4s 154ms/step - accuracy: 0.5713 - loss: 1.1832 - val_accuracy: 0.5700 - val_loss: 1.2238
Epoch 29/200
25/25 ————— 4s 152ms/step - accuracy: 0.5994 - loss: 1.0920 - val_accuracy: 0.5350 - val_loss: 1.3166
Epoch 30/200
25/25 ————— 4s 152ms/step - accuracy: 0.6277 - loss: 1.0387 - val_accuracy: 0.5650 - val_loss: 1.3302
Epoch 31/200
25/25 ————— 4s 150ms/step - accuracy: 0.6816 - loss: 0.9319 - val_accuracy: 0.5200 - val_loss: 1.3509
Epoch 32/200
25/25 ————— 4s 153ms/step - accuracy: 0.6372 - loss: 1.0041 - val_accuracy: 0.5250 - val_loss: 1.3733
Epoch 33/200
25/25 ————— 4s 157ms/step - accuracy: 0.6570 - loss: 0.9495 - val_accuracy: 0.6100 - val_loss: 1.1951
Epoch 34/200
25/25 ————— 4s 154ms/step - accuracy: 0.7004 - loss: 0.8949 - val_accuracy: 0.5900 - val_loss: 1.2032
Epoch 35/200
25/25 ————— 4s 153ms/step - accuracy: 0.7214 - loss: 0.7897 - val_accuracy: 0.5300 - val_loss: 1.3989
Epoch 36/200
25/25 ————— 4s 155ms/step - accuracy: 0.6723 - loss: 0.9051 - val_accuracy: 0.5700 - val_loss: 1.3947
Epoch 37/200
25/25 ————— 4s 157ms/step - accuracy: 0.7025 - loss: 0.8452 - val_accuracy: 0.5950 - val_loss: 1.4084
Epoch 38/200
25/25 ————— 4s 152ms/step - accuracy: 0.7058 - loss: 0.8214 - val_accuracy: 0.6000 - val_loss: 1.3120
Epoch 39/200
25/25 ————— 4s 154ms/step - accuracy: 0.7523 - loss: 0.7262 - val_accuracy: 0.6000 - val_loss: 1.3909
Epoch 40/200
25/25 ————— 4s 153ms/step - accuracy: 0.7442 - loss: 0.6782 - val_accuracy: 0.6300 - val_loss: 1.2302
Epoch 41/200
25/25 ————— 4s 154ms/step - accuracy: 0.7542 - loss: 0.6868 - val_accuracy: 0.5750 - val_loss: 1.3111

Epoch 42/200
25/25 ————— 4s 150ms/step - accuracy: 0.7372 - loss: 0.7429 - val_accuracy: 0.6000 - val_loss: 1.3485

Epoch 43/200
25/25 ————— 4s 151ms/step - accuracy: 0.7675 - loss: 0.6476 - val_accuracy: 0.5900 - val_loss: 1.5109

Epoch 44/200
25/25 ————— 4s 151ms/step - accuracy: 0.7961 - loss: 0.6001 - val_accuracy: 0.6050 - val_loss: 1.4317

Epoch 45/200
25/25 ————— 4s 154ms/step - accuracy: 0.8014 - loss: 0.5804 - val_accuracy: 0.6100 - val_loss: 1.2404

Epoch 46/200
25/25 ————— 4s 153ms/step - accuracy: 0.7928 - loss: 0.5734 - val_accuracy: 0.6250 - val_loss: 1.4807

Epoch 47/200
25/25 ————— 4s 152ms/step - accuracy: 0.8146 - loss: 0.5881 - val_accuracy: 0.6300 - val_loss: 1.3544

Epoch 48/200
25/25 ————— 4s 155ms/step - accuracy: 0.8335 - loss: 0.4861 - val_accuracy: 0.6150 - val_loss: 1.5126

Epoch 49/200
25/25 ————— 4s 151ms/step - accuracy: 0.8099 - loss: 0.5352 - val_accuracy: 0.5750 - val_loss: 1.4304

Epoch 50/200
25/25 ————— 4s 153ms/step - accuracy: 0.8612 - loss: 0.4229 - val_accuracy: 0.5750 - val_loss: 1.5456

Epoch 51/200
25/25 ————— 4s 150ms/step - accuracy: 0.8352 - loss: 0.5294 - val_accuracy: 0.5600 - val_loss: 1.8977

Epoch 52/200
25/25 ————— 4s 150ms/step - accuracy: 0.8287 - loss: 0.4984 - val_accuracy: 0.6050 - val_loss: 1.4981

Epoch 53/200
25/25 ————— 4s 156ms/step - accuracy: 0.8452 - loss: 0.4254 - val_accuracy: 0.6400 - val_loss: 1.5920

Epoch 54/200
25/25 ————— 4s 154ms/step - accuracy: 0.8350 - loss: 0.4264 - val_accuracy: 0.6150 - val_loss: 1.4441

Epoch 55/200
25/25 ————— 4s 157ms/step - accuracy: 0.9038 - loss: 0.3091 - val_accuracy: 0.6400 - val_loss: 1.4488

Epoch 56/200
25/25 ————— 4s 152ms/step - accuracy: 0.8716 - loss: 0.3435 - val_accuracy: 0.6200 - val_loss: 1.4544

Epoch 57/200
25/25 ————— 4s 156ms/step - accuracy: 0.8840 - loss: 0.3251 - val_accuracy: 0.6000 - val_loss: 1.4671

Epoch 58/200
25/25 ————— 4s 155ms/step - accuracy: 0.8853 - loss: 0.3201 - val_accuracy: 0.6200 - val_loss: 1.4904

Epoch 59/200
25/25 ————— 4s 154ms/step - accuracy: 0.8687 - loss: 0.3920 - val_accuracy: 0.5850 - val_loss: 1.6867

Epoch 60/200
25/25 ————— 4s 152ms/step - accuracy: 0.8751 - loss: 0.3240 - val_accuracy: 0.6350 - val_loss: 1.5783

Epoch 61/200
25/25 ————— 4s 150ms/step - accuracy: 0.8843 - loss: 0.3223 - val_accuracy: 0.6150 - val_loss: 1.7564

Epoch 62/200
25/25 ————— 4s 155ms/step - accuracy: 0.8901 - loss: 0.3179 - val_accuracy: 0.6050 - val_loss: 1.6935

Epoch 63/200
25/25 ————— 4s 156ms/step - accuracy: 0.9084 - loss: 0.2651 - val_accuracy: 0.6450 - val_loss: 1.5367

Epoch 64/200
25/25 ————— 4s 159ms/step - accuracy: 0.9011 - loss: 0.2681 - val_accuracy: 0.6550 - val_loss: 1.5078

Epoch 65/200
25/25 ————— 4s 153ms/step - accuracy: 0.9111 - loss: 0.2012 - val_accuracy: 0.6550 - val_loss: 1.7399

Epoch 66/200
25/25 ————— 4s 154ms/step - accuracy: 0.9056 - loss: 0.2551 - val_accuracy: 0.6450 - val_loss: 1.7057

Epoch 67/200
25/25 ————— 4s 153ms/step - accuracy: 0.9123 - loss: 0.2318 - val_accuracy: 0.6200 - val_loss: 1.7583

Epoch 68/200
25/25 ————— 4s 148ms/step - accuracy: 0.9229 - loss: 0.2184 - val_accuracy: 0.6200 - val_loss: 1.7297

Epoch 69/200
25/25 ————— 4s 151ms/step - accuracy: 0.9322 - loss: 0.1998 - val_accuracy: 0.6450 - val_loss: 1.6918

Epoch 70/200
25/25 ————— 4s 153ms/step - accuracy: 0.9346 - loss: 0.1579 - val_accuracy: 0.6300 - val_loss: 1.9413

Epoch 71/200
25/25 ————— 4s 149ms/step - accuracy: 0.9307 - loss: 0.1901 - val_accuracy: 0.6100 - val_loss: 1.6549

Epoch 72/200
25/25 ————— 4s 154ms/step - accuracy: 0.8986 - loss: 0.2894 - val_accuracy: 0.6350 - val_loss: 1.7343

Epoch 73/200
25/25 ————— 4s 156ms/step - accuracy: 0.9419 - loss: 0.1680 - val_accuracy: 0.6350 - val_loss: 1.8229

Epoch 74/200
25/25 ————— 4s 152ms/step - accuracy: 0.9522 - loss: 0.1810 - val_accuracy: 0.6350 - val_loss: 1.8504

Epoch 75/200
25/25 ————— 4s 152ms/step - accuracy: 0.9320 - loss: 0.2007 - val_accuracy: 0.5950 - val_loss: 1.8805

Epoch 76/200
25/25 ————— 4s 152ms/step - accuracy: 0.9319 - loss: 0.1726 - val_accuracy: 0.6050 - val_loss: 2.3080

Epoch 77/200
25/25 ————— 4s 152ms/step - accuracy: 0.9378 - loss: 0.1729 - val_accuracy: 0.5850 - val_loss: 2.5177

Epoch 78/200
25/25 ————— 4s 153ms/step - accuracy: 0.9300 - loss: 0.2347 - val_accuracy: 0.6300 - val_loss: 1.8858

Epoch 79/200
25/25 ————— 4s 150ms/step - accuracy: 0.9570 - loss: 0.1181 - val_accuracy: 0.6500 - val_loss: 1.7856

Epoch 80/200
25/25 ————— 4s 156ms/step - accuracy: 0.9564 - loss: 0.1307 - val_accuracy: 0.6600 - val_loss: 1.9016

Epoch 81/200
25/25 ————— 4s 153ms/step - accuracy: 0.9573 - loss: 0.1220 - val_accuracy: 0.6350 - val_loss: 2.2473

Epoch 82/200
25/25 ————— 4s 147ms/step - accuracy: 0.9390 - loss: 0.1833 - val_accuracy: 0.6300 - val_loss: 1.9236

Epoch 83/200
25/25 ————— 4s 148ms/step - accuracy: 0.9190 - loss: 0.2433 - val_accuracy: 0.6200 - val_loss: 1.9956

Epoch 84/200
25/25 ————— 4s 148ms/step - accuracy: 0.9646 - loss: 0.0943 - val_accuracy: 0.6400 - val_loss: 1.9079

Epoch 85/200
25/25 ————— 4s 151ms/step - accuracy: 0.9253 - loss: 0.2035 - val_accuracy: 0.6650 - val_loss: 1.9023

Epoch 86/200
25/25 ————— 4s 151ms/step - accuracy: 0.9570 - loss: 0.1210 - val_accuracy: 0.6300 - val_loss: 2.2414

Epoch 87/200
25/25 ————— 4s 155ms/step - accuracy: 0.9634 - loss: 0.1130 - val_accuracy: 0.6800 - val_loss: 1.7807

Epoch 88/200
25/25 ————— 4s 152ms/step - accuracy: 0.9449 - loss: 0.1721 - val_accuracy: 0.5950 - val_loss: 2.3276

Epoch 89/200
25/25 ————— 4s 158ms/step - accuracy: 0.9632 - loss: 0.1476 - val_accuracy: 0.6400 - val_loss: 1.9755

Epoch 90/200
25/25 ————— 4s 146ms/step - accuracy: 0.9655 - loss: 0.1029 - val_accuracy: 0.6450 - val_loss: 2.1367

Epoch 91/200
25/25 ————— 4s 147ms/step - accuracy: 0.9527 - loss: 0.1291 - val_accuracy: 0.6400 - val_loss: 2.1940

Epoch 92/200
25/25 ————— 4s 148ms/step - accuracy: 0.9660 - loss: 0.0966 - val_accuracy: 0.6100 - val_loss: 2.1825

Epoch 93/200
25/25 ————— 4s 148ms/step - accuracy: 0.9590 - loss: 0.1117 - val_accuracy: 0.6500 - val_loss: 2.4047

Epoch 94/200
25/25 ————— 4s 153ms/step - accuracy: 0.9492 - loss: 0.1480 - val_accuracy: 0.6200 - val_loss: 2.1937

Epoch 95/200
25/25 ————— 4s 153ms/step - accuracy: 0.9725 - loss: 0.0931 - val_accuracy: 0.5950 - val_loss: 2.4867

Epoch 96/200
25/25 ————— 4s 155ms/step - accuracy: 0.9345 - loss: 0.1980 - val_accuracy: 0.5950 - val_loss: 2.5017

Epoch 97/200
25/25 ————— 4s 156ms/step - accuracy: 0.9536 - loss: 0.1214 - val_accuracy: 0.6600 - val_loss: 2.3244

Epoch 98/200
25/25 ————— 4s 153ms/step - accuracy: 0.9634 - loss: 0.1116 - val_accuracy: 0.6050 - val_loss: 2.7636

Epoch 99/200
25/25 ————— 4s 151ms/step - accuracy: 0.9595 - loss: 0.1351 - val_accuracy: 0.6100 - val_loss: 2.1780

Epoch 100/200
25/25 ————— 4s 154ms/step - accuracy: 0.9683 - loss: 0.0903 - val_accuracy: 0.6450 - val_loss: 2.2073

Epoch 101/200
25/25 ————— 4s 150ms/step - accuracy: 0.9749 - loss: 0.0874 - val_accuracy: 0.6300 - val_loss: 2.6755

Epoch 102/200
25/25 ————— 4s 150ms/step - accuracy: 0.9796 - loss: 0.0895 - val_accuracy: 0.6300 - val_loss: 2.2189

Epoch 103/200
25/25 ————— 4s 149ms/step - accuracy: 0.9510 - loss: 0.1476 - val_accuracy: 0.5800 - val_loss: 2.4473

Epoch 104/200
25/25 ————— 4s 157ms/step - accuracy: 0.9651 - loss: 0.1142 - val_accuracy: 0.6400 - val_loss: 2.1473

Epoch 105/200
25/25 ————— 4s 151ms/step - accuracy: 0.9715 - loss: 0.1023 - val_accuracy: 0.5900 - val_loss: 2.3329

Epoch 106/200
25/25 ————— 4s 154ms/step - accuracy: 0.9530 - loss: 0.1820 - val_accuracy: 0.6000 - val_loss: 2.7490

Epoch 107/200
25/25 ————— 4s 152ms/step - accuracy: 0.9564 - loss: 0.1376 - val_accuracy: 0.6500 - val_loss: 2.3251

Epoch 108/200
25/25 ————— 4s 154ms/step - accuracy: 0.9596 - loss: 0.1519 - val_accuracy: 0.6550 - val_loss: 2.4093

Epoch 109/200
25/25 ————— 4s 152ms/step - accuracy: 0.9653 - loss: 0.0969 - val_accuracy: 0.5850 - val_loss: 2.8712

Epoch 110/200
25/25 ————— 4s 157ms/step - accuracy: 0.9487 - loss: 0.1415 - val_accuracy: 0.6350 - val_loss: 2.0791

Epoch 111/200
25/25 ————— 4s 153ms/step - accuracy: 0.9632 - loss: 0.1062 - val_accuracy: 0.6350 - val_loss: 2.1070

Epoch 112/200
25/25 ————— 4s 154ms/step - accuracy: 0.9727 - loss: 0.0791 - val_accuracy: 0.6250 - val_loss: 2.1753

Epoch 113/200
25/25 ————— 4s 150ms/step - accuracy: 0.9769 - loss: 0.0716 - val_accuracy: 0.6500 - val_loss: 2.0458

Epoch 114/200
25/25 ————— 4s 152ms/step - accuracy: 0.9788 - loss: 0.0582 - val_accuracy: 0.6400 - val_loss: 2.2886

Epoch 115/200
25/25 ————— 4s 156ms/step - accuracy: 0.9330 - loss: 0.2581 - val_accuracy: 0.6450 - val_loss: 2.3485

Epoch 116/200
25/25 ————— 4s 151ms/step - accuracy: 0.9665 - loss: 0.1004 - val_accuracy: 0.6450 - val_loss: 2.5679

Epoch 117/200
25/25 ————— 4s 150ms/step - accuracy: 0.9841 - loss: 0.0572 - val_accuracy: 0.6350 - val_loss: 2.4322

Epoch 118/200
25/25 ————— 4s 153ms/step - accuracy: 0.9653 - loss: 0.0872 - val_accuracy: 0.6350 - val_loss: 2.3860

Epoch 119/200
25/25 ————— 4s 152ms/step - accuracy: 0.9739 - loss: 0.0839 - val_accuracy: 0.6250 - val_loss: 2.6356

Epoch 120/200
25/25 ————— 4s 147ms/step - accuracy: 0.9751 - loss: 0.0728 - val_accuracy: 0.6400 - val_loss: 2.5431

Epoch 121/200
25/25 ————— 4s 157ms/step - accuracy: 0.9766 - loss: 0.0772 - val_accuracy: 0.6450 - val_loss: 2.2422

Epoch 122/200
25/25 ————— 4s 152ms/step - accuracy: 0.9807 - loss: 0.0660 - val_accuracy: 0.6250 - val_loss: 2.4464

Epoch 123/200
25/25 ————— 4s 156ms/step - accuracy: 0.9760 - loss: 0.0668 - val_accuracy: 0.6100 - val_loss: 2.8307

Epoch 124/200
25/25 ————— 4s 153ms/step - accuracy: 0.9725 - loss: 0.0801 - val_accuracy: 0.6250 - val_loss: 2.3358
Epoch 125/200
25/25 ————— 4s 155ms/step - accuracy: 0.9702 - loss: 0.1228 - val_accuracy: 0.6500 - val_loss: 2.3098
Epoch 126/200
25/25 ————— 4s 152ms/step - accuracy: 0.9603 - loss: 0.1196 - val_accuracy: 0.6550 - val_loss: 2.4535
Epoch 127/200
25/25 ————— 4s 152ms/step - accuracy: 0.9828 - loss: 0.0520 - val_accuracy: 0.6400 - val_loss: 2.5397
Epoch 128/200
25/25 ————— 4s 153ms/step - accuracy: 0.9824 - loss: 0.0516 - val_accuracy: 0.6250 - val_loss: 2.7237
Epoch 129/200
25/25 ————— 4s 155ms/step - accuracy: 0.9866 - loss: 0.0471 - val_accuracy: 0.6250 - val_loss: 2.4941
Epoch 130/200
25/25 ————— 4s 154ms/step - accuracy: 0.9766 - loss: 0.0721 - val_accuracy: 0.6350 - val_loss: 2.6602
Epoch 131/200
25/25 ————— 4s 153ms/step - accuracy: 0.9856 - loss: 0.0327 - val_accuracy: 0.6400 - val_loss: 3.0325
Epoch 132/200
25/25 ————— 4s 151ms/step - accuracy: 0.9794 - loss: 0.0530 - val_accuracy: 0.6200 - val_loss: 2.9336
Epoch 133/200
25/25 ————— 4s 153ms/step - accuracy: 0.9736 - loss: 0.0823 - val_accuracy: 0.6100 - val_loss: 2.8365
Epoch 134/200
25/25 ————— 4s 154ms/step - accuracy: 0.9675 - loss: 0.1192 - val_accuracy: 0.6150 - val_loss: 2.5089
Epoch 135/200
25/25 ————— 4s 156ms/step - accuracy: 0.9666 - loss: 0.1056 - val_accuracy: 0.6550 - val_loss: 2.5584
Epoch 136/200
25/25 ————— 4s 156ms/step - accuracy: 0.9863 - loss: 0.0400 - val_accuracy: 0.6100 - val_loss: 2.6453
Epoch 137/200
25/25 ————— 4s 157ms/step - accuracy: 0.9780 - loss: 0.0677 - val_accuracy: 0.6500 - val_loss: 2.3392
Epoch 138/200
25/25 ————— 4s 154ms/step - accuracy: 0.9820 - loss: 0.0461 - val_accuracy: 0.6400 - val_loss: 2.6747
Epoch 139/200
25/25 ————— 4s 151ms/step - accuracy: 0.9669 - loss: 0.1188 - val_accuracy: 0.6150 - val_loss: 2.7924
Epoch 140/200
25/25 ————— 4s 152ms/step - accuracy: 0.9767 - loss: 0.0923 - val_accuracy: 0.6300 - val_loss: 2.4425
Epoch 141/200
25/25 ————— 4s 152ms/step - accuracy: 0.9895 - loss: 0.0273 - val_accuracy: 0.6000 - val_loss: 2.8783
Epoch 142/200
25/25 ————— 4s 159ms/step - accuracy: 0.9818 - loss: 0.0588 - val_accuracy: 0.6200 - val_loss: 3.2892
Epoch 143/200
25/25 ————— 4s 152ms/step - accuracy: 0.9920 - loss: 0.0224 - val_accuracy: 0.6350 - val_loss: 2.8748
Epoch 144/200
25/25 ————— 4s 152ms/step - accuracy: 0.9902 - loss: 0.0372 - val_accuracy: 0.6450 - val_loss: 2.9861
Epoch 145/200
25/25 ————— 4s 149ms/step - accuracy: 0.9819 - loss: 0.0749 - val_accuracy: 0.6250 - val_loss: 2.6517
Epoch 146/200
25/25 ————— 4s 154ms/step - accuracy: 0.9738 - loss: 0.0858 - val_accuracy: 0.6450 - val_loss: 2.7020
Epoch 147/200
25/25 ————— 4s 155ms/step - accuracy: 0.9814 - loss: 0.0641 - val_accuracy: 0.6250 - val_loss: 2.7762
Epoch 148/200
25/25 ————— 4s 152ms/step - accuracy: 0.9870 - loss: 0.0443 - val_accuracy: 0.5850 - val_loss: 3.0334
Epoch 149/200
25/25 ————— 4s 156ms/step - accuracy: 0.9825 - loss: 0.0449 - val_accuracy: 0.6450 - val_loss: 2.9334
Epoch 150/200
25/25 ————— 4s 153ms/step - accuracy: 0.9744 - loss: 0.1010 - val_accuracy: 0.6250 - val_loss: 2.8921
Epoch 151/200
25/25 ————— 4s 152ms/step - accuracy: 0.9817 - loss: 0.0701 - val_accuracy: 0.6300 - val_loss: 2.8880
Epoch 152/200
25/25 ————— 4s 155ms/step - accuracy: 0.9804 - loss: 0.0619 - val_accuracy: 0.6450 - val_loss: 2.9605
Epoch 153/200
25/25 ————— 4s 153ms/step - accuracy: 0.9932 - loss: 0.0343 - val_accuracy: 0.6550 - val_loss: 2.7758
Epoch 154/200
25/25 ————— 4s 148ms/step - accuracy: 0.9638 - loss: 0.1059 - val_accuracy: 0.6250 - val_loss: 3.0840
Epoch 155/200
25/25 ————— 4s 152ms/step - accuracy: 0.9836 - loss: 0.0825 - val_accuracy: 0.5850 - val_loss: 3.1181
Epoch 156/200
25/25 ————— 4s 156ms/step - accuracy: 0.9827 - loss: 0.1365 - val_accuracy: 0.6250 - val_loss: 2.3661
Epoch 157/200
25/25 ————— 4s 156ms/step - accuracy: 0.9843 - loss: 0.0540 - val_accuracy: 0.6200 - val_loss: 3.2481
Epoch 158/200
25/25 ————— 4s 157ms/step - accuracy: 0.9880 - loss: 0.0368 - val_accuracy: 0.6100 - val_loss: 2.9156
Epoch 159/200
25/25 ————— 4s 156ms/step - accuracy: 0.9795 - loss: 0.0625 - val_accuracy: 0.6350 - val_loss: 2.9362
Epoch 160/200
25/25 ————— 4s 154ms/step - accuracy: 0.9871 - loss: 0.0501 - val_accuracy: 0.6100 - val_loss: 2.6436
Epoch 161/200
25/25 ————— 4s 155ms/step - accuracy: 0.9893 - loss: 0.0645 - val_accuracy: 0.6000 - val_loss: 3.0421
Epoch 162/200
25/25 ————— 4s 155ms/step - accuracy: 0.9654 - loss: 0.0766 - val_accuracy: 0.6750 - val_loss: 2.8712
Epoch 163/200
25/25 ————— 4s 161ms/step - accuracy: 0.9946 - loss: 0.0277 - val_accuracy: 0.6450 - val_loss: 3.0525
Epoch 164/200
25/25 ————— 4s 151ms/step - accuracy: 0.9867 - loss: 0.0390 - val_accuracy: 0.6050 - val_loss: 3.7059

```

Epoch 165/200
25/25 ----- 4s 151ms/step - accuracy: 0.9782 - loss: 0.0850 - val_accuracy: 0.6400 - val_loss: 3.2004
Epoch 166/200
25/25 ----- 4s 150ms/step - accuracy: 0.9659 - loss: 0.1274 - val_accuracy: 0.6400 - val_loss: 2.7282
Epoch 167/200
25/25 ----- 4s 156ms/step - accuracy: 0.9869 - loss: 0.0290 - val_accuracy: 0.6350 - val_loss: 2.9333
Epoch 168/200
25/25 ----- 4s 151ms/step - accuracy: 0.9765 - loss: 0.0736 - val_accuracy: 0.6200 - val_loss: 3.0869
Epoch 169/200
25/25 ----- 4s 153ms/step - accuracy: 0.9709 - loss: 0.0626 - val_accuracy: 0.6200 - val_loss: 3.1077
Epoch 170/200
25/25 ----- 4s 152ms/step - accuracy: 0.9798 - loss: 0.0398 - val_accuracy: 0.6150 - val_loss: 3.1138
Epoch 171/200
25/25 ----- 4s 152ms/step - accuracy: 0.9849 - loss: 0.0470 - val_accuracy: 0.6300 - val_loss: 3.1697
Epoch 172/200
25/25 ----- 4s 153ms/step - accuracy: 0.9905 - loss: 0.0274 - val_accuracy: 0.6200 - val_loss: 2.9240
Epoch 173/200
25/25 ----- 4s 151ms/step - accuracy: 0.9799 - loss: 0.0688 - val_accuracy: 0.6250 - val_loss: 3.0502
Epoch 174/200
25/25 ----- 4s 152ms/step - accuracy: 0.9785 - loss: 0.0580 - val_accuracy: 0.5950 - val_loss: 3.1900
Epoch 175/200
25/25 ----- 4s 153ms/step - accuracy: 0.9910 - loss: 0.0383 - val_accuracy: 0.6300 - val_loss: 3.1558
Epoch 176/200
25/25 ----- 4s 152ms/step - accuracy: 0.9687 - loss: 0.0718 - val_accuracy: 0.6350 - val_loss: 3.1910
Epoch 177/200
25/25 ----- 4s 153ms/step - accuracy: 0.9696 - loss: 0.0787 - val_accuracy: 0.6450 - val_loss: 2.9201
Epoch 178/200
25/25 ----- 4s 155ms/step - accuracy: 0.9831 - loss: 0.0540 - val_accuracy: 0.6400 - val_loss: 2.9415
Epoch 179/200
25/25 ----- 4s 152ms/step - accuracy: 0.9862 - loss: 0.0781 - val_accuracy: 0.6650 - val_loss: 3.0149
Epoch 180/200
25/25 ----- 4s 152ms/step - accuracy: 0.9834 - loss: 0.0509 - val_accuracy: 0.6450 - val_loss: 2.8482
Epoch 181/200
25/25 ----- 4s 152ms/step - accuracy: 0.9862 - loss: 0.0647 - val_accuracy: 0.6500 - val_loss: 3.4474
Epoch 182/200
25/25 ----- 4s 155ms/step - accuracy: 0.9794 - loss: 0.1101 - val_accuracy: 0.6500 - val_loss: 2.7705
Epoch 183/200
25/25 ----- 4s 152ms/step - accuracy: 0.9850 - loss: 0.0286 - val_accuracy: 0.6600 - val_loss: 3.5766
Epoch 184/200
25/25 ----- 4s 156ms/step - accuracy: 0.9922 - loss: 0.0405 - val_accuracy: 0.6650 - val_loss: 2.8988
Epoch 185/200
25/25 ----- 4s 151ms/step - accuracy: 0.9823 - loss: 0.0602 - val_accuracy: 0.6500 - val_loss: 2.8050
Epoch 186/200
25/25 ----- 4s 156ms/step - accuracy: 0.9909 - loss: 0.0293 - val_accuracy: 0.6450 - val_loss: 2.7298
Epoch 187/200
25/25 ----- 4s 154ms/step - accuracy: 0.9860 - loss: 0.0411 - val_accuracy: 0.6450 - val_loss: 3.0310
Epoch 188/200
25/25 ----- 4s 151ms/step - accuracy: 0.9940 - loss: 0.0242 - val_accuracy: 0.6600 - val_loss: 3.1644
Epoch 189/200
25/25 ----- 4s 151ms/step - accuracy: 0.9837 - loss: 0.0473 - val_accuracy: 0.6550 - val_loss: 2.8926
Epoch 190/200
25/25 ----- 4s 156ms/step - accuracy: 0.9790 - loss: 0.1066 - val_accuracy: 0.6300 - val_loss: 2.9581
Epoch 191/200
25/25 ----- 4s 152ms/step - accuracy: 0.9876 - loss: 0.0275 - val_accuracy: 0.6350 - val_loss: 3.0488
Epoch 192/200
25/25 ----- 4s 153ms/step - accuracy: 0.9851 - loss: 0.0696 - val_accuracy: 0.6200 - val_loss: 3.1620
Epoch 193/200
25/25 ----- 4s 154ms/step - accuracy: 0.9921 - loss: 0.0154 - val_accuracy: 0.6500 - val_loss: 3.8708
Epoch 194/200
25/25 ----- 4s 153ms/step - accuracy: 0.9761 - loss: 0.1093 - val_accuracy: 0.6750 - val_loss: 2.9113
Epoch 195/200
25/25 ----- 4s 151ms/step - accuracy: 0.9773 - loss: 0.0973 - val_accuracy: 0.6550 - val_loss: 3.0089
Epoch 196/200
25/25 ----- 4s 158ms/step - accuracy: 0.9829 - loss: 0.0561 - val_accuracy: 0.6550 - val_loss: 2.9303
Epoch 197/200
25/25 ----- 4s 152ms/step - accuracy: 0.9827 - loss: 0.0508 - val_accuracy: 0.6300 - val_loss: 3.1155
Epoch 198/200
25/25 ----- 4s 155ms/step - accuracy: 0.9807 - loss: 0.0829 - val_accuracy: 0.6350 - val_loss: 2.8135
Epoch 199/200
25/25 ----- 4s 155ms/step - accuracy: 0.9841 - loss: 0.0454 - val_accuracy: 0.6450 - val_loss: 3.5844
Epoch 200/200
25/25 ----- 4s 148ms/step - accuracy: 0.9732 - loss: 0.0892 - val_accuracy: 0.6450 - val_loss: 2.8868

```

```
In [8]: print(f"Execution time: {elapsed_time:.2f} seconds")
```

Execution time: 770.62 seconds

```
In [9]: def append_core_data(score_path, num_cores, elapsed_time):
        # Check if the file already exists
        file_exists = os.path.exists(score_path)

        # Open the file in append mode
        with open(score_path, mode='a', newline='') as file:
```

```
writer = csv.writer(file)

# If the file is new, write the header
if not file_exists:
    writer.writerow(["Number of Cores", "Elapsed Time"])

# Write the new data
writer.writerow([num_cores, elapsed_time])
```

```
In [10]: score_path = r"C:\Users\nikhi\OneDrive\Desktop\Final Project\DEEP LEARNING WITH HPSC\core_data.txt"
append_core_data(score_path, number_of_worker, elapsed_time)
```

Import Req Lib

```
In [1]: %matplotlib inline

import shutil
import random
import numpy as np
from warnings import filterwarnings
filterwarnings('ignore')

from tensorflow.keras import layers, regularizers, optimizers
from tensorflow.keras import models
from tensorflow.keras.models import Sequential, Model
from tensorflow.keras.layers import LeakyReLU, Dense, Activation, Flatten, Dropout, BatchNormalization, Conv2D, MaxPooling2D
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import tensorflow as tf

import os
import time
import csv

from matplotlib import figure
```

Define 8 worker

```
In [2]: # Set the number of threads
number_of_worker = 8
os.environ['OMP_NUM_THREADS'] = '8' # OpenMP threads for parallelism
os.environ['TF_NUM_INTEROP_THREADS'] = '8' # Threads for inter-operation parallelism
os.environ['TF_NUM_INTRAOP_THREADS'] = '8' # Threads for intra-operation parallelism

# Confirm TensorFlow is using the specified number of threads
tf.config.threading.set_inter_op_parallelism_threads(number_of_worker)
tf.config.threading.set_intra_op_parallelism_threads(number_of_worker)
```

Train Val data Split

```
In [3]: source_dir = r"C:\Users\nikhi\OneDrive\Desktop\Final Project\DATA\Convert_Audio_File_to_jpg_file"
target_dir = r'genres_train_val_split_data'
split_ratio = 0.8

def Train_Test_Split(source_dir, target_dir, split_ratio):
    # Define source and target directories
    train_dir = os.path.join(target_dir, 'train')
    val_dir = os.path.join(target_dir, 'val')

    # Create target directories if they don't exist
    os.makedirs(train_dir, exist_ok=True)
    os.makedirs(val_dir, exist_ok=True)

    # Get the list of class directories
    classes = [d for d in os.listdir(source_dir) if os.path.isdir(os.path.join(source_dir, d))]

    for class_name in classes:
        # Create class directories in train and val folders
        os.makedirs(os.path.join(train_dir, class_name), exist_ok=True)
        os.makedirs(os.path.join(val_dir, class_name), exist_ok=True)

        # Get list of images in the class directory
        class_dir = os.path.join(source_dir, class_name)
        images = [f for f in os.listdir(class_dir) if os.path.isfile(os.path.join(class_dir, f))]

        # Shuffle the images
        random.shuffle(images)

        # Compute the split point
        split_point = int(len(images) * split_ratio)

        # Split the images into training and validation sets
        train_images = images[:split_point]
        val_images = images[split_point:]
```

```

# Move the images to the respective directories
for img in train_images:
    shutil.copy(os.path.join(class_dir, img), os.path.join(train_dir, class_name, img))

for img in val_images:
    shutil.copy(os.path.join(class_dir, img), os.path.join(val_dir, class_name, img))

print("Data split completed successfully!")

```

In [4]: Train_Test_Split(source_dir,target_dir,split_ratio)

Data split completed successfully!

Load the Data

```

In [5]: WIDTH = 64
HEIGHT = 64
BATCH_SIZE = 32
TRAIN_DIR=r'genres_train_val_split_data/train'
val_dir = r'genres_train_val_split_data/val'

# data prep
train_datagen = ImageDataGenerator(
    rescale=1./255.,validation_split=0.25)

train_generator = train_datagen.flow_from_directory(
    TRAIN_DIR,
    target_size=(HEIGHT, WIDTH),
    batch_size=BATCH_SIZE,
    class_mode='categorical')

validation_gen = train_datagen.flow_from_directory(
    val_dir,target_size = (HEIGHT,WIDTH),
    batch_size = BATCH_SIZE,
    class_mode = 'categorical'
)

```

Found 800 images belonging to 10 classes.

Found 200 images belonging to 10 classes.

Model Architecture

```

In [6]: model = Sequential()
model.add(Conv2D(32, (3, 3), padding='same',
    input_shape=(64,64,3)))
model.add(Activation('relu'))
model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Conv2D(64, (3, 3), padding='same'))
model.add(Activation('relu'))
model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.5))
model.add(Conv2D(128, (3, 3), padding='same'))
model.add(Activation('relu'))
model.add(Conv2D(128, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.5))
model.add(Flatten())
model.add(Dense(512))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(10, activation='softmax'))
model.compile(optimizer=RMSprop(learning_rate=0.0005, decay=1e-6),loss="categorical_crossentropy",metrics=["accuracy"])
model.summary()

```

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 64, 64, 32)	896
activation (Activation)	(None, 64, 64, 32)	0
conv2d_1 (Conv2D)	(None, 62, 62, 64)	18,496
activation_1 (Activation)	(None, 62, 62, 64)	0
max_pooling2d (MaxPooling2D)	(None, 31, 31, 64)	0
dropout (Dropout)	(None, 31, 31, 64)	0
conv2d_2 (Conv2D)	(None, 31, 31, 64)	36,928
activation_2 (Activation)	(None, 31, 31, 64)	0
conv2d_3 (Conv2D)	(None, 29, 29, 64)	36,928
activation_3 (Activation)	(None, 29, 29, 64)	0
max_pooling2d_1 (MaxPooling2D)	(None, 14, 14, 64)	0
dropout_1 (Dropout)	(None, 14, 14, 64)	0
conv2d_4 (Conv2D)	(None, 14, 14, 128)	73,856
activation_4 (Activation)	(None, 14, 14, 128)	0
conv2d_5 (Conv2D)	(None, 12, 12, 128)	147,584
activation_5 (Activation)	(None, 12, 12, 128)	0
max_pooling2d_2 (MaxPooling2D)	(None, 6, 6, 128)	0
dropout_2 (Dropout)	(None, 6, 6, 128)	0
flatten (Flatten)	(None, 4608)	0
dense (Dense)	(None, 512)	2,359,808
activation_6 (Activation)	(None, 512)	0
dropout_3 (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 10)	5,130

Total params: 2,679,626 (10.22 MB)

Trainable params: 2,679,626 (10.22 MB)

Non-trainable params: 0 (0.00 B)

```
In [7]: STEP_SIZE_TRAIN=train_generator.n//train_generator.batch_size
# Measure the execution time
start_time = time.time()

model.fit(train_generator,validation_data=validation_gen,epochs=200)

end_time = time.time()
elapsed_time = end_time - start_time
```


Epoch 1/200
25/25 ————— 4s 126ms/step - accuracy: 0.0908 - loss: 2.3262 - val_accuracy: 0.1000 - val_loss: 2.3015
Epoch 2/200
25/25 ————— 3s 135ms/step - accuracy: 0.1362 - loss: 2.2996 - val_accuracy: 0.1950 - val_loss: 2.2538
Epoch 3/200
25/25 ————— 3s 136ms/step - accuracy: 0.1901 - loss: 2.2619 - val_accuracy: 0.1550 - val_loss: 2.1788
Epoch 4/200
25/25 ————— 3s 136ms/step - accuracy: 0.2647 - loss: 2.0942 - val_accuracy: 0.1650 - val_loss: 2.1146
Epoch 5/200
25/25 ————— 3s 135ms/step - accuracy: 0.2221 - loss: 2.0821 - val_accuracy: 0.2550 - val_loss: 2.0688
Epoch 6/200
25/25 ————— 4s 140ms/step - accuracy: 0.2638 - loss: 1.9729 - val_accuracy: 0.1850 - val_loss: 2.1266
Epoch 7/200
25/25 ————— 4s 140ms/step - accuracy: 0.2670 - loss: 2.0135 - val_accuracy: 0.3050 - val_loss: 1.9701
Epoch 8/200
25/25 ————— 3s 137ms/step - accuracy: 0.3221 - loss: 1.8983 - val_accuracy: 0.3200 - val_loss: 1.9209
Epoch 9/200
25/25 ————— 4s 142ms/step - accuracy: 0.3465 - loss: 1.8112 - val_accuracy: 0.3850 - val_loss: 1.7874
Epoch 10/200
25/25 ————— 3s 134ms/step - accuracy: 0.3333 - loss: 1.7993 - val_accuracy: 0.3800 - val_loss: 1.7778
Epoch 11/200
25/25 ————— 4s 141ms/step - accuracy: 0.3197 - loss: 1.7694 - val_accuracy: 0.3800 - val_loss: 1.8889
Epoch 12/200
25/25 ————— 4s 139ms/step - accuracy: 0.4073 - loss: 1.7037 - val_accuracy: 0.3950 - val_loss: 1.7497
Epoch 13/200
25/25 ————— 4s 139ms/step - accuracy: 0.4157 - loss: 1.6517 - val_accuracy: 0.4050 - val_loss: 1.6553
Epoch 14/200
25/25 ————— 4s 139ms/step - accuracy: 0.4353 - loss: 1.5729 - val_accuracy: 0.4100 - val_loss: 1.5987
Epoch 15/200
25/25 ————— 4s 139ms/step - accuracy: 0.4026 - loss: 1.6035 - val_accuracy: 0.4650 - val_loss: 1.5758
Epoch 16/200
25/25 ————— 4s 142ms/step - accuracy: 0.4446 - loss: 1.5463 - val_accuracy: 0.4250 - val_loss: 1.5314
Epoch 17/200
25/25 ————— 4s 142ms/step - accuracy: 0.4380 - loss: 1.5158 - val_accuracy: 0.3150 - val_loss: 1.9509
Epoch 18/200
25/25 ————— 4s 140ms/step - accuracy: 0.4421 - loss: 1.5638 - val_accuracy: 0.5050 - val_loss: 1.5129
Epoch 19/200
25/25 ————— 3s 137ms/step - accuracy: 0.4866 - loss: 1.4481 - val_accuracy: 0.4600 - val_loss: 1.4984
Epoch 20/200
25/25 ————— 4s 141ms/step - accuracy: 0.5063 - loss: 1.3240 - val_accuracy: 0.5150 - val_loss: 1.4143
Epoch 21/200
25/25 ————— 4s 139ms/step - accuracy: 0.4910 - loss: 1.3623 - val_accuracy: 0.5200 - val_loss: 1.4117
Epoch 22/200
25/25 ————— 4s 144ms/step - accuracy: 0.5047 - loss: 1.3392 - val_accuracy: 0.4800 - val_loss: 1.5531
Epoch 23/200
25/25 ————— 3s 138ms/step - accuracy: 0.5424 - loss: 1.2446 - val_accuracy: 0.5400 - val_loss: 1.3136
Epoch 24/200
25/25 ————— 4s 140ms/step - accuracy: 0.5654 - loss: 1.2154 - val_accuracy: 0.4050 - val_loss: 1.7990
Epoch 25/200
25/25 ————— 4s 140ms/step - accuracy: 0.5409 - loss: 1.2645 - val_accuracy: 0.5100 - val_loss: 1.5017
Epoch 26/200
25/25 ————— 4s 139ms/step - accuracy: 0.6027 - loss: 1.1757 - val_accuracy: 0.5600 - val_loss: 1.3537
Epoch 27/200
25/25 ————— 4s 140ms/step - accuracy: 0.5567 - loss: 1.1760 - val_accuracy: 0.5450 - val_loss: 1.3311
Epoch 28/200
25/25 ————— 4s 140ms/step - accuracy: 0.6314 - loss: 1.0326 - val_accuracy: 0.4900 - val_loss: 1.6634
Epoch 29/200
25/25 ————— 3s 137ms/step - accuracy: 0.5486 - loss: 1.2524 - val_accuracy: 0.5700 - val_loss: 1.2929
Epoch 30/200
25/25 ————— 3s 138ms/step - accuracy: 0.6574 - loss: 1.0052 - val_accuracy: 0.5450 - val_loss: 1.3549
Epoch 31/200
25/25 ————— 3s 139ms/step - accuracy: 0.6195 - loss: 1.0483 - val_accuracy: 0.5400 - val_loss: 1.2411
Epoch 32/200
25/25 ————— 4s 142ms/step - accuracy: 0.6598 - loss: 0.9453 - val_accuracy: 0.5750 - val_loss: 1.2204
Epoch 33/200
25/25 ————— 4s 141ms/step - accuracy: 0.6279 - loss: 1.0213 - val_accuracy: 0.6000 - val_loss: 1.2158
Epoch 34/200
25/25 ————— 4s 138ms/step - accuracy: 0.6599 - loss: 0.9575 - val_accuracy: 0.6000 - val_loss: 1.2094
Epoch 35/200
25/25 ————— 4s 139ms/step - accuracy: 0.6886 - loss: 0.8812 - val_accuracy: 0.5850 - val_loss: 1.1563
Epoch 36/200
25/25 ————— 4s 139ms/step - accuracy: 0.6898 - loss: 0.8930 - val_accuracy: 0.5550 - val_loss: 1.2696
Epoch 37/200
25/25 ————— 3s 137ms/step - accuracy: 0.6576 - loss: 0.8930 - val_accuracy: 0.5500 - val_loss: 1.2612
Epoch 38/200
25/25 ————— 3s 139ms/step - accuracy: 0.6722 - loss: 0.9251 - val_accuracy: 0.5900 - val_loss: 1.2177
Epoch 39/200
25/25 ————— 4s 141ms/step - accuracy: 0.6881 - loss: 0.8160 - val_accuracy: 0.6000 - val_loss: 1.1883
Epoch 40/200
25/25 ————— 3s 138ms/step - accuracy: 0.7392 - loss: 0.7417 - val_accuracy: 0.5600 - val_loss: 1.1867
Epoch 41/200
25/25 ————— 3s 138ms/step - accuracy: 0.6914 - loss: 0.8287 - val_accuracy: 0.5600 - val_loss: 1.3192

Epoch 42/200
25/25 ————— 3s 137ms/step - accuracy: 0.7687 - loss: 0.6921 - val_accuracy: 0.5750 - val_loss: 1.3051

Epoch 43/200
25/25 ————— 4s 140ms/step - accuracy: 0.7411 - loss: 0.7306 - val_accuracy: 0.5950 - val_loss: 1.1532

Epoch 44/200
25/25 ————— 4s 141ms/step - accuracy: 0.7491 - loss: 0.6981 - val_accuracy: 0.5800 - val_loss: 1.4488

Epoch 45/200
25/25 ————— 4s 139ms/step - accuracy: 0.7577 - loss: 0.6257 - val_accuracy: 0.5050 - val_loss: 1.5136

Epoch 46/200
25/25 ————— 3s 137ms/step - accuracy: 0.7645 - loss: 0.6471 - val_accuracy: 0.5900 - val_loss: 1.2315

Epoch 47/200
25/25 ————— 4s 140ms/step - accuracy: 0.7872 - loss: 0.5983 - val_accuracy: 0.6150 - val_loss: 1.2239

Epoch 48/200
25/25 ————— 4s 142ms/step - accuracy: 0.7524 - loss: 0.7278 - val_accuracy: 0.6250 - val_loss: 1.1958

Epoch 49/200
25/25 ————— 4s 142ms/step - accuracy: 0.7851 - loss: 0.6215 - val_accuracy: 0.6300 - val_loss: 1.2098

Epoch 50/200
25/25 ————— 3s 138ms/step - accuracy: 0.8122 - loss: 0.5169 - val_accuracy: 0.6100 - val_loss: 1.3742

Epoch 51/200
25/25 ————— 4s 141ms/step - accuracy: 0.8208 - loss: 0.4952 - val_accuracy: 0.6300 - val_loss: 1.3932

Epoch 52/200
25/25 ————— 4s 139ms/step - accuracy: 0.8011 - loss: 0.5288 - val_accuracy: 0.5900 - val_loss: 1.3119

Epoch 53/200
25/25 ————— 4s 140ms/step - accuracy: 0.8117 - loss: 0.4985 - val_accuracy: 0.6000 - val_loss: 1.4959

Epoch 54/200
25/25 ————— 4s 139ms/step - accuracy: 0.8080 - loss: 0.4890 - val_accuracy: 0.6050 - val_loss: 1.4572

Epoch 55/200
25/25 ————— 4s 141ms/step - accuracy: 0.8633 - loss: 0.3825 - val_accuracy: 0.6000 - val_loss: 1.4302

Epoch 56/200
25/25 ————— 4s 141ms/step - accuracy: 0.8359 - loss: 0.4682 - val_accuracy: 0.5900 - val_loss: 1.4071

Epoch 57/200
25/25 ————— 4s 140ms/step - accuracy: 0.8595 - loss: 0.4078 - val_accuracy: 0.5450 - val_loss: 1.7839

Epoch 58/200
25/25 ————— 4s 142ms/step - accuracy: 0.8430 - loss: 0.3921 - val_accuracy: 0.5900 - val_loss: 1.5943

Epoch 59/200
25/25 ————— 4s 140ms/step - accuracy: 0.8386 - loss: 0.4653 - val_accuracy: 0.6600 - val_loss: 1.2770

Epoch 60/200
25/25 ————— 4s 141ms/step - accuracy: 0.8801 - loss: 0.3475 - val_accuracy: 0.5850 - val_loss: 1.5956

Epoch 61/200
25/25 ————— 4s 141ms/step - accuracy: 0.8934 - loss: 0.3076 - val_accuracy: 0.6100 - val_loss: 1.5560

Epoch 62/200
25/25 ————— 4s 141ms/step - accuracy: 0.8729 - loss: 0.2943 - val_accuracy: 0.5500 - val_loss: 2.1760

Epoch 63/200
25/25 ————— 3s 133ms/step - accuracy: 0.8659 - loss: 0.3484 - val_accuracy: 0.6050 - val_loss: 1.3787

Epoch 64/200
25/25 ————— 4s 139ms/step - accuracy: 0.8979 - loss: 0.3263 - val_accuracy: 0.6150 - val_loss: 1.2772

Epoch 65/200
25/25 ————— 4s 139ms/step - accuracy: 0.9240 - loss: 0.2520 - val_accuracy: 0.6150 - val_loss: 1.5082

Epoch 66/200
25/25 ————— 4s 143ms/step - accuracy: 0.9125 - loss: 0.2537 - val_accuracy: 0.6450 - val_loss: 1.6815

Epoch 67/200
25/25 ————— 4s 142ms/step - accuracy: 0.9034 - loss: 0.3102 - val_accuracy: 0.6150 - val_loss: 1.6648

Epoch 68/200
25/25 ————— 4s 142ms/step - accuracy: 0.8977 - loss: 0.2868 - val_accuracy: 0.6150 - val_loss: 1.5470

Epoch 69/200
25/25 ————— 4s 144ms/step - accuracy: 0.9120 - loss: 0.2509 - val_accuracy: 0.6050 - val_loss: 1.7457

Epoch 70/200
25/25 ————— 4s 141ms/step - accuracy: 0.8936 - loss: 0.2831 - val_accuracy: 0.5750 - val_loss: 2.1970

Epoch 71/200
25/25 ————— 4s 140ms/step - accuracy: 0.8854 - loss: 0.3099 - val_accuracy: 0.6350 - val_loss: 1.7903

Epoch 72/200
25/25 ————— 4s 139ms/step - accuracy: 0.9302 - loss: 0.2035 - val_accuracy: 0.6450 - val_loss: 1.5624

Epoch 73/200
25/25 ————— 3s 139ms/step - accuracy: 0.9376 - loss: 0.1596 - val_accuracy: 0.6400 - val_loss: 1.7233

Epoch 74/200
25/25 ————— 4s 140ms/step - accuracy: 0.9390 - loss: 0.1822 - val_accuracy: 0.6400 - val_loss: 1.7426

Epoch 75/200
25/25 ————— 3s 138ms/step - accuracy: 0.9161 - loss: 0.2388 - val_accuracy: 0.6150 - val_loss: 1.8065

Epoch 76/200
25/25 ————— 3s 138ms/step - accuracy: 0.9217 - loss: 0.2217 - val_accuracy: 0.5850 - val_loss: 2.0870

Epoch 77/200
25/25 ————— 4s 139ms/step - accuracy: 0.9474 - loss: 0.1783 - val_accuracy: 0.5600 - val_loss: 1.9619

Epoch 78/200
25/25 ————— 4s 140ms/step - accuracy: 0.9547 - loss: 0.1430 - val_accuracy: 0.6400 - val_loss: 1.7234

Epoch 79/200
25/25 ————— 4s 139ms/step - accuracy: 0.9492 - loss: 0.1798 - val_accuracy: 0.6450 - val_loss: 1.7415

Epoch 80/200
25/25 ————— 3s 139ms/step - accuracy: 0.9351 - loss: 0.1791 - val_accuracy: 0.6250 - val_loss: 2.0626

Epoch 81/200
25/25 ————— 4s 141ms/step - accuracy: 0.9433 - loss: 0.1763 - val_accuracy: 0.6250 - val_loss: 1.8965

Epoch 82/200
25/25 ————— 4s 140ms/step - accuracy: 0.9281 - loss: 0.2056 - val_accuracy: 0.6350 - val_loss: 1.8599

Epoch 83/200
25/25 ————— 4s 142ms/step - accuracy: 0.9496 - loss: 0.1491 - val_accuracy: 0.5900 - val_loss: 2.2457
Epoch 84/200
25/25 ————— 4s 142ms/step - accuracy: 0.9162 - loss: 0.2485 - val_accuracy: 0.5650 - val_loss: 2.3358
Epoch 85/200
25/25 ————— 4s 144ms/step - accuracy: 0.9598 - loss: 0.1338 - val_accuracy: 0.6200 - val_loss: 2.0118
Epoch 86/200
25/25 ————— 4s 141ms/step - accuracy: 0.9405 - loss: 0.1782 - val_accuracy: 0.6250 - val_loss: 1.8883
Epoch 87/200
25/25 ————— 4s 139ms/step - accuracy: 0.9804 - loss: 0.0627 - val_accuracy: 0.6350 - val_loss: 2.0973
Epoch 88/200
25/25 ————— 4s 141ms/step - accuracy: 0.9515 - loss: 0.1667 - val_accuracy: 0.6250 - val_loss: 2.0799
Epoch 89/200
25/25 ————— 3s 138ms/step - accuracy: 0.9499 - loss: 0.1645 - val_accuracy: 0.6100 - val_loss: 2.1547
Epoch 90/200
25/25 ————— 4s 140ms/step - accuracy: 0.9439 - loss: 0.1523 - val_accuracy: 0.6250 - val_loss: 2.1661
Epoch 91/200
25/25 ————— 3s 138ms/step - accuracy: 0.9245 - loss: 0.1971 - val_accuracy: 0.5800 - val_loss: 2.1904
Epoch 92/200
25/25 ————— 4s 141ms/step - accuracy: 0.9467 - loss: 0.1535 - val_accuracy: 0.6200 - val_loss: 2.0070
Epoch 93/200
25/25 ————— 4s 141ms/step - accuracy: 0.9467 - loss: 0.2095 - val_accuracy: 0.6500 - val_loss: 1.8759
Epoch 94/200
25/25 ————— 3s 135ms/step - accuracy: 0.9725 - loss: 0.0735 - val_accuracy: 0.6050 - val_loss: 2.1894
Epoch 95/200
25/25 ————— 3s 139ms/step - accuracy: 0.9641 - loss: 0.1138 - val_accuracy: 0.6100 - val_loss: 2.1479
Epoch 96/200
25/25 ————— 4s 139ms/step - accuracy: 0.9625 - loss: 0.1299 - val_accuracy: 0.5650 - val_loss: 1.9899
Epoch 97/200
25/25 ————— 4s 142ms/step - accuracy: 0.9669 - loss: 0.1340 - val_accuracy: 0.6200 - val_loss: 2.0378
Epoch 98/200
25/25 ————— 4s 140ms/step - accuracy: 0.9557 - loss: 0.1493 - val_accuracy: 0.5800 - val_loss: 2.0380
Epoch 99/200
25/25 ————— 3s 137ms/step - accuracy: 0.9268 - loss: 0.1841 - val_accuracy: 0.6200 - val_loss: 2.1253
Epoch 100/200
25/25 ————— 4s 141ms/step - accuracy: 0.9663 - loss: 0.1215 - val_accuracy: 0.6300 - val_loss: 2.0655
Epoch 101/200
25/25 ————— 4s 140ms/step - accuracy: 0.9568 - loss: 0.1693 - val_accuracy: 0.6250 - val_loss: 1.7176
Epoch 102/200
25/25 ————— 4s 142ms/step - accuracy: 0.9743 - loss: 0.0622 - val_accuracy: 0.5800 - val_loss: 2.7079
Epoch 103/200
25/25 ————— 4s 142ms/step - accuracy: 0.9500 - loss: 0.1408 - val_accuracy: 0.6400 - val_loss: 2.3660
Epoch 104/200
25/25 ————— 4s 140ms/step - accuracy: 0.9346 - loss: 0.1862 - val_accuracy: 0.6000 - val_loss: 2.5210
Epoch 105/200
25/25 ————— 4s 143ms/step - accuracy: 0.9648 - loss: 0.0955 - val_accuracy: 0.5850 - val_loss: 2.5043
Epoch 106/200
25/25 ————— 4s 142ms/step - accuracy: 0.9383 - loss: 0.2160 - val_accuracy: 0.6200 - val_loss: 2.3773
Epoch 107/200
25/25 ————— 3s 137ms/step - accuracy: 0.9772 - loss: 0.0925 - val_accuracy: 0.6300 - val_loss: 2.3062
Epoch 108/200
25/25 ————— 4s 140ms/step - accuracy: 0.9854 - loss: 0.0618 - val_accuracy: 0.5800 - val_loss: 2.2823
Epoch 109/200
25/25 ————— 3s 137ms/step - accuracy: 0.9648 - loss: 0.1287 - val_accuracy: 0.6500 - val_loss: 2.2313
Epoch 110/200
25/25 ————— 4s 139ms/step - accuracy: 0.9834 - loss: 0.0841 - val_accuracy: 0.6150 - val_loss: 2.8426
Epoch 111/200
25/25 ————— 4s 140ms/step - accuracy: 0.9567 - loss: 0.1696 - val_accuracy: 0.6000 - val_loss: 2.2645
Epoch 112/200
25/25 ————— 4s 143ms/step - accuracy: 0.9763 - loss: 0.0623 - val_accuracy: 0.5850 - val_loss: 3.1558
Epoch 113/200
25/25 ————— 4s 142ms/step - accuracy: 0.9779 - loss: 0.0871 - val_accuracy: 0.6000 - val_loss: 2.3316
Epoch 114/200
25/25 ————— 3s 139ms/step - accuracy: 0.9771 - loss: 0.0701 - val_accuracy: 0.6100 - val_loss: 2.5346
Epoch 115/200
25/25 ————— 4s 140ms/step - accuracy: 0.9910 - loss: 0.0450 - val_accuracy: 0.6200 - val_loss: 2.6165
Epoch 116/200
25/25 ————— 3s 133ms/step - accuracy: 0.9698 - loss: 0.0784 - val_accuracy: 0.6100 - val_loss: 2.5954
Epoch 117/200
25/25 ————— 4s 141ms/step - accuracy: 0.9836 - loss: 0.0577 - val_accuracy: 0.5850 - val_loss: 2.7277
Epoch 118/200
25/25 ————— 4s 139ms/step - accuracy: 0.9925 - loss: 0.0346 - val_accuracy: 0.6300 - val_loss: 2.8001
Epoch 119/200
25/25 ————— 3s 139ms/step - accuracy: 0.9619 - loss: 0.1764 - val_accuracy: 0.6100 - val_loss: 2.6465
Epoch 120/200
25/25 ————— 4s 141ms/step - accuracy: 0.9778 - loss: 0.0528 - val_accuracy: 0.5600 - val_loss: 3.3701
Epoch 121/200
25/25 ————— 4s 139ms/step - accuracy: 0.9516 - loss: 0.1404 - val_accuracy: 0.6050 - val_loss: 2.2734
Epoch 122/200
25/25 ————— 4s 144ms/step - accuracy: 0.9612 - loss: 0.1199 - val_accuracy: 0.5950 - val_loss: 3.1950
Epoch 123/200
25/25 ————— 3s 138ms/step - accuracy: 0.9744 - loss: 0.1086 - val_accuracy: 0.6000 - val_loss: 2.6465

Epoch 124/200
25/25 ————— 4s 143ms/step - accuracy: 0.9637 - loss: 0.0909 - val_accuracy: 0.6150 - val_loss: 2.5651

Epoch 125/200
25/25 ————— 4s 141ms/step - accuracy: 0.9712 - loss: 0.0857 - val_accuracy: 0.6150 - val_loss: 2.9187

Epoch 126/200
25/25 ————— 4s 139ms/step - accuracy: 0.9763 - loss: 0.0642 - val_accuracy: 0.6500 - val_loss: 2.5741

Epoch 127/200
25/25 ————— 3s 138ms/step - accuracy: 0.9774 - loss: 0.0712 - val_accuracy: 0.6100 - val_loss: 2.8007

Epoch 128/200
25/25 ————— 4s 143ms/step - accuracy: 0.9647 - loss: 0.1249 - val_accuracy: 0.6250 - val_loss: 2.3439

Epoch 129/200
25/25 ————— 4s 141ms/step - accuracy: 0.9923 - loss: 0.0272 - val_accuracy: 0.6050 - val_loss: 2.8320

Epoch 130/200
25/25 ————— 4s 142ms/step - accuracy: 0.9654 - loss: 0.0755 - val_accuracy: 0.6050 - val_loss: 2.5047

Epoch 131/200
25/25 ————— 3s 137ms/step - accuracy: 0.9896 - loss: 0.0388 - val_accuracy: 0.6000 - val_loss: 2.9230

Epoch 132/200
25/25 ————— 4s 139ms/step - accuracy: 0.9675 - loss: 0.1126 - val_accuracy: 0.5800 - val_loss: 2.9205

Epoch 133/200
25/25 ————— 3s 139ms/step - accuracy: 0.9697 - loss: 0.1045 - val_accuracy: 0.6300 - val_loss: 2.8700

Epoch 134/200
25/25 ————— 4s 141ms/step - accuracy: 0.9584 - loss: 0.1553 - val_accuracy: 0.6100 - val_loss: 3.2127

Epoch 135/200
25/25 ————— 4s 143ms/step - accuracy: 0.9559 - loss: 0.1463 - val_accuracy: 0.6400 - val_loss: 2.4648

Epoch 136/200
25/25 ————— 4s 143ms/step - accuracy: 0.9850 - loss: 0.0413 - val_accuracy: 0.6400 - val_loss: 2.4898

Epoch 137/200
25/25 ————— 4s 142ms/step - accuracy: 0.9781 - loss: 0.0685 - val_accuracy: 0.6100 - val_loss: 3.1533

Epoch 138/200
25/25 ————— 4s 140ms/step - accuracy: 0.9778 - loss: 0.0885 - val_accuracy: 0.6400 - val_loss: 2.9685

Epoch 139/200
25/25 ————— 3s 137ms/step - accuracy: 0.9756 - loss: 0.0563 - val_accuracy: 0.5950 - val_loss: 3.0030

Epoch 140/200
25/25 ————— 3s 140ms/step - accuracy: 0.9734 - loss: 0.1123 - val_accuracy: 0.6200 - val_loss: 2.6312

Epoch 141/200
25/25 ————— 4s 144ms/step - accuracy: 0.9894 - loss: 0.0447 - val_accuracy: 0.6000 - val_loss: 2.7892

Epoch 142/200
25/25 ————— 3s 136ms/step - accuracy: 0.9839 - loss: 0.0571 - val_accuracy: 0.6100 - val_loss: 3.5329

Epoch 143/200
25/25 ————— 4s 140ms/step - accuracy: 0.9584 - loss: 0.2007 - val_accuracy: 0.6000 - val_loss: 3.1208

Epoch 144/200
25/25 ————— 4s 140ms/step - accuracy: 0.9645 - loss: 0.0972 - val_accuracy: 0.6100 - val_loss: 3.0806

Epoch 145/200
25/25 ————— 4s 141ms/step - accuracy: 0.9682 - loss: 0.1397 - val_accuracy: 0.6200 - val_loss: 2.8241

Epoch 146/200
25/25 ————— 4s 141ms/step - accuracy: 0.9820 - loss: 0.0489 - val_accuracy: 0.6100 - val_loss: 2.8053

Epoch 147/200
25/25 ————— 4s 140ms/step - accuracy: 0.9878 - loss: 0.0593 - val_accuracy: 0.6150 - val_loss: 2.6795

Epoch 148/200
25/25 ————— 4s 145ms/step - accuracy: 0.9806 - loss: 0.0387 - val_accuracy: 0.6450 - val_loss: 3.2969

Epoch 149/200
25/25 ————— 4s 141ms/step - accuracy: 0.9720 - loss: 0.0867 - val_accuracy: 0.5750 - val_loss: 3.8260

Epoch 150/200
25/25 ————— 4s 144ms/step - accuracy: 0.9751 - loss: 0.0840 - val_accuracy: 0.6100 - val_loss: 3.1916

Epoch 151/200
25/25 ————— 4s 142ms/step - accuracy: 0.9577 - loss: 0.2086 - val_accuracy: 0.6250 - val_loss: 2.7172

Epoch 152/200
25/25 ————— 4s 141ms/step - accuracy: 0.9896 - loss: 0.0274 - val_accuracy: 0.6300 - val_loss: 2.7414

Epoch 153/200
25/25 ————— 4s 144ms/step - accuracy: 0.9864 - loss: 0.0404 - val_accuracy: 0.6200 - val_loss: 2.8900

Epoch 154/200
25/25 ————— 4s 140ms/step - accuracy: 0.9674 - loss: 0.1702 - val_accuracy: 0.6150 - val_loss: 2.7966

Epoch 155/200
25/25 ————— 3s 138ms/step - accuracy: 0.9900 - loss: 0.0298 - val_accuracy: 0.6150 - val_loss: 3.0291

Epoch 156/200
25/25 ————— 4s 141ms/step - accuracy: 0.9665 - loss: 0.1125 - val_accuracy: 0.6450 - val_loss: 2.6484

Epoch 157/200
25/25 ————— 3s 137ms/step - accuracy: 0.9833 - loss: 0.0632 - val_accuracy: 0.6200 - val_loss: 2.6650

Epoch 158/200
25/25 ————— 4s 140ms/step - accuracy: 0.9877 - loss: 0.0304 - val_accuracy: 0.5750 - val_loss: 3.2498

Epoch 159/200
25/25 ————— 4s 141ms/step - accuracy: 0.9653 - loss: 0.1523 - val_accuracy: 0.6350 - val_loss: 2.5158

Epoch 160/200
25/25 ————— 3s 138ms/step - accuracy: 0.9975 - loss: 0.0103 - val_accuracy: 0.6350 - val_loss: 2.9519

Epoch 161/200
25/25 ————— 4s 139ms/step - accuracy: 0.9862 - loss: 0.0531 - val_accuracy: 0.6300 - val_loss: 2.7204

Epoch 162/200
25/25 ————— 3s 138ms/step - accuracy: 0.9858 - loss: 0.0496 - val_accuracy: 0.6450 - val_loss: 3.1435

Epoch 163/200
25/25 ————— 4s 139ms/step - accuracy: 0.9840 - loss: 0.0454 - val_accuracy: 0.6350 - val_loss: 3.5224

Epoch 164/200
25/25 ————— 4s 139ms/step - accuracy: 0.9805 - loss: 0.0615 - val_accuracy: 0.6050 - val_loss: 3.2094

```

Epoch 165/200
25/25 ————— 4s 140ms/step - accuracy: 0.9871 - loss: 0.0618 - val_accuracy: 0.6350 - val_loss: 3.2888
Epoch 166/200
25/25 ————— 4s 140ms/step - accuracy: 0.9696 - loss: 0.1007 - val_accuracy: 0.6400 - val_loss: 2.6891
Epoch 167/200
25/25 ————— 4s 145ms/step - accuracy: 0.9872 - loss: 0.0435 - val_accuracy: 0.5900 - val_loss: 3.3853
Epoch 168/200
25/25 ————— 4s 139ms/step - accuracy: 0.9914 - loss: 0.0244 - val_accuracy: 0.6200 - val_loss: 3.1923
Epoch 169/200
25/25 ————— 4s 139ms/step - accuracy: 0.9791 - loss: 0.0804 - val_accuracy: 0.6300 - val_loss: 2.9425
Epoch 170/200
25/25 ————— 3s 138ms/step - accuracy: 0.9858 - loss: 0.0313 - val_accuracy: 0.6400 - val_loss: 3.1421
Epoch 171/200
25/25 ————— 4s 141ms/step - accuracy: 0.9780 - loss: 0.1068 - val_accuracy: 0.6050 - val_loss: 2.7188
Epoch 172/200
25/25 ————— 4s 145ms/step - accuracy: 0.9848 - loss: 0.0615 - val_accuracy: 0.6200 - val_loss: 2.4824
Epoch 173/200
25/25 ————— 4s 143ms/step - accuracy: 0.9863 - loss: 0.0414 - val_accuracy: 0.6400 - val_loss: 2.7057
Epoch 174/200
25/25 ————— 4s 141ms/step - accuracy: 0.9884 - loss: 0.0382 - val_accuracy: 0.6700 - val_loss: 2.7627
Epoch 175/200
25/25 ————— 4s 141ms/step - accuracy: 0.9889 - loss: 0.0458 - val_accuracy: 0.6450 - val_loss: 3.2791
Epoch 176/200
25/25 ————— 4s 142ms/step - accuracy: 0.9926 - loss: 0.0161 - val_accuracy: 0.6350 - val_loss: 3.3782
Epoch 177/200
25/25 ————— 4s 140ms/step - accuracy: 0.9737 - loss: 0.0997 - val_accuracy: 0.5950 - val_loss: 2.9673
Epoch 178/200
25/25 ————— 4s 140ms/step - accuracy: 0.9740 - loss: 0.0582 - val_accuracy: 0.6250 - val_loss: 2.7655
Epoch 179/200
25/25 ————— 4s 140ms/step - accuracy: 0.9769 - loss: 0.0929 - val_accuracy: 0.6100 - val_loss: 3.2484
Epoch 180/200
25/25 ————— 4s 140ms/step - accuracy: 0.9844 - loss: 0.0512 - val_accuracy: 0.6450 - val_loss: 3.1637
Epoch 181/200
25/25 ————— 4s 141ms/step - accuracy: 0.9837 - loss: 0.0645 - val_accuracy: 0.6100 - val_loss: 3.5123
Epoch 182/200
25/25 ————— 3s 138ms/step - accuracy: 0.9856 - loss: 0.0575 - val_accuracy: 0.6300 - val_loss: 3.5459
Epoch 183/200
25/25 ————— 4s 139ms/step - accuracy: 0.9792 - loss: 0.0598 - val_accuracy: 0.6250 - val_loss: 3.2964
Epoch 184/200
25/25 ————— 4s 140ms/step - accuracy: 0.9887 - loss: 0.0531 - val_accuracy: 0.6150 - val_loss: 3.4861
Epoch 185/200
25/25 ————— 4s 143ms/step - accuracy: 0.9784 - loss: 0.0492 - val_accuracy: 0.6350 - val_loss: 3.3608
Epoch 186/200
25/25 ————— 4s 141ms/step - accuracy: 0.9810 - loss: 0.0573 - val_accuracy: 0.6200 - val_loss: 3.4178
Epoch 187/200
25/25 ————— 4s 139ms/step - accuracy: 0.9845 - loss: 0.0397 - val_accuracy: 0.6050 - val_loss: 3.8684
Epoch 188/200
25/25 ————— 4s 141ms/step - accuracy: 0.9640 - loss: 0.1283 - val_accuracy: 0.6350 - val_loss: 3.2243
Epoch 189/200
25/25 ————— 4s 140ms/step - accuracy: 0.9854 - loss: 0.0466 - val_accuracy: 0.6200 - val_loss: 4.0521
Epoch 190/200
25/25 ————— 3s 136ms/step - accuracy: 0.9810 - loss: 0.0668 - val_accuracy: 0.6050 - val_loss: 3.4468
Epoch 191/200
25/25 ————— 4s 138ms/step - accuracy: 0.9773 - loss: 0.0749 - val_accuracy: 0.6200 - val_loss: 3.5472
Epoch 192/200
25/25 ————— 4s 145ms/step - accuracy: 0.9875 - loss: 0.0332 - val_accuracy: 0.5800 - val_loss: 4.8018
Epoch 193/200
25/25 ————— 4s 140ms/step - accuracy: 0.9606 - loss: 0.1226 - val_accuracy: 0.6250 - val_loss: 3.5109
Epoch 194/200
25/25 ————— 3s 139ms/step - accuracy: 0.9761 - loss: 0.0790 - val_accuracy: 0.6300 - val_loss: 4.0045
Epoch 195/200
25/25 ————— 4s 140ms/step - accuracy: 0.9658 - loss: 0.1862 - val_accuracy: 0.6150 - val_loss: 3.4841
Epoch 196/200
25/25 ————— 4s 141ms/step - accuracy: 0.9821 - loss: 0.0515 - val_accuracy: 0.5900 - val_loss: 3.7746
Epoch 197/200
25/25 ————— 4s 142ms/step - accuracy: 0.9773 - loss: 0.0986 - val_accuracy: 0.6100 - val_loss: 3.8439
Epoch 198/200
25/25 ————— 4s 141ms/step - accuracy: 0.9911 - loss: 0.0289 - val_accuracy: 0.6600 - val_loss: 3.3768
Epoch 199/200
25/25 ————— 4s 141ms/step - accuracy: 0.9782 - loss: 0.0690 - val_accuracy: 0.6250 - val_loss: 3.4474
Epoch 200/200
25/25 ————— 4s 143ms/step - accuracy: 0.9942 - loss: 0.0190 - val_accuracy: 0.6150 - val_loss: 4.5955

```

```
In [8]: print(f"Execution time: {elapsed_time:.2f} seconds")
```

Execution time: 707.66 seconds

```
In [9]: def append_core_data(score_path, num_cores, elapsed_time):
        # Check if the file already exists
        file_exists = os.path.exists(score_path)

        # Open the file in append mode
        with open(score_path, mode='a', newline='') as file:
```

```
writer = csv.writer(file)

# If the file is new, write the header
if not file_exists:
    writer.writerow(["Number of Cores", "Elapsed Time"])

# Write the new data
writer.writerow([num_cores, elapsed_time])
```

```
In [10]: score_path = r"C:\Users\nikhi\OneDrive\Desktop\Final Project\DEEP LEARNING WITH HPSC\core_data.txt"
append_core_data(score_path, number_of_worker, elapsed_time)
```