TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT
TP. HỒ CHÍ MINH
KHOA ĐIỆN – ĐIỆN TỬ
NGÀNH CÔNG NGHỆ KỸ THUẬT MÁY TÍNH

HCMUTE

BÁO CÁO
MÔN HỌC: CƠ SỞ VÀ ỨNG DỤNG AI
IMAGE CLASSIFICATION PROGRAM WITH DEEP LEARNING

MÃ SỐ LỚP HP: AIFA436864_23_2_09CLC
GVHD: HUỲNH THẾ THIỆN
SINH VIÊN THỰC HIỆN:
VÕ TẤN PHÁT – 21119115
HỌC KỲ: II – **NĂM HỌC:** 2023 – 2024
Tp. Thủ Đức, tháng 5, năm 2024

MỤC LỤC

1. DEEP LEARNING BACKGROUND 3

1.1 Tìm hiểu về Deep learning 3

1.2 Cơ chế hoạt động của Deep learning 3

1.3 Ưu và nhược điểm của Deep learning 4

1.3.1 Ưu điểm 4

1.3.2 Nhược điểm 4

2. RELEVANT ALGORITHMS 5

2.1 Mạng Noron tích chập 5

2.2 Adam Optimizer 5

3. NETWORK ARCHITECTURE 6

4. NETWORK ARCHITECTURE 11

5. CODE 11

6. SIMULATION RESULTS 13

7. DISCUSSION AND INSIGHTS 26

1. Deep learning background

Những năm gần đây, AI - Artificial Intelligence (Trí Tuệ Nhân Tạo), và cụ thể hơn là Machine Learning (Máy Học) nổi lên như một minh chứng của cuộc cách mạng công nghiệp lần thứ tư (1 - động cơ hơi nước, 2 - năng lượng điện, 3 - công nghệ thông tin). AI hiện diện trong mọi lĩnh vực của đời sống con người, từ kinh tế, giáo dục, y khoa cho đến những công việc nhà, giải trí hay thậm chí là trong quân sự. Những ứng dụng nổi bật trong việc phát triển AI đến từ nhiều lĩnh vực để giải quyết nhiều vấn đề khác nhau. Nhưng những đột phá phần nhiều đến từ Deep Learning (học sâu) - một mảng nhỏ đang mở rộng dần đến từng loại công việc, từ đơn giản đến phức tạp. Deep Learning đã giúp máy tính thực thi những việc tưởng chừng như không thể vào nhiều năm trước như: phân loại cả ngàn vật thể khác nhau trong các bức ảnh, tự tạo chú thích cho ảnh, bắt chước giọng nói và chữ viết của con người, giao tiếp với con người, hay thậm chí cả sáng tác văn, phim, ảnh, âm nhạc.

Chúng ta có thể thấy Deep learning chỉ là một nhánh nhỏ của Machine Learning. Tuy nhiên trong khoảng thời gian gần đây thì Deep Learning được nhắc đến rất nhiều như một xu hướng mới của cuộc cách mạng AI.

1.1 Tìm hiểu về Deep learning

Deep Learning (học sâu) có thể được xem là một lĩnh vực con của Machine Learning (học máy) – ở đó các máy tính sẽ học và cải thiện chính nó thông qua các thuật toán. Deep Learning được xây dựng dựa trên các khái niệm phức tạp hơn rất nhiều, chủ yếu hoạt động với các mạng nơ-ron nhân tạo để bắt chước khả năng tư duy và suy nghĩ của bộ não con người.

Khái niệm liên quan đến mạng nơ-ron nhân tạo và Deep Learning đã xuất hiện từ khoảng những năm 1960, tuy nhiên nó lại bị giới hạn bởi khả năng tính toán và số lượng dữ liệu lúc bấy giờ. Trong những năm gần đây, những tiến bộ trong phân tích dữ liệu lớn (Big Data) đã cho phép ta tận dụng được tối đa khả năng của mạng nơ-ron nhân tạo.

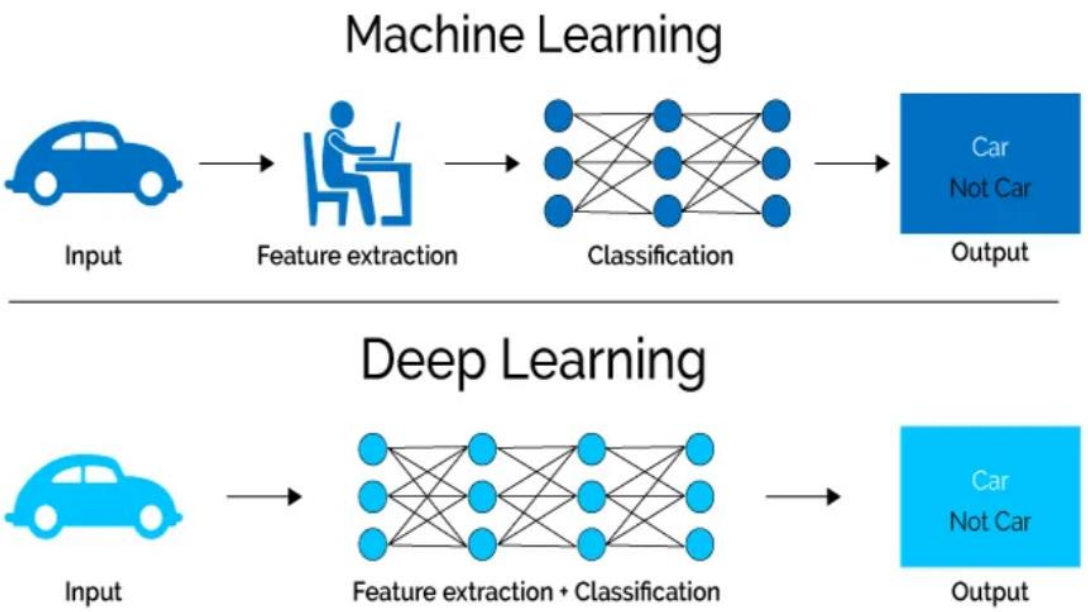
Mạng nơ-ron nhân tạo chính là động lực chính để phát triển Deep Learning. Các mạng nơ-ron sâu (DNN) bao gồm nhiều lớp nơ-ron khác nhau, có khả năng thực hiện các tính toán có độ phức tạp rất cao. Deep Learning hiện đang phát triển rất nhanh và được xem là một trong những bước đột phá lớn nhất trong Machine Learning.

1.2 Cơ chế hoạt động của Deep learning

Deep Learning là một phương pháp của Machine Learning. Mạng nơ-ron nhân tạo trong Deep Learning được xây dựng để mô phỏng khả năng tư duy của bộ não con người.

Một mạng nơ-ron bao gồm nhiều lớp (layer) khác nhau, số lượng layer càng nhiều thì mạng sẽ càng "sâu". Trong mỗi layer là các nút mạng (node) và được liên kết với những lớp liền kề khác. Mỗi kết nối giữa các node sẽ có một trọng số tương ứng, trọng số càng cao thì ảnh hưởng của kết nối này đến mạng nơ-ron càng lớn.

Mỗi nơ-ron sẽ có một hàm kích hoạt, về cơ bản thì có nhiệm vụ "chuẩn hoá" đầu ra từ nơ-ron này. Dữ liệu được người dùng đưa vào mạng nơ-ron sẽ đi qua tất cả layer và trả về kết quả ở layer cuối cùng, gọi là output layer.



Hình 1.1: Cơ chế hoạt động của Deep learning

Trong quá trình huấn luyện mô hình mạng nơ-ron, các trọng số sẽ được thay đổi và nhiệm vụ của mô hình là tìm ra bộ giá trị của trọng số sao cho phán đoán là tốt nhất.

Các hệ thống Deep Learning yêu cầu phần cứng phải rất mạnh để có thể xử lý được lượng dữ liệu lớn và thực hiện các phép tính phức tạp. Nhiều mô hình Deep Learning có thể mất nhiều tuần, thậm chí nhiều tháng để triển khai trên những phần cứng tiên tiến nhất hiện nay.

1.3 Ưu và nhược điểm của Deep learning

1.3.1 Ưu điểm

Deep Learning là một bước ngoặt to lớn trong lĩnh vực trí tuệ nhân tạo, cho phép khác nhà khoa học dữ liệu xây dựng nhiều mô hình có độ chính xác rất cao trong lĩnh vực nhận dạng ảnh, xử lý ngôn ngữ tự nhiên, xử lý giọng nói,… Một số ưu điểm vượt trội của Deep Learning gồm có:

- Kiến trúc mạng nơ-ron linh hoạt, có thể dễ dàng thay đổi để phù hợp với nhiều vấn đề khác nhau.
- Có khả năng giải quyết nhiều bài toán phức tạp với độ chính xác rất cao.
- Tính tự động hoá cao, có khả năng tự điều chỉnh và tự tối ưu.
- Có khả năng thực hiện tính toán song song, hiệu năng tốt, xử lý được lượng dữ liệu lớn.

1.3.2 Nhược điểm

Bên cạnh những ưu điểm, mặt khác, hiện nay Deep Learning vẫn còn nhiều khó khăn và hạn chế, chẳng hạn như:

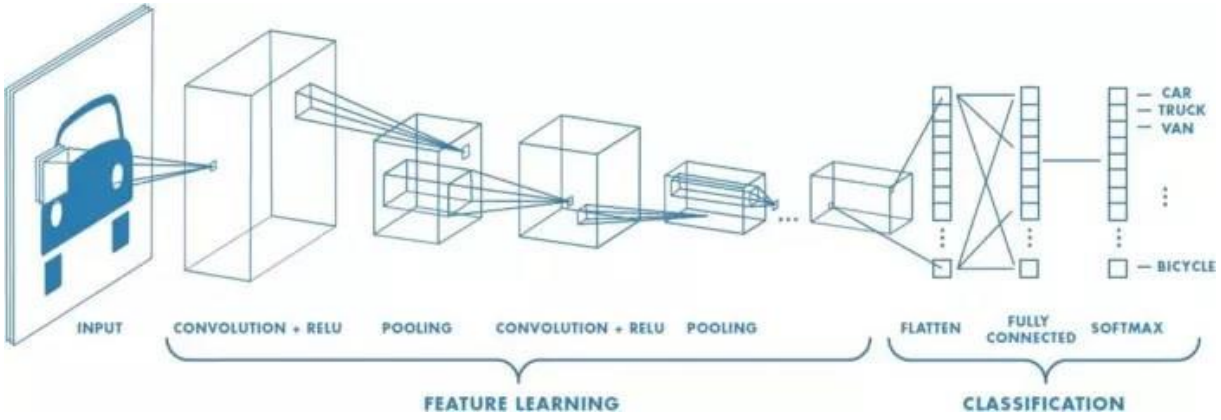
- Cần có khối lượng dữ liệu rất lớn để tận dụng tối đa khả năng của Deep Learning.
- Chi phí tính toán cao vì phải xử lý nhiều mô hình phức tạp.
- Chưa có nền tảng lý thuyết mạnh mẽ để lựa chọn các công cụ tối ưu cho Deep Learning.

2. Relevant algorithms

2.1 Mạng Noron tích chập

Mạng nơron tích chập (còn gọi là ConvNet / CNN) là một thuật toán DeepLearning có thể lấy hình ảnh đầu vào, gán độ quan trọng (các trọng số - weightsvà độ lệch - bias có thể học được) cho các đặc trưng/đối tượng khác nhau tronghình ảnh và có thể phân biệt được từng đặc trưng/đối tượng này với nhau. Côngviệc tiền xử lý được yêu cầu cho mạng nơron tích chập thì ít hơn nhiều so vớicác thuật toán phân loại khác. Trong các phương thức sơ khai, các bộ lọc đượcthiết kế bằng tay (hand - engineered), với một quá trình huấn luyện để chọn racác bộ lọc/đặc trưng phù hợp thì mạng nơron tích chập lại có khả năng tự học đểchọn ra các bộ lọc/ đặc trưng tối ưu nhất.

Kiến trúc của nơron tích chập tương tự như mô hình kết nối của các nơron trongbộ não con người và được lấy cảm hứng từ hệ thống vỏ thị giác trong bộ não(visual cortex). Các nơ-ron riêng lẻ chỉ phản ứng với các kích thích trong mộtkhu vực hạn chế của trường thị giác được gọi là Trường tiếp nhận (ReceptiveField). Một tập hợp các trường như vậy chồng lên nhau để bao phủ toàn bộ khuvực thị giác.



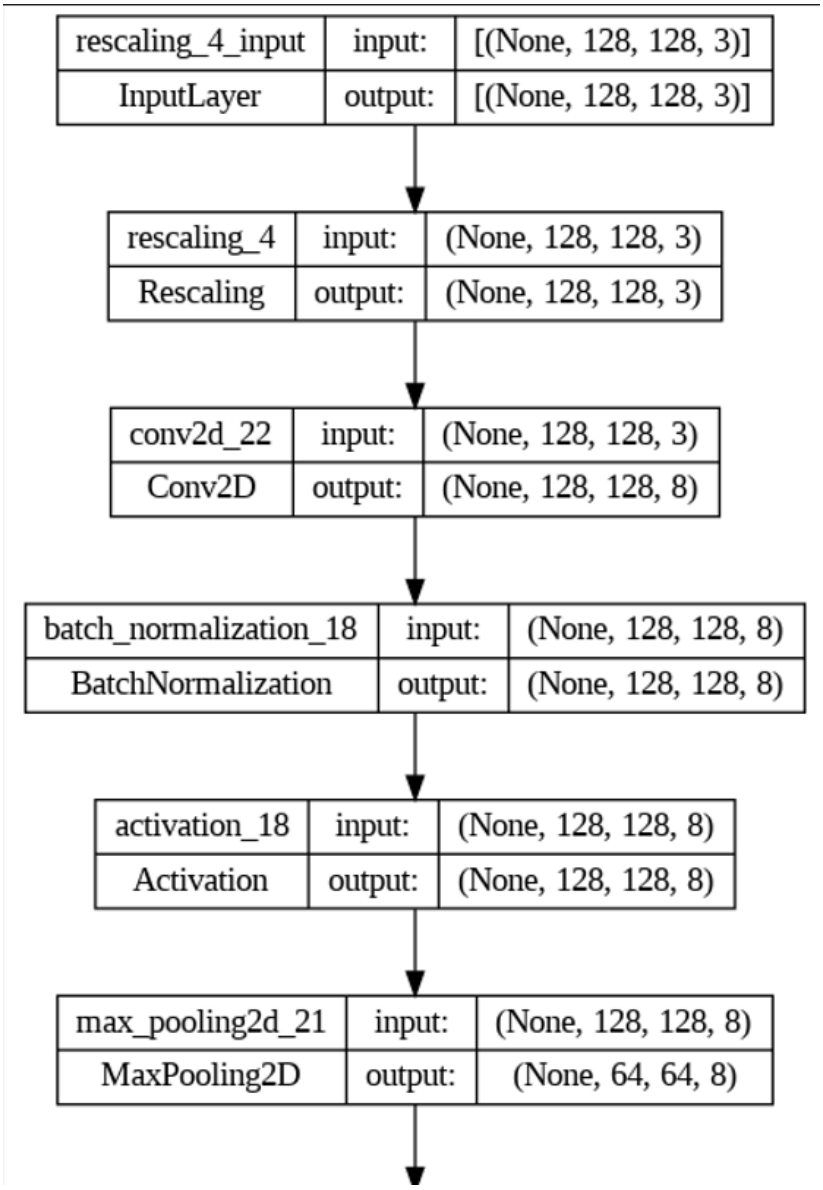
Hình 2.1: Mạng Noron tích chập

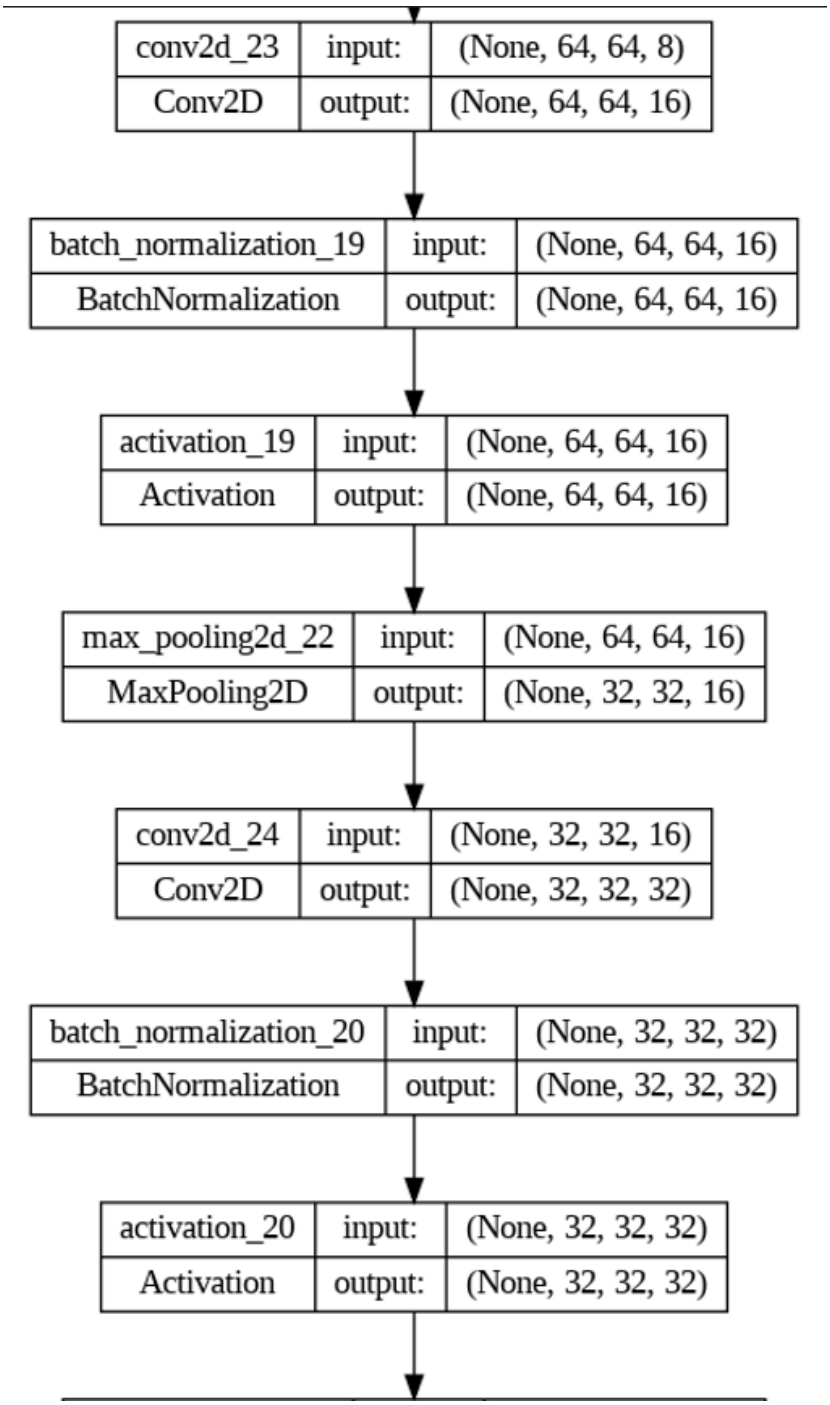
2.2 Adam Optimizer

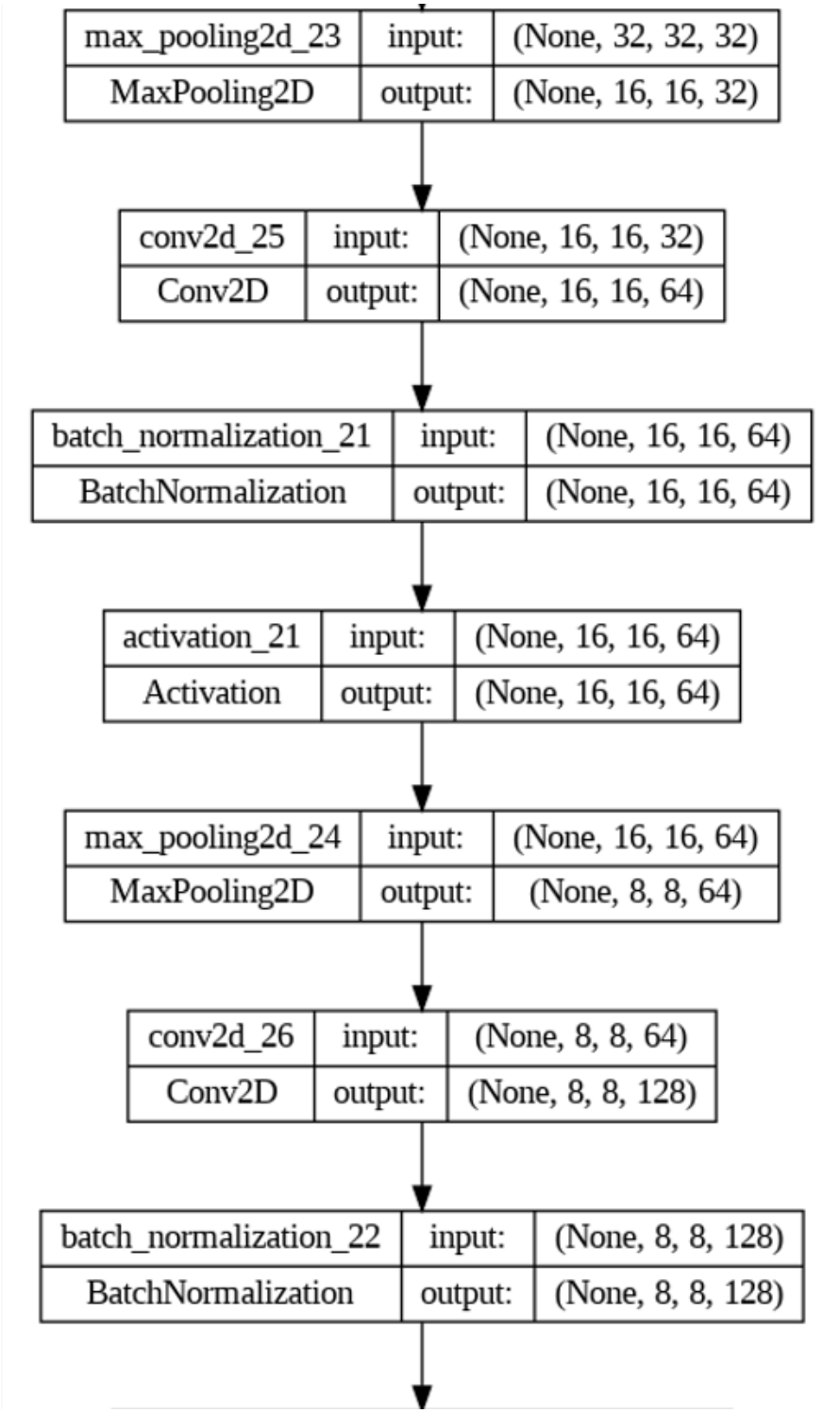
Adam Optimizer là một thuật toán tối ưu hóa được thiết kế đặc biệt cho việc học sâu. Đó là một phương pháp tính toán tốc độ học thích ứng cho các thông số khác nhau. Nói một cách đơn giản hơn, nó sử dụng tốc độ học thích ứng để giúp các mô hình học sâu học hiệu quả hơn. Trình tối ưu hóa này nổi bật nhờ hiệu quả tính

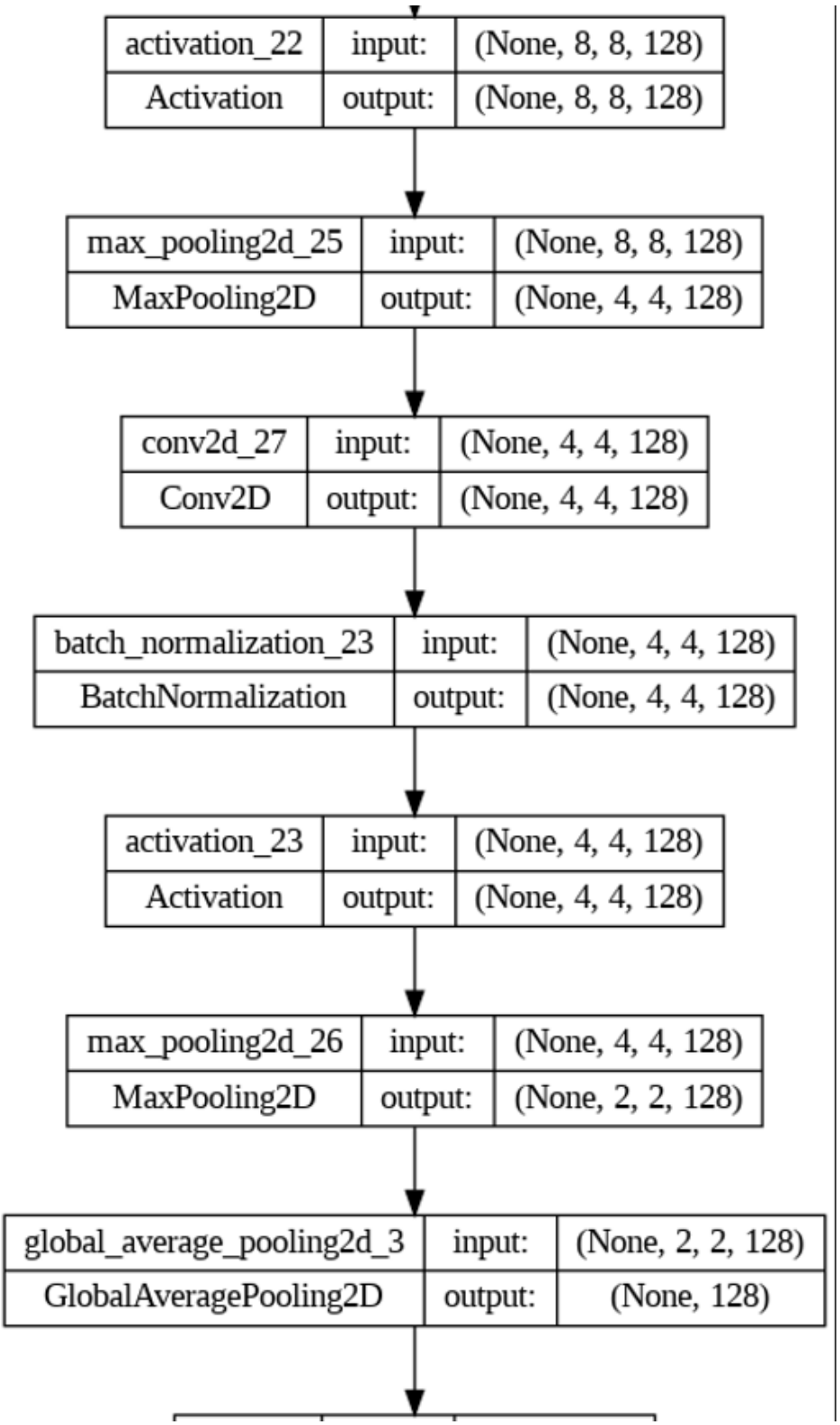
toán vượt trội, độ bền cao đối với độ dốc lớn, độ nhiễu và khả năng xử lý độ dốc thưa thớt trong các vấn đề phức tạp

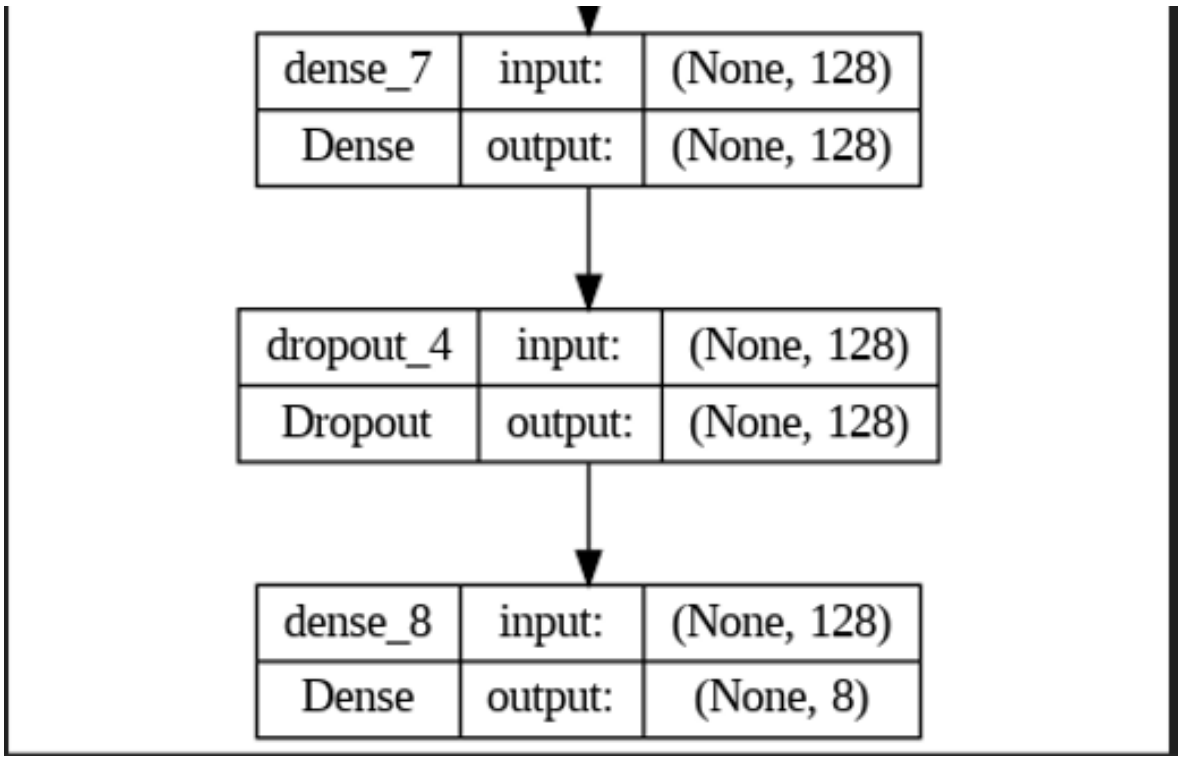
3. Network architecture




conv2d_23 | input: | (None, 64, 64, 8)
Conv2D | output: | (None, 64, 64, 16)
batch_normalization_19 | input: | (None, 64, 64, 16)
BatchNormalization | output: | (None, 64, 64, 16)
activation_19 | input: | (None, 64, 64, 16)
Activation | output: | (None, 64, 64, 16)
max_pooling2d_22 | input: | (None, 64, 64, 16)
MaxPooling2D | output: | (None, 32, 32, 16)
conv2d_24 | input: | (None, 32, 32, 16)
Conv2D | output: | (None, 32, 32, 32)
batch_normalization_20 | input: | (None, 32, 32, 32)
BatchNormalization | output: | (None, 32, 32, 32)
activation_20 | input: | (None, 32, 32, 32)
Activation | output: | (None, 32, 32, 32)


max_pooling2d_23 | input: | (None, 32, 32, 32)
MaxPooling2D | output: | (None, 16, 16, 32)
conv2d_25 | input: | (None, 16, 16, 32)
Conv2D | output: | (None, 16, 16, 64)
batch_normalization_21 | input: | (None, 16, 16, 64)
BatchNormalization | output: | (None, 16, 16, 64)
activation_21 | input: | (None, 16, 16, 64)
Activation | output: | (None, 16, 16, 64)
max_pooling2d_24 | input: | (None, 16, 16, 64)
MaxPooling2D | output: | (None, 8, 8, 64)
conv2d_26 | input: | (None, 8, 8, 64)
Conv2D | output: | (None, 8, 8, 128)
batch_normalization_22 | input: | (None, 8, 8, 128)
BatchNormalization | output: | (None, 8, 8, 128)


activation_22 | input: | (None, 8, 8, 128)
Activation | output: | (None, 8, 8, 128)
max_pooling2d_25 | input: | (None, 8, 8, 128)
MaxPooling2D | output: | (None, 4, 4, 128)
conv2d_27 | input: | (None, 4, 4, 128)
Conv2D | output: | (None, 4, 4, 128)
batch_normalization_23 | input: | (None, 4, 4, 128)
BatchNormalization | output: | (None, 4, 4, 128)
activation_23 | input: | (None, 4, 4, 128)
Activation | output: | (None, 4, 4, 128)
max_pooling2d_26 | input: | (None, 4, 4, 128)
MaxPooling2D | output: | (None, 2, 2, 128)
global_average_pooling2d_3 | input: | (None, 2, 2, 128)
GlobalAveragePooling2D | output: | (None, 128)


dense_7 | input: | (None, 128)
Dense | output: | (None, 128)
dropout_4 | input: | (None, 128)
Dropout | output: | (None, 128)
dense_8 | input: | (None, 128)
Dense | output: | (None, 8)

4. Network architecture

- Mô hình bao gồm một chuỗi các layer và các lớp như Convolutional, BatchNormalization, Activation và MaxPooling2D. Cuối cùng là các lớp GlobalAveragePooling2D và Dense để đưa ra dự đoán
- Trước khi đưa vào mô hình, hình ảnh được chuẩn hóa bằng lớp **Rescaling(1./255)** để đảm bảo giá trị pixel nằm trong khoảng [0,1].
- Dùng trình tối ưu hóa Adam và hàm mất mát SparseCategoricalCrossentropy để biên dịch mô hình
- Mô hình được đào tạo thông qua phương thức **fit()** với dữ liệu được chia thành tập huấn luyện và tập validation.
- Dùng thư viện Matplotlib để vẽ biểu đồ cho độ chính xác và mất mát trên cả tập huấn luyện và tập validation qua các epochs.

5. Code

```
import matplotlib.pyplot as plt
import numpy as np
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
from keras.layers import BatchNormalization
from tensorflow.keras.models import Sequential

import pathlib
training_dir = pathlib.Path('../input/radardataset/training_set')
training_count = len(list(training_dir.glob('*/*.png')))
print(training_count)

test_dir = pathlib.Path('../input/radardataset/test_set')
test_count = len(list(test_dir.glob('*/*.png')))
print(test_count)

batch_size = 64
img_height = 128
img_width = 128
train_ds = tf.keras.utils.image_dataset_from_directory(
  training_dir,
  validation_split=0.2,
  subset='training',
  seed=123,
  image_size=(img_height, img_width),
  batch_size=batch_size)

val_ds = tf.keras.utils.image_dataset_from_directory(
```

```
  test_dir,
  validation_split=0.2,
  subset='validation',
  seed=123,
  image_size=(img_height, img_width),
  batch_size=batch_size)

class_names = train_ds.class_names
print(class_names)
num_classes = len(class_names)
# Xây dựng mô hình 300,000 tham số
model = Sequential([
    layers.Rescaling(1./255, input_shape=(128, 128, 3)),
    layers.Conv2D(8, (3, 3), padding='same'),
    layers.BatchNormalization(),
    layers.Activation('relu'),
    layers.MaxPooling2D(),
    layers.Conv2D(16, (3, 3), padding='same'),
    layers.BatchNormalization(),
    layers.Activation('relu'),
    layers.MaxPooling2D(),
    layers.Conv2D(32, (3, 3), padding='same'),
    layers.BatchNormalization(),
    layers.Activation('relu'),
    layers.MaxPooling2D(),
    layers.Conv2D(64, (3, 3), padding='same'),
    layers.BatchNormalization(),
    layers.Activation('relu'),
    layers.MaxPooling2D(),
    layers.Conv2D(128, (3, 3), padding='same'),
    layers.BatchNormalization(),
    layers.Activation('relu'),
    layers.MaxPooling2D(),
    layers.Conv2D(128, (3, 3), padding='same'),
    layers.BatchNormalization(),
    layers.Activation('relu'),
    layers.MaxPooling2D(),

    layers.GlobalAveragePooling2D(),
    layers.Dense(128, activation='relu'),
    layers.Dropout(0.2),
    layers.Dense(len(class_names), activation='softmax')
])

model.summary()

# Compile mô hình
model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=0.001),
              loss='sparse_categorical_crossentropy',
             metrics=['accuracy'])

```

```
# Callback để giảm learning rate nếu cần
learning_rate_reduction = tf.keras.callbacks.ReduceLROnPlateau(
    monitor='val_loss', factor=0.2, patience=2, verbose=1)

# Huấn luyện mô hình
epochs = 40
history = model.fit(
    train_ds,
    validation_data=val_ds,
    epochs=epochs,shuffle=True, callbacks=[learning_rate_reduction])

# Đánh giá độ chính xác
accuracy = model.evaluate(val_ds)

    # Vẽ đồ thị training và validation loss, accuracy
    acc = history.history['accuracy']
    val_acc = history.history['val_accuracy']
    loss = history.history['loss']
    val_loss = history.history['val_loss']
    epochs_range = range(epochs)

    plt.figure(figsize=(16, 8))
    plt.subplot(1, 2, 1)
    plt.plot(epochs_range, acc, label='Training Accuracy')
    plt.plot(epochs_range, val_acc, label='Validation Accuracy')
    plt.legend(loc='lower right')
    plt.title('Training and Validation Accuracy')

    plt.subplot(1, 2, 2)
    plt.plot(epochs_range, loss, label='Training Loss')
    plt.plot(epochs_range, val_loss, label='Validation Loss')
    plt.legend(loc='upper right')
    plt.title('Training and Validation Loss')
    plt.show()

```

6. Simulation results

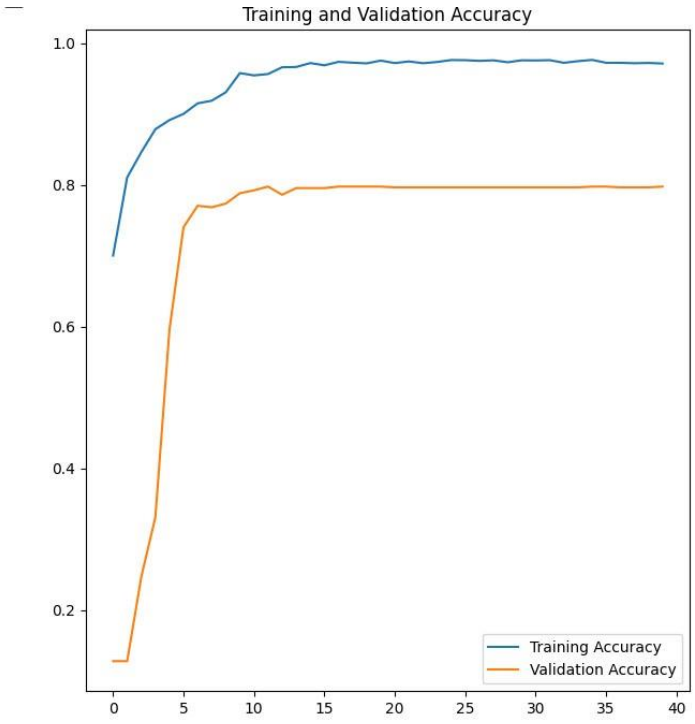
- ***Chạy lần 1:***

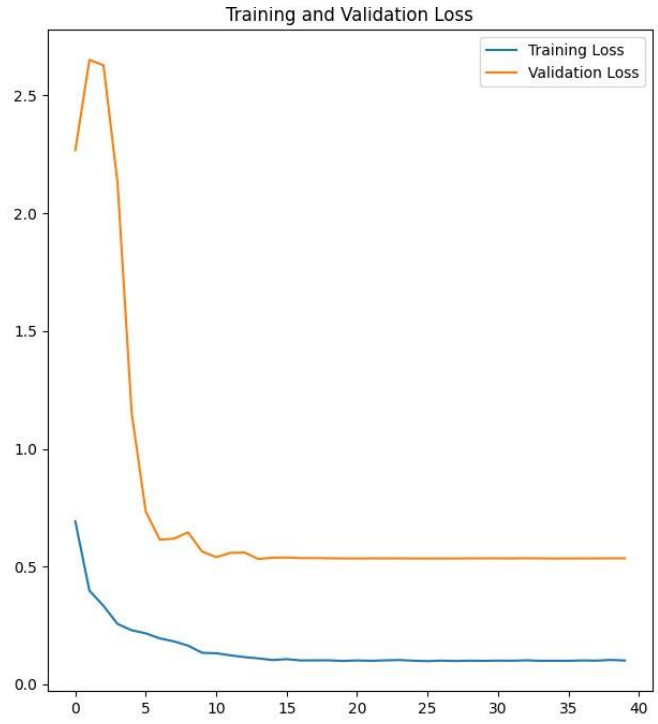
```
6400
4800
Found 6400 files belonging to 8 classes.
Using 5120 files for training.
Found 4800 files belonging to 8 classes.
Using 960 files for validation.
['B-FM', 'Barker', 'CPFSK', 'DSB-AM', 'GFSK', 'LFM', 'Rect', 'SSB-AM']
Model: "sequential_4"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 rescaling_4 (Rescaling)     (None, 128, 128, 3)       0

 conv2d_22 (Conv2D)          (None, 128, 128, 8)       224

 batch_normalization_18 (Ba  (None, 128, 128, 8)       32
 tchNormalization)

 activation_18 (Activation)  (None, 128, 128, 8)       0

 max_pooling2d_21 (MaxPooli  (None, 64, 64, 8)         0
 ng2D)

 conv2d_23 (Conv2D)          (None, 64, 64, 16)        1168

 batch_normalization_19 (Ba  (None, 64, 64, 16)        64
 tchNormalization)

 activation_19 (Activation)  (None, 64, 64, 16)        0

 max_pooling2d_22 (MaxPooli  (None, 32, 32, 16)        0
 ng2D)

 conv2d_24 (Conv2D)          (None, 32, 32, 32)        4640

 batch_normalization_20 (Ba  (None, 32, 32, 32)        128
 tchNormalization)

 activation_20 (Activation)  (None, 32, 32, 32)        0

 max_pooling2d_23 (MaxPooli  (None, 16, 16, 32)        0
 ng2D)

 conv2d_25 (Conv2D)          (None, 16, 16, 64)        18496

 batch_normalization_21 (Ba  (None, 16, 16, 64)        256
```

```
conv2d_25 (Conv2D)          (None, 16, 16, 64)        18496

batch_normalization_21 (Ba  (None, 16, 16, 64)        256
tchNormalization)

activation_21 (Activation)  (None, 16, 16, 64)        0

max_pooling2d_24 (MaxPooli  (None, 8, 8, 64)          0
ng2D)

conv2d_26 (Conv2D)          (None, 8, 8, 128)         73856

batch_normalization_22 (Ba  (None, 8, 8, 128)         512
tchNormalization)

activation_22 (Activation)  (None, 8, 8, 128)         0

max_pooling2d_25 (MaxPooli  (None, 4, 4, 128)         0
ng2D)

conv2d_27 (Conv2D)          (None, 4, 4, 128)         147584

batch_normalization_23 (Ba  (None, 4, 4, 128)         512
tchNormalization)

activation_23 (Activation)  (None, 4, 4, 128)         0

max_pooling2d_26 (MaxPooli  (None, 2, 2, 128)         0
ng2D)

global_average_pooling2d_3  (None, 128)               0
 (GlobalAveragePooling2D)

dense_7 (Dense)             (None, 128)               16512

dropout_4 (Dropout)         (None, 128)               0

dense_8 (Dense)             (None, 8)                 1032

=================================================================
Total params: 265016 (1.01 MB)
Trainable params: 264264 (1.01 MB)
Non-trainable params: 752 (2.94 KB)
_________________________________________________________________
Epoch 1/40
80/80 [==============================] - 8s 59ms/step - loss: 0.6915 - accuracy: 6
```

```
Epoch 1/40
80/80 [==============================] - 8s 50ms/step - loss: 0.6915 - accuracy: 0.7006 - val_loss: 2.2680 - val_accuracy: 0.1281 - lr: 0.0010
Epoch 2/40
80/80 [==============================] - 4s 43ms/step - loss: 0.3979 - accuracy: 0.8109 - val_loss: 2.6512 - val_accuracy: 0.1281 - lr: 0.0010
Epoch 3/40
79/80 [============================>.] - ETA: 0s - loss: 0.3343 - accuracy: 0.8459
Epoch 3: ReduceLROnPlateau reducing learning rate to 0.00020000000949949026.
80/80 [==============================] - 3s 34ms/step - loss: 0.3328 - accuracy: 0.8465 - val_loss: 2.6285 - val_accuracy: 0.2469 - lr: 0.0010
Epoch 4/40
80/80 [==============================] - 3s 37ms/step - loss: 0.2563 - accuracy: 0.8789 - val_loss: 2.1269 - val_accuracy: 0.3313 - lr: 2.0000e-04
Epoch 5/40
80/80 [==============================] - 3s 34ms/step - loss: 0.2295 - accuracy: 0.8918 - val_loss: 1.1518 - val_accuracy: 0.5958 - lr: 2.0000e-04
Epoch 6/40
80/80 [==============================] - 3s 37ms/step - loss: 0.2165 - accuracy: 0.9006 - val_loss: 0.7331 - val_accuracy: 0.7406 - lr: 2.0000e-04
Epoch 7/40
80/80 [==============================] - 3s 37ms/step - loss: 0.1949 - accuracy: 0.9154 - val_loss: 0.6142 - val_accuracy: 0.7708 - lr: 2.0000e-04
Epoch 8/40
80/80 [==============================] - 4s 51ms/step - loss: 0.1822 - accuracy: 0.9191 - val_loss: 0.6184 - val_accuracy: 0.7688 - lr: 2.0000e-04
Epoch 9/40
79/80 [============================>.] - ETA: 0s - loss: 0.1647 - accuracy: 0.9310
Epoch 9: ReduceLROnPlateau reducing learning rate to 4.0000001899898055e-05.
80/80 [==============================] - 3s 35ms/step - loss: 0.1644 - accuracy: 0.9311 - val_loss: 0.6453 - val_accuracy: 0.7740 - lr: 2.0000e-04
Epoch 10/40
80/80 [==============================] - 3s 37ms/step - loss: 0.1339 - accuracy: 0.9582 - val_loss: 0.5637 - val_accuracy: 0.7885 - lr: 4.0000e-05
Epoch 11/40
80/80 [==============================] - 4s 50ms/step - loss: 0.1319 - accuracy: 0.9549 - val_loss: 0.5398 - val_accuracy: 0.7927 - lr: 4.0000e-05
Epoch 12/40
80/80 [==============================] - 3s 37ms/step - loss: 0.1231 - accuracy: 0.9568 - val_loss: 0.5580 - val_accuracy: 0.7979 - lr: 4.0000e-05
Epoch 13/40
79/80 [============================>.] - ETA: 0s - loss: 0.1159 - accuracy: 0.9662
Epoch 13: ReduceLROnPlateau reducing learning rate to 8.000000525498762e-06.
80/80 [==============================] - 3s 36ms/step - loss: 0.1155 - accuracy: 0.9664 - val_loss: 0.5596 - val_accuracy: 0.7865 - lr: 4.0000e-05
Epoch 14/40
80/80 [==============================] - 3s 35ms/step - loss: 0.1101 - accuracy: 0.9668 - val_loss: 0.5322 - val_accuracy: 0.7958 - lr: 8.0000e-06
Epoch 15/40
80/80 [==============================] - 5s 57ms/step - loss: 0.1031 - accuracy: 0.9723 - val_loss: 0.5379 - val_accuracy: 0.7958 - lr: 8.0000e-06
Epoch 16/40
78/80 [============================>.] - ETA: 0s - loss: 0.1077 - accuracy: 0.9685
Epoch 16: ReduceLROnPlateau reducing learning rate to 1.6000001778593287e-06.
80/80 [==============================] - 3s 38ms/step - loss: 0.1069 - accuracy: 0.9691 - val_loss: 0.5386 - val_accuracy: 0.7958 - lr: 8.0000e-06
Epoch 17/40
80/80 [==============================] - 4s 47ms/step - loss: 0.1014 - accuracy: 0.9740 - val_loss: 0.5358 - val_accuracy: 0.7979 - lr: 1.6000e-06
Epoch 18/40
78/80 [============================>.] - ETA: 0s - loss: 0.1020 - accuracy: 0.9730
Epoch 18: ReduceLROnPlateau reducing learning rate to 3.200000264769187e-07.
80/80 [==============================] - 3s 37ms/step - loss: 0.1017 - accuracy: 0.9729 - val_loss: 0.5360 - val_accuracy: 0.7979 - lr: 1.6000e-06
Epoch 19/40
80/80 [==============================] - 3s 36ms/step - loss: 0.1019 - accuracy: 0.9719 - val_loss: 0.5354 - val_accuracy: 0.7979 - lr: 3.2000e-07
Epoch 20/40
78/80 [============================>.] - ETA: 0s - loss: 0.0992 - accuracy: 0.9760
Epoch 20: ReduceLROnPlateau reducing learning rate to 6.400000529538374e-08.
80/80 [==============================] - 4s 47ms/step - loss: 0.0992 - accuracy: 0.9758 - val_loss: 0.5346 - val_accuracy: 0.7979 - lr: 3.2000e-07
Epoch 21/40
80/80 [==============================] - 3s 35ms/step - loss: 0.1015 - accuracy: 0.9723 - val_loss: 0.5345 - val_accuracy: 0.7969 - lr: 6.4000e-08
Epoch 22/40
78/80 [============================>.] - ETA: 0s - loss: 0.0991 - accuracy: 0.9752
Epoch 22: ReduceLROnPlateau reducing learning rate to 1.2800001059076749e-08.
80/80 [==============================] - 3s 38ms/step - loss: 0.0998 - accuracy: 0.9746 - val_loss: 0.5348 - val_accuracy: 0.7969 - lr: 6.4000e-08
Epoch 23/40
80/80 [==============================] - 3s 36ms/step - loss: 0.1018 - accuracy: 0.9721 - val_loss: 0.5347 - val_accuracy: 0.7969 - lr: 1.2800e-08
Epoch 24/40
79/80 [============================>.] - ETA: 0s - loss: 0.1032 - accuracy: 0.9737
Epoch 24: ReduceLROnPlateau reducing learning rate to 2.5600002118153498e-09.
80/80 [==============================] - 3s 35ms/step - loss: 0.1030 - accuracy: 0.9738 - val_loss: 0.5348 - val_accuracy: 0.7969 - lr: 1.2800e-08
Epoch 25/40
80/80 [==============================] - 3s 35ms/step - loss: 0.1003 - accuracy: 0.9766 - val_loss: 0.5344 - val_accuracy: 0.7969 - lr: 2.5600e-09
Epoch 26/40
79/80 [============================>.] - ETA: 0s - loss: 0.0990 - accuracy: 0.9763
Epoch 26: ReduceLROnPlateau reducing learning rate to 5.1200004236307e-10.
80/80 [==============================] - 3s 37ms/step - loss: 0.0984 - accuracy: 0.9764 - val_loss: 0.5344 - val_accuracy: 0.7969 - lr: 2.5600e-09
Epoch 27/40
80/80 [==============================] - 3s 35ms/step - loss: 0.1006 - accuracy: 0.9754 - val_loss: 0.5343 - val_accuracy: 0.7969 - lr: 5.1200e-10
Epoch 28/40
78/80 [============================>.] - ETA: 0s - loss: 0.0993 - accuracy: 0.9762
Epoch 28: ReduceLROnPlateau reducing learning rate to 1.0240001069306004e-10.
80/80 [==============================] - 3s 35ms/step - loss: 0.0990 - accuracy: 0.9762 - val_loss: 0.5344 - val_accuracy: 0.7969 - lr: 5.1200e-10
Epoch 29/40
80/80 [==============================] - 4s 48ms/step - loss: 0.1003 - accuracy: 0.9734 - val_loss: 0.5349 - val_accuracy: 0.7969 - lr: 1.0240e-10
Epoch 30/40
79/80 [============================>.] - ETA: 0s - loss: 0.0998 - accuracy: 0.9763
Epoch 30: ReduceLROnPlateau reducing learning rate to 2.0480002416167767e-11.
```

```
Epoch 30/40
79/80 [============================>.] - ETA: 0s - loss: 0.0998 - accuracy: 0.9763
Epoch 30: ReduceLROnPlateau reducing learning rate to 2.0480002416167767e-11.
80/80 [==============================] - 3s 38ms/step - loss: 0.0997 - accuracy: 0.9762 - val_loss: 0.5349 - val_accuracy: 0.7969 - lr: 1.0240e-10
Epoch 31/40
80/80 [==============================] - 3s 35ms/step - loss: 0.1006 - accuracy: 0.9760 - val_loss: 0.5350 - val_accuracy: 0.7969 - lr: 2.0480e-11
Epoch 32/40
80/80 [==============================] - ETA: 0s - loss: 0.1001 - accuracy: 0.9764
Epoch 32: ReduceLROnPlateau reducing learning rate to 4.096000622011431e-12.
80/80 [==============================] - 3s 41ms/step - loss: 0.1001 - accuracy: 0.9764 - val_loss: 0.5345 - val_accuracy: 0.7969 - lr: 2.0480e-11
Epoch 33/40
80/80 [==============================] - 4s 44ms/step - loss: 0.1019 - accuracy: 0.9727 - val_loss: 0.5353 - val_accuracy: 0.7969 - lr: 4.0960e-12
Epoch 34/40
79/80 [============================>.] - ETA: 0s - loss: 0.1003 - accuracy: 0.9747
Epoch 34: ReduceLROnPlateau reducing learning rate to 8.192000897078167e-13.
80/80 [==============================] - 3s 35ms/step - loss: 0.0997 - accuracy: 0.9750 - val_loss: 0.5348 - val_accuracy: 0.7969 - lr: 4.0960e-12
Epoch 35/40
80/80 [==============================] - 3s 34ms/step - loss: 0.1002 - accuracy: 0.9768 - val_loss: 0.5343 - val_accuracy: 0.7979 - lr: 8.1920e-13
Epoch 36/40
78/80 [============================>.] - ETA: 0s - loss: 0.1006 - accuracy: 0.9720
Epoch 36: ReduceLROnPlateau reducing learning rate to 1.6384001360475466e-13.
80/80 [==============================] - 4s 53ms/step - loss: 0.0996 - accuracy: 0.9727 - val_loss: 0.5347 - val_accuracy: 0.7979 - lr: 8.1920e-13
Epoch 37/40
80/80 [==============================] - 3s 38ms/step - loss: 0.1014 - accuracy: 0.9727 - val_loss: 0.5346 - val_accuracy: 0.7969 - lr: 1.6384e-13
Epoch 38/40
78/80 [============================>.] - ETA: 0s - loss: 0.1004 - accuracy: 0.9726
Epoch 38: ReduceLROnPlateau reducing learning rate to 3.2768002178849846e-14.
80/80 [==============================] - 3s 39ms/step - loss: 0.1005 - accuracy: 0.9721 - val_loss: 0.5347 - val_accuracy: 0.7969 - lr: 1.6384e-13
Epoch 39/40
80/80 [==============================] - 4s 47ms/step - loss: 0.1036 - accuracy: 0.9725 - val_loss: 0.5350 - val_accuracy: 0.7969 - lr: 3.2768e-14
Epoch 40/40
79/80 [============================>.] - ETA: 0s - loss: 0.1006 - accuracy: 0.9719
Epoch 40: ReduceLROnPlateau reducing learning rate to 6.553600300244697e-15.
80/80 [==============================] - 3s 36ms/step - loss: 0.1011 - accuracy: 0.9717 - val_loss: 0.5351 - val_accuracy: 0.7979 - lr: 3.2768e-14
15/15 [==============================] - 0s 20ms/step - loss: 0.5351 - accuracy: 0.7979
```





Trong 40 epochs mô hình đạt độ mất mát 0.5351 (val_loss) và độ chính xác là 0.7979 (val_accuracy).

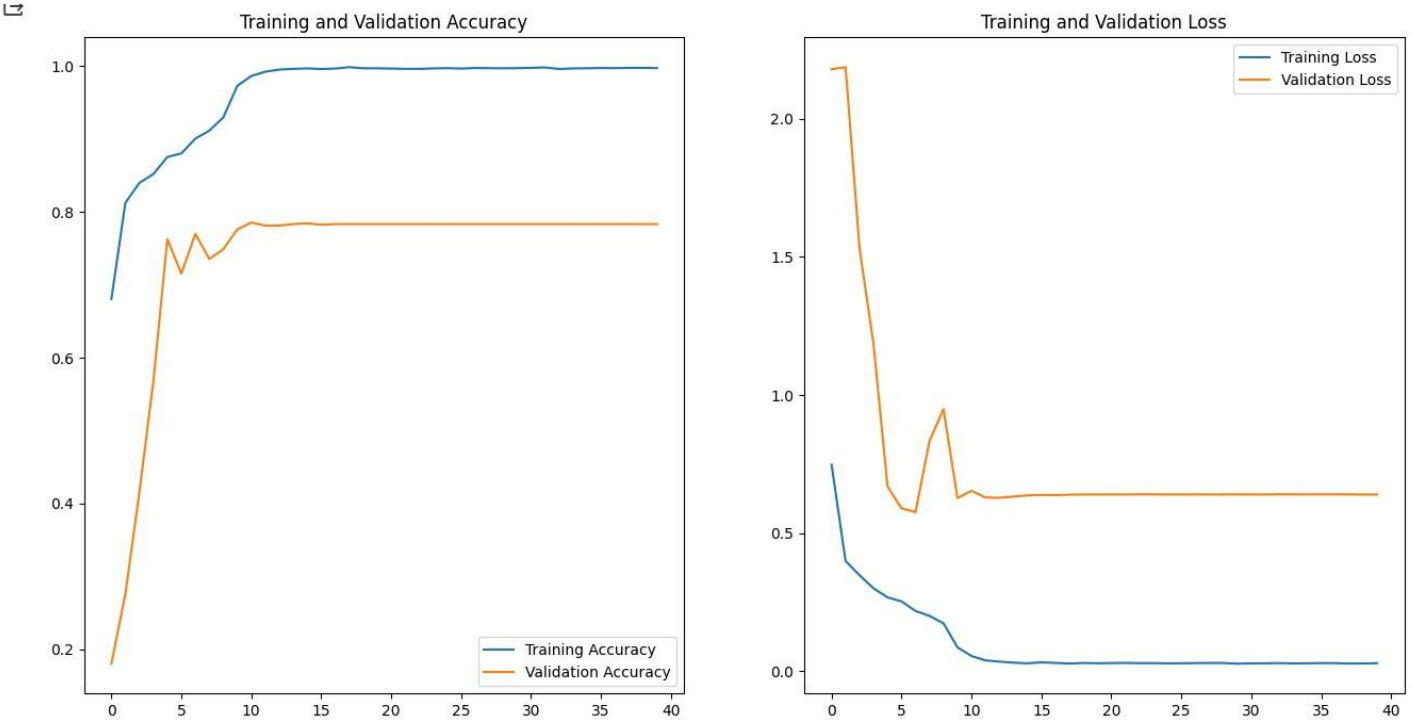
- ***Chạy lần 2:***

```
6400
4800
Found 6400 files belonging to 8 classes.
Using 5120 files for training.
Found 4800 files belonging to 8 classes.
Using 960 files for validation.
['B-FM', 'Barker', 'CPFSK', 'DSB-AM', 'GFSK', 'LFM', 'Rect', 'SSB-AM']
Model: "sequential_5"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 rescaling_5 (Rescaling)     (None, 128, 128, 3)       0

 conv2d_28 (Conv2D)          (None, 128, 128, 8)       224

 batch_normalization_24 (Ba  (None, 128, 128, 8)       32
 tchNormalization)

 activation_24 (Activation)  (None, 128, 128, 8)       0

 max_pooling2d_27 (MaxPooli  (None, 64, 64, 8)         0
 ng2D)

 conv2d_29 (Conv2D)          (None, 64, 64, 16)        1168

 batch_normalization_25 (Ba  (None, 64, 64, 16)        64
 tchNormalization)

 activation_25 (Activation)  (None, 64, 64, 16)        0

 max_pooling2d_28 (MaxPooli  (None, 32, 32, 16)        0
 ng2D)

 conv2d_30 (Conv2D)          (None, 32, 32, 32)        4640

 batch_normalization_26 (Ba  (None, 32, 32, 32)        128
 tchNormalization)

 activation_26 (Activation)  (None, 32, 32, 32)        0

 max_pooling2d_29 (MaxPooli  (None, 16, 16, 32)        0
 ng2D)

 conv2d_31 (Conv2D)          (None, 16, 16, 64)        18496

 batch_normalization_27 (Ba  (None, 16, 16, 64)        256
```

```
batch_normalization_27 (Ba  (None, 16, 16, 64)        256
tchNormalization)

activation_27 (Activation)  (None, 16, 16, 64)        0

max_pooling2d_30 (MaxPooli  (None, 8, 8, 64)          0
ng2D)

conv2d_32 (Conv2D)          (None, 8, 8, 128)         73856

batch_normalization_28 (Ba  (None, 8, 8, 128)         512
tchNormalization)

activation_28 (Activation)  (None, 8, 8, 128)         0

max_pooling2d_31 (MaxPooli  (None, 4, 4, 128)         0
ng2D)

conv2d_33 (Conv2D)          (None, 4, 4, 128)         147584

batch_normalization_29 (Ba  (None, 4, 4, 128)         512
tchNormalization)

activation_29 (Activation)  (None, 4, 4, 128)         0

max_pooling2d_32 (MaxPooli  (None, 2, 2, 128)         0
ng2D)

global_average_pooling2d_4  (None, 128)               0
 (GlobalAveragePooling2D)

dense_9 (Dense)             (None, 128)               16512

dropout_5 (Dropout)         (None, 128)               0

dense_10 (Dense)            (None, 8)                 1032

=================================================================
Total params: 265016 (1.01 MB)
Trainable params: 264264 (1.01 MB)
Non-trainable params: 752 (2.94 KB)
_________________________________________________________________
Epoch 1/40
80/80 [==============================] - 7s 38ms/step - loss: 0.7477 - accuracy: 0.6807 - val_loss: 2.1801 - val_accuracy: 0.1802 - lr: 0.0010
Epoch 2/40
```

```
Epoch 2/40
80/80 [==============================] - 4s 42ms/step - loss: 0.3991 - accuracy: 0.8127 - val_loss: 2.1876 - val_accuracy: 0.2760 - lr: 0.0010
Epoch 3/40
80/80 [==============================] - 3s 36ms/step - loss: 0.3474 - accuracy: 0.8398 - val_loss: 1.5306 - val_accuracy: 0.4156 - lr: 0.0010
Epoch 4/40
80/80 [==============================] - 3s 37ms/step - loss: 0.2999 - accuracy: 0.8520 - val_loss: 1.1842 - val_accuracy: 0.5677 - lr: 0.0010
Epoch 5/40
80/80 [==============================] - 3s 36ms/step - loss: 0.2674 - accuracy: 0.8754 - val_loss: 0.6692 - val_accuracy: 0.7625 - lr: 0.0010
Epoch 6/40
80/80 [==============================] - 3s 39ms/step - loss: 0.2527 - accuracy: 0.8803 - val_loss: 0.5894 - val_accuracy: 0.7156 - lr: 0.0010
Epoch 7/40
80/80 [==============================] - 3s 38ms/step - loss: 0.2181 - accuracy: 0.9006 - val_loss: 0.5765 - val_accuracy: 0.7698 - lr: 0.0010
Epoch 8/40
80/80 [==============================] - 4s 51ms/step - loss: 0.2001 - accuracy: 0.9115 - val_loss: 0.8338 - val_accuracy: 0.7354 - lr: 0.0010
Epoch 9/40
79/80 [============================>.] - ETA: 0s - loss: 0.1726 - accuracy: 0.9302
Epoch 9: ReduceLROnPlateau reducing learning rate to 0.00020000000949949026.
80/80 [==============================] - 3s 37ms/step - loss: 0.1733 - accuracy: 0.9297 - val_loss: 0.9491 - val_accuracy: 0.7490 - lr: 0.0010
Epoch 10/40
80/80 [==============================] - 3s 34ms/step - loss: 0.0861 - accuracy: 0.9730 - val_loss: 0.6268 - val_accuracy: 0.7760 - lr: 2.0000e-04
Epoch 11/40
78/80 [============================>.] - ETA: 0s - loss: 0.0556 - accuracy: 0.9864
Epoch 11: ReduceLROnPlateau reducing learning rate to 4.0000001899898055e-05.
80/80 [==============================] - 4s 52ms/step - loss: 0.0549 - accuracy: 0.9865 - val_loss: 0.6534 - val_accuracy: 0.7854 - lr: 2.0000e-04
Epoch 12/40
80/80 [==============================] - 3s 38ms/step - loss: 0.0391 - accuracy: 0.9924 - val_loss: 0.6292 - val_accuracy: 0.7812 - lr: 4.0000e-05
Epoch 13/40
79/80 [============================>.] - ETA: 0s - loss: 0.0346 - accuracy: 0.9953
Epoch 13: ReduceLROnPlateau reducing learning rate to 8.000000525498762e-06.
80/80 [==============================] - 3s 38ms/step - loss: 0.0345 - accuracy: 0.9953 - val_loss: 0.6276 - val_accuracy: 0.7812 - lr: 4.0000e-05
Epoch 14/40
80/80 [==============================] - 4s 52ms/step - loss: 0.0308 - accuracy: 0.9963 - val_loss: 0.6323 - val_accuracy: 0.7833 - lr: 8.0000e-06
Epoch 15/40
79/80 [============================>.] - ETA: 0s - loss: 0.0286 - accuracy: 0.9968
Epoch 15: ReduceLROnPlateau reducing learning rate to 1.6000001778593287e-06.
80/80 [==============================] - 3s 34ms/step - loss: 0.0285 - accuracy: 0.9969 - val_loss: 0.6367 - val_accuracy: 0.7844 - lr: 8.0000e-06
Epoch 16/40
80/80 [==============================] - 3s 36ms/step - loss: 0.0319 - accuracy: 0.9961 - val_loss: 0.6383 - val_accuracy: 0.7823 - lr: 1.6000e-06
Epoch 17/40
79/80 [============================>.] - ETA: 0s - loss: 0.0297 - accuracy: 0.9966
Epoch 17: ReduceLROnPlateau reducing learning rate to 3.200000264769187e-07.
80/80 [==============================] - 3s 35ms/step - loss: 0.0298 - accuracy: 0.9967 - val_loss: 0.6378 - val_accuracy: 0.7833 - lr: 1.6000e-06
Epoch 18/40
80/80 [==============================] - 3s 35ms/step - loss: 0.0276 - accuracy: 0.9986 - val_loss: 0.6391 - val_accuracy: 0.7833 - lr: 3.2000e-07
Epoch 19/40
```

```
Epoch 19/40
78/80 [============================>.] - ETA: 0s - loss: 0.0298 - accuracy: 0.9970
Epoch 19: ReduceLROnPlateau reducing learning rate to 6.400000529538374e-08.
80/80 [==============================] - 3s 35ms/step - loss: 0.0297 - accuracy: 0.9971 - val_loss: 0.6399 - val_accuracy: 0.7833 - lr: 3.2000e-07
Epoch 20/40
80/80 [==============================] - 5s 59ms/step - loss: 0.0286 - accuracy: 0.9971 - val_loss: 0.6398 - val_accuracy: 0.7833 - lr: 6.4000e-08
Epoch 21/40
79/80 [============================>.] - ETA: 0s - loss: 0.0295 - accuracy: 0.9966
Epoch 21: ReduceLROnPlateau reducing learning rate to 1.2800001059076749e-08.
80/80 [==============================] - 3s 35ms/step - loss: 0.0293 - accuracy: 0.9967 - val_loss: 0.6399 - val_accuracy: 0.7833 - lr: 6.4000e-08
Epoch 22/40
80/80 [==============================] - 3s 35ms/step - loss: 0.0298 - accuracy: 0.9963 - val_loss: 0.6401 - val_accuracy: 0.7833 - lr: 1.2800e-08
Epoch 23/40
80/80 [==============================] - ETA: 0s - loss: 0.0290 - accuracy: 0.9963
Epoch 23: ReduceLROnPlateau reducing learning rate to 2.5600002118153498e-09.
80/80 [==============================] - 4s 46ms/step - loss: 0.0290 - accuracy: 0.9963 - val_loss: 0.6403 - val_accuracy: 0.7833 - lr: 1.2800e-08
Epoch 24/40
80/80 [==============================] - 3s 36ms/step - loss: 0.0291 - accuracy: 0.9969 - val_loss: 0.6402 - val_accuracy: 0.7833 - lr: 2.5600e-09
Epoch 25/40
79/80 [============================>.] - ETA: 0s - loss: 0.0284 - accuracy: 0.9972
Epoch 25: ReduceLROnPlateau reducing learning rate to 5.1200004236307e-10.
80/80 [==============================] - 3s 38ms/step - loss: 0.0283 - accuracy: 0.9973 - val_loss: 0.6400 - val_accuracy: 0.7833 - lr: 2.5600e-09
Epoch 26/40
80/80 [==============================] - 4s 51ms/step - loss: 0.0287 - accuracy: 0.9967 - val_loss: 0.6400 - val_accuracy: 0.7833 - lr: 5.1200e-10
Epoch 27/40
79/80 [============================>.] - ETA: 0s - loss: 0.0293 - accuracy: 0.9974
Epoch 27: ReduceLROnPlateau reducing learning rate to 1.0240001069306004e-10.
80/80 [==============================] - 3s 35ms/step - loss: 0.0292 - accuracy: 0.9975 - val_loss: 0.6403 - val_accuracy: 0.7833 - lr: 5.1200e-10
Epoch 28/40
80/80 [==============================] - 3s 37ms/step - loss: 0.0296 - accuracy: 0.9973 - val_loss: 0.6397 - val_accuracy: 0.7833 - lr: 1.0240e-10
Epoch 29/40
79/80 [============================>.] - ETA: 0s - loss: 0.0295 - accuracy: 0.9972
Epoch 29: ReduceLROnPlateau reducing learning rate to 2.0480002416167767e-11.
80/80 [==============================] - 3s 35ms/step - loss: 0.0297 - accuracy: 0.9971 - val_loss: 0.6402 - val_accuracy: 0.7833 - lr: 1.0240e-10
Epoch 30/40
80/80 [==============================] - 4s 50ms/step - loss: 0.0269 - accuracy: 0.9973 - val_loss: 0.6401 - val_accuracy: 0.7833 - lr: 2.0480e-11
Epoch 31/40
78/80 [============================>.] - ETA: 0s - loss: 0.0288 - accuracy: 0.9976
Epoch 31: ReduceLROnPlateau reducing learning rate to 4.096000622011431e-12.
80/80 [==============================] - 3s 38ms/step - loss: 0.0284 - accuracy: 0.9977 - val_loss: 0.6402 - val_accuracy: 0.7833 - lr: 2.0480e-11
Epoch 32/40
80/80 [==============================] - 3s 40ms/step - loss: 0.0284 - accuracy: 0.9982 - val_loss: 0.6400 - val_accuracy: 0.7833 - lr: 4.0960e-12
Epoch 33/40
78/80 [============================>.] - ETA: 0s - loss: 0.0291 - accuracy: 0.9962
Epoch 33: ReduceLROnPlateau reducing learning rate to 8.192000897078167e-13.
80/80 [==============================] - 4s 49ms/step - loss: 0.0292 - accuracy: 0.9961 - val_loss: 0.6405 - val_accuracy: 0.7833 - lr: 4.0960e-12
Epoch 34/40
80/80 [==============================] - 3s 35ms/step - loss: 0.0280 - accuracy: 0.9969 - val_loss: 0.6404 - val_accuracy: 0.7833 - lr: 8.1920e-13
Epoch 35/40
78/80 [============================>.] - ETA: 0s - loss: 0.0286 - accuracy: 0.9970
Epoch 35: ReduceLROnPlateau reducing learning rate to 1.6384001360475466e-13.
80/80 [==============================] - 3s 35ms/step - loss: 0.0285 - accuracy: 0.9971 - val_loss: 0.6402 - val_accuracy: 0.7833 - lr: 8.1920e-13
Epoch 36/40
80/80 [==============================] - 3s 37ms/step - loss: 0.0291 - accuracy: 0.9975 - val_loss: 0.6404 - val_accuracy: 0.7833 - lr: 1.6384e-13
Epoch 37/40
78/80 [============================>.] - ETA: 0s - loss: 0.0293 - accuracy: 0.9974
Epoch 37: ReduceLROnPlateau reducing learning rate to 3.2768002178849846e-14.
80/80 [==============================] - 4s 46ms/step - loss: 0.0293 - accuracy: 0.9973 - val_loss: 0.6403 - val_accuracy: 0.7833 - lr: 1.6384e-13
Epoch 38/40
80/80 [==============================] - 3s 36ms/step - loss: 0.0276 - accuracy: 0.9977 - val_loss: 0.6403 - val_accuracy: 0.7833 - lr: 3.2768e-14
Epoch 39/40
78/80 [============================>.] - ETA: 0s - loss: 0.0278 - accuracy: 0.9976
Epoch 39: ReduceLROnPlateau reducing learning rate to 6.553600300244697e-15.
80/80 [==============================] - 3s 34ms/step - loss: 0.0280 - accuracy: 0.9977 - val_loss: 0.6397 - val_accuracy: 0.7833 - lr: 3.2768e-14
Epoch 40/40
80/80 [==============================] - 3s 39ms/step - loss: 0.0289 - accuracy: 0.9973 - val_loss: 0.6397 - val_accuracy: 0.7833 - lr: 6.5536e-15
15/15 [==============================] - 0s 21ms/step - loss: 0.6397 - accuracy: 0.7833
```



Trong 40 epochs mô hình đạt độ mất mát 0.6397 (val_loss) và độ chính xác là 0.7833 (val_accuracy).

- ***Chạy lần 3:***

```
6400
4800
Found 6400 files belonging to 8 classes.
Using 5120 files for training.
Found 4800 files belonging to 8 classes.
Using 960 files for validation.
['B-FM', 'Barker', 'CPFSK', 'DSB-AM', 'GFSK', 'LFM', 'Rect', 'SSB-AM']
Model: "sequential_6"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 rescaling_6 (Rescaling)     (None, 128, 128, 3)       0

 conv2d_34 (Conv2D)          (None, 128, 128, 8)       224

 batch_normalization_30 (Ba  (None, 128, 128, 8)       32
 tchNormalization)

 activation_30 (Activation)  (None, 128, 128, 8)       0

 max_pooling2d_33 (MaxPooli  (None, 64, 64, 8)         0
 ng2D)

 conv2d_35 (Conv2D)          (None, 64, 64, 16)        1168

 batch_normalization_31 (Ba  (None, 64, 64, 16)        64
 tchNormalization)

 activation_31 (Activation)  (None, 64, 64, 16)        0

 max_pooling2d_34 (MaxPooli  (None, 32, 32, 16)        0
 ng2D)

 conv2d_36 (Conv2D)          (None, 32, 32, 32)        4640

 batch_normalization_32 (Ba  (None, 32, 32, 32)        128
 tchNormalization)

 activation_32 (Activation)  (None, 32, 32, 32)        0

 max_pooling2d_35 (MaxPooli  (None, 16, 16, 32)        0
 ng2D)

 conv2d_37 (Conv2D)          (None, 16, 16, 64)        18496
```

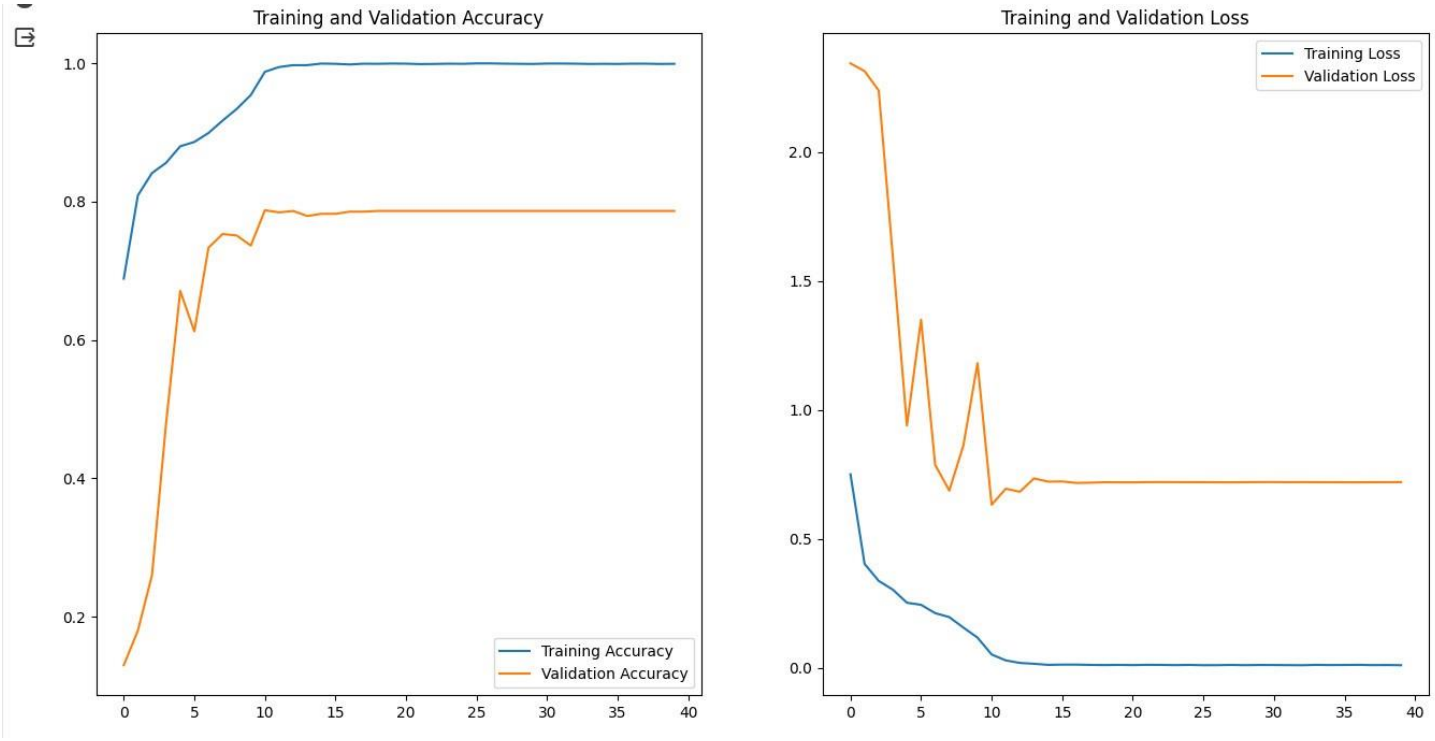
conv2d_37 (Conv2D)	(None, 16, 16, 64)	18496
batch_normalization_33 (BatchNormalization)	(None, 16, 16, 64)	256
activation_33 (Activation)	(None, 16, 16, 64)	0
max_pooling2d_36 (MaxPooling2D)	(None, 8, 8, 64)	0
conv2d_38 (Conv2D)	(None, 8, 8, 128)	73856
batch_normalization_34 (BatchNormalization)	(None, 8, 8, 128)	512
activation_34 (Activation)	(None, 8, 8, 128)	0
max_pooling2d_37 (MaxPooling2D)	(None, 4, 4, 128)	0
conv2d_39 (Conv2D)	(None, 4, 4, 128)	147584
batch_normalization_35 (BatchNormalization)	(None, 4, 4, 128)	512
activation_35 (Activation)	(None, 4, 4, 128)	0
max_pooling2d_38 (MaxPooling2D)	(None, 2, 2, 128)	0
global_average_pooling2d_5 (GlobalAveragePooling2D)	(None, 128)	0
dense_11 (Dense)	(None, 128)	16512
dropout_6 (Dropout)	(None, 128)	0
dense_12 (Dense)	(None, 8)	1032

```
=================================================================
Total params: 265016 (1.01 MB)
Trainable params: 264264 (1.01 MB)
Non-trainable params: 752 (2.94 KB)
```

```
Epoch 1/40
80/80 [==============================] - 8s 40ms/step - loss: 0.7508 - accuracy: 0.6887 - val_loss: 2.3437 - val_accuracy: 0.1302 - lr: 0.0010
Epoch 2/40
80/80 [==============================] - 3s 35ms/step - loss: 0.4032 - accuracy: 0.8090 - val_loss: 2.3136 - val_accuracy: 0.1802 - lr: 0.0010
Epoch 3/40
80/80 [==============================] - 3s 35ms/step - loss: 0.3374 - accuracy: 0.8412 - val_loss: 2.2384 - val_accuracy: 0.2604 - lr: 0.0010
Epoch 4/40
80/80 [==============================] - 4s 52ms/step - loss: 0.3036 - accuracy: 0.8561 - val_loss: 1.5947 - val_accuracy: 0.4802 - lr: 0.0010
Epoch 5/40
80/80 [==============================] - 3s 38ms/step - loss: 0.2528 - accuracy: 0.8799 - val_loss: 0.9400 - val_accuracy: 0.6708 - lr: 0.0010
Epoch 6/40
80/80 [==============================] - 3s 35ms/step - loss: 0.2447 - accuracy: 0.8861 - val_loss: 1.3498 - val_accuracy: 0.6125 - lr: 0.0010
Epoch 7/40
80/80 [==============================] - 4s 51ms/step - loss: 0.2125 - accuracy: 0.8990 - val_loss: 0.7874 - val_accuracy: 0.7333 - lr: 0.0010
Epoch 8/40
80/80 [==============================] - 3s 36ms/step - loss: 0.1970 - accuracy: 0.9170 - val_loss: 0.6875 - val_accuracy: 0.7531 - lr: 0.0010
Epoch 9/40
80/80 [==============================] - 3s 35ms/step - loss: 0.1565 - accuracy: 0.9338 - val_loss: 0.8624 - val_accuracy: 0.7510 - lr: 0.0010
Epoch 10/40
79/80 [============================>.] - ETA: 0s - loss: 0.1181 - accuracy: 0.9535
Epoch 10: ReduceLROnPlateau reducing learning rate to 0.00020000000949949026.
80/80 [==============================] - 4s 48ms/step - loss: 0.1175 - accuracy: 0.9539 - val_loss: 1.1815 - val_accuracy: 0.7365 - lr: 0.0010
Epoch 11/40
80/80 [==============================] - 3s 35ms/step - loss: 0.0521 - accuracy: 0.9875 - val_loss: 0.6328 - val_accuracy: 0.7875 - lr: 2.0000e-04
Epoch 12/40
80/80 [==============================] - 3s 36ms/step - loss: 0.0293 - accuracy: 0.9943 - val_loss: 0.6947 - val_accuracy: 0.7844 - lr: 2.0000e-04
Epoch 13/40
78/80 [============================>.] - ETA: 0s - loss: 0.0197 - accuracy: 0.9970
Epoch 13: ReduceLROnPlateau reducing learning rate to 4.0000001899898055e-05.
80/80 [==============================] - 3s 35ms/step - loss: 0.0194 - accuracy: 0.9971 - val_loss: 0.6830 - val_accuracy: 0.7865 - lr: 2.0000e-04
Epoch 14/40
80/80 [==============================] - 4s 50ms/step - loss: 0.0162 - accuracy: 0.9971 - val_loss: 0.7348 - val_accuracy: 0.7792 - lr: 4.0000e-05
Epoch 15/40
79/80 [============================>.] - ETA: 0s - loss: 0.0120 - accuracy: 0.9994
Epoch 15: ReduceLROnPlateau reducing learning rate to 8.000000525498762e-06.
80/80 [==============================] - 3s 40ms/step - loss: 0.0120 - accuracy: 0.9994 - val_loss: 0.7224 - val_accuracy: 0.7823 - lr: 4.0000e-05
Epoch 16/40
80/80 [==============================] - 3s 38ms/step - loss: 0.0129 - accuracy: 0.9990 - val_loss: 0.7230 - val_accuracy: 0.7823 - lr: 8.0000e-06
Epoch 17/40
79/80 [============================>.] - ETA: 0s - loss: 0.0130 - accuracy: 0.9980
Epoch 17: ReduceLROnPlateau reducing learning rate to 1.6000001778593287e-06.
80/80 [==============================] - 3s 41ms/step - loss: 0.0130 - accuracy: 0.9980 - val_loss: 0.7172 - val_accuracy: 0.7854 - lr: 8.0000e-06
Epoch 18/40
80/80 [==============================] - 3s 37ms/step - loss: 0.0118 - accuracy: 0.9992 - val_loss: 0.7182 - val_accuracy: 0.7854 - lr: 1.6000e-06
Epoch 19/40
```

```
Epoch 19/40
78/80 [============================>.] - ETA: 0s - loss: 0.0111 - accuracy: 0.9990
Epoch 19: ReduceLROnPlateau reducing learning rate to 3.200000264769187e-07.
80/80 [==============================] - 3s 36ms/step - loss: 0.0111 - accuracy: 0.9990 - val_loss: 0.7200 - val_accuracy: 0.7865 - lr: 1.6000e-06
Epoch 20/40
80/80 [==============================] - 3s 34ms/step - loss: 0.0119 - accuracy: 0.9994 - val_loss: 0.7196 - val_accuracy: 0.7865 - lr: 3.2000e-07
Epoch 21/40
79/80 [============================>.] - ETA: 0s - loss: 0.0111 - accuracy: 0.9992
Epoch 21: ReduceLROnPlateau reducing learning rate to 6.400000529538374e-08.
80/80 [==============================] - 4s 47ms/step - loss: 0.0110 - accuracy: 0.9992 - val_loss: 0.7196 - val_accuracy: 0.7865 - lr: 3.2000e-07
Epoch 22/40
80/80 [==============================] - 3s 36ms/step - loss: 0.0121 - accuracy: 0.9986 - val_loss: 0.7203 - val_accuracy: 0.7865 - lr: 6.4000e-08
Epoch 23/40
79/80 [============================>.] - ETA: 0s - loss: 0.0118 - accuracy: 0.9988
Epoch 23: ReduceLROnPlateau reducing learning rate to 1.2800001059076749e-08.
80/80 [==============================] - 3s 38ms/step - loss: 0.0118 - accuracy: 0.9988 - val_loss: 0.7206 - val_accuracy: 0.7865 - lr: 6.4000e-08
Epoch 24/40
80/80 [==============================] - 4s 47ms/step - loss: 0.0110 - accuracy: 0.9992 - val_loss: 0.7203 - val_accuracy: 0.7865 - lr: 1.2800e-08
Epoch 25/40
78/80 [============================>.] - ETA: 0s - loss: 0.0119 - accuracy: 0.9990
Epoch 25: ReduceLROnPlateau reducing learning rate to 2.5600002118153498e-09.
80/80 [==============================] - 3s 36ms/step - loss: 0.0118 - accuracy: 0.9990 - val_loss: 0.7202 - val_accuracy: 0.7865 - lr: 1.2800e-08
Epoch 26/40
80/80 [==============================] - 3s 35ms/step - loss: 0.0106 - accuracy: 0.9996 - val_loss: 0.7201 - val_accuracy: 0.7865 - lr: 2.5600e-09
Epoch 27/40
79/80 [============================>.] - ETA: 0s - loss: 0.0109 - accuracy: 0.9996
Epoch 27: ReduceLROnPlateau reducing learning rate to 5.1200004236307e-10.
80/80 [==============================] - 4s 49ms/step - loss: 0.0109 - accuracy: 0.9996 - val_loss: 0.7197 - val_accuracy: 0.7865 - lr: 2.5600e-09
Epoch 28/40
80/80 [==============================] - 3s 35ms/step - loss: 0.0117 - accuracy: 0.9992 - val_loss: 0.7196 - val_accuracy: 0.7865 - lr: 5.1200e-10
Epoch 29/40
79/80 [============================>.] - ETA: 0s - loss: 0.0108 - accuracy: 0.9990
Epoch 29: ReduceLROnPlateau reducing learning rate to 1.0240001069306004e-10.
80/80 [==============================] - 3s 35ms/step - loss: 0.0109 - accuracy: 0.9990 - val_loss: 0.7202 - val_accuracy: 0.7865 - lr: 5.1200e-10
Epoch 30/40
80/80 [==============================] - 3s 37ms/step - loss: 0.0115 - accuracy: 0.9988 - val_loss: 0.7205 - val_accuracy: 0.7865 - lr: 1.0240e-10
Epoch 31/40
```

```
Epoch 31: ReduceLROnPlateau reducing learning rate to 2.0480002416167767e-11.
80/80 [==============================] - 4s 46ms/step - loss: 0.0113 - accuracy: 0.9994 - val_loss: 0.7208 - val_accuracy: 0.7865 - lr: 1.0240e-10
Epoch 32/40
80/80 [==============================] - 4s 45ms/step - loss: 0.0110 - accuracy: 0.9994 - val_loss: 0.7200 - val_accuracy: 0.7865 - lr: 2.0480e-11
Epoch 33/40
79/80 [============================>.] - ETA: 0s - loss: 0.0106 - accuracy: 0.9992
Epoch 33: ReduceLROnPlateau reducing learning rate to 4.096000622011431e-12.
80/80 [==============================] - 3s 36ms/step - loss: 0.0106 - accuracy: 0.9992 - val_loss: 0.7205 - val_accuracy: 0.7865 - lr: 2.0480e-11
Epoch 34/40
80/80 [==============================] - 4s 51ms/step - loss: 0.0119 - accuracy: 0.9988 - val_loss: 0.7201 - val_accuracy: 0.7865 - lr: 4.0960e-12
Epoch 35/40
79/80 [============================>.] - ETA: 0s - loss: 0.0112 - accuracy: 0.9990
Epoch 35: ReduceLROnPlateau reducing learning rate to 8.192000897078167e-13.
80/80 [==============================] - 3s 35ms/step - loss: 0.0112 - accuracy: 0.9990 - val_loss: 0.7199 - val_accuracy: 0.7865 - lr: 4.0960e-12
Epoch 36/40
80/80 [==============================] - 3s 35ms/step - loss: 0.0116 - accuracy: 0.9988 - val_loss: 0.7199 - val_accuracy: 0.7865 - lr: 8.1920e-13
Epoch 37/40
78/80 [============================>.] - ETA: 0s - loss: 0.0118 - accuracy: 0.9992
Epoch 37: ReduceLROnPlateau reducing learning rate to 1.6384001360475466e-13.
80/80 [==============================] - 4s 54ms/step - loss: 0.0121 - accuracy: 0.9992 - val_loss: 0.7196 - val_accuracy: 0.7865 - lr: 8.1920e-13
Epoch 38/40
80/80 [==============================] - 3s 36ms/step - loss: 0.0109 - accuracy: 0.9992 - val_loss: 0.7201 - val_accuracy: 0.7865 - lr: 1.6384e-13
Epoch 39/40
79/80 [============================>.] - ETA: 0s - loss: 0.0114 - accuracy: 0.9988
Epoch 39: ReduceLROnPlateau reducing learning rate to 3.2768002178849846e-14.
80/80 [==============================] - 3s 38ms/step - loss: 0.0114 - accuracy: 0.9988 - val_loss: 0.7201 - val_accuracy: 0.7865 - lr: 1.6384e-13
Epoch 40/40
80/80 [==============================] - 4s 49ms/step - loss: 0.0106 - accuracy: 0.9990 - val_loss: 0.7204 - val_accuracy: 0.7865 - lr: 3.2768e-14
15/15 [==============================] - 0s 20ms/step - loss: 0.7204 - accuracy: 0.7865
```



Trong 40 epochs mô hình đạt độ mất mát 0.7204 (val_loss) và độ chính xác là 0.7865 (val_accuracy).

7. Discussion and insights

Qua project cuối kỳ chúng ta có thể tiếp cận và hiểu biết nhiều hơn về mạng Nơron tích chập để giải quyết các bài toán về phân biệt hình ảnh, thị giác máy tính.

Có một vấn đề đặt ra ở đây là tại sao lại dùng mạng Nơron tích chập cho bài toán nhận diện hình ảnh? Có thể do các nguyên nhân sau:

- CNN có khả năng xử lý tính cục bộ và không gian của dữ liệu hình ảnh một cách hiệu quả. Bằng cách sử dụng các lớp tích chập, CNN có thể nhận biết các đặc trưng cụ thể ở bất kỳ vị trí nào trong hình ảnh, mà không cần biết trước vị trí cụ thể của chúng.
- Tính chia sẻ trọng số của CNN giúp giảm số lượng tham số cần học, làm giảm nguy cơ overfitting và tăng khả năng tổng quát hóa của mô hình.
- Hiệu suất tính toán cao của CNNs làm cho chúng phù hợp với các ứng dụng thời gian thực và các thiết bị có tài nguyên hạn chế.

Tiếp theo ta cần xem xét rằng tại sao với số lượng tham số ít mà mạng Nơ-ron tích chập lại có thể giải quyết được các bài toán phức tạp? Ta có thể hiểu thông qua các yếu tố sau:

- CNN sử dụng các lớp tích chập để chia sẻ các trọng số trong một cửa sổ trượt qua toàn bộ hình ảnh. Điều này giúp giảm số lượng tham số cần học, đặc biệt là khi xử lý ảnh, nơi các đặc trưng có thể được phát hiện ở bất kỳ vị trí nào trong ảnh.
- Mặc dù số lượng tham số có thể ít, nhưng với các lớp phi tuyến tính như lớp kích hoạt ReLU giúp mô hình học được các biểu diễn phức tạp của dữ liệu.
- CNNs hiểu rõ sự quan hệ không gian giữa các pixel trong ảnh. Thay vì xem các pixel một cách độc lập, CNNs xử lý các phần của ảnh theo cụm thông qua các bộ lọc tích chập. Điều này giúp mô hình nhận diện các đặc trưng cục bộ trong một cấu trúc không gian.

Sau khi thực hiện bài tập cuối kỳ nhóm chúng em còn hạn chế do chưa đạt yêu cầu về Accuracy trên 85%. Cách khắc phục vấn đề này có thể hướng tới việc như:

- Sử dụng các mô hình được đào tạo trước (pre-trained models) như VGG, ResNet, hoặc Inception và thay đổi các phần của mô hình
- Điều chỉnh các siêu tham số như tỷ lệ học, số lượng epochs, kích thước batch
- Thử nghiệm với các mô hình lớn hơn hoặc thêm các lớp để xem liệu một mô hình phức tạp hơn có thể học được biểu diễn tốt hơn cho dữ liệu hay không