Performance Comparison with Pre-trained Model

Summary of Convolutional Neural Network

1. What is a Convolutional Neural Network (CNN) and why to use it?

Summary of Convolutional Neural Network

1. What is a Convolutional Neural Network (CNN) and why to use it?
2. Understanding the components of CNN

Summary of Convolutional Neural Network

1. What is a Convolutional Neural Network (CNN) and why to use it?
2. Understanding the components of CNN
3. CNN Architecture and its working

Summary of Convolutional Neural Network

1. What is a Convolutional Neural Network (CNN) and why to use it?
2. Understanding the components of CNN
3. CNN Architecture and its working
4. Solving Emergency Classification problem using CNN

Summary of Convolutional Neural Network

1. What is a Convolutional Neural Network (CNN) and why to use it?
2. Understanding the components of CNN
3. CNN Architecture and its working
4. Solving Emergency Classification problem using CNN
5. HyperParameter tuning for CNN

Comparison with Pre-trained model

```
Epoch 102/200
1646/1646 [==============================] - 16s 10ms/step - loss: 0.1616 - acc: 0.9338 - val_loss: 0.4896 - val_
acc: 0.8385
Epoch 103/200
1646/1646 [==============================] - 16s 10ms/step - loss: 0.2296 - acc: 0.9131 - val_loss: 0.4143 - val_
acc: 0.8428
Epoch 104/200
1646/1646 [==============================] - 16s 10ms/step - loss: 0.2243 - acc: 0.9058 - val_loss: 0.4374 - val_
acc: 0.8343
Epoch 105/200
1646/1646 [==============================] - 16s 10ms/step - loss: 0.1461 - acc: 0.9435 - val_loss: 0.4834 - val_
acc: 0.8428
Epoch 106/200
1646/1646 [==============================] - 16s 10ms/step - loss: 0.1208 - acc: 0.9532 - val_loss: 0.4772 - val_
acc: 0.8569
Epoch 107/200
1646/1646 [==============================] - 16s 10ms/step - loss: 0.1454 - acc: 0.9417 - val_loss: 0.4948 - val_
acc: 0.8470
Epoch 108/200
1646/1646 [==============================] - 16s 10ms/step - loss: 0.1605 - acc: 0.9405 - val_loss: 0.4064 - val_
acc: 0.8527
```

Result of same Architecture as pre-trained VGG model

Comparison with Pre-trained model

```
hist = model.fit(X_train, Y_train, epochs=3, batch_size=128, validation_data=(X_valid, Y_valid))
```

```
Train on 1646 samples, validate on 706 samples
Epoch 1/3
1646/1646 [==============================] - 17s 10ms/step - loss: 0.3211 - acc: 0.8639 - val_loss: 0.2422 - val_ac
c: 0.8909
Epoch 2/3
1646/1646 [==============================] - 16s 10ms/step - loss: 0.1386 - acc: 0.9526 - val_loss: 0.2168 - val_ac
c: 0.9122
Epoch 3/3
1646/1646 [==============================] - 17s 10ms/step - loss: 0.0906 - acc: 0.9751 - val_loss: 0.2018 - val_ac
c: 0.9207
```

Pre-trained VGG model result

Thank you!