Artificial Neural Networks and Deep Learning 2021

Homework 1 - Image Classification

Prof. Matteo Matteucci

Prof. Giacomo Boracchi

Group members:

Hamid Salehi 10759997

Hossein Mohammadalizadeh 10717597

Anastasia Cotov 10767910

Academic Year 2021/2022

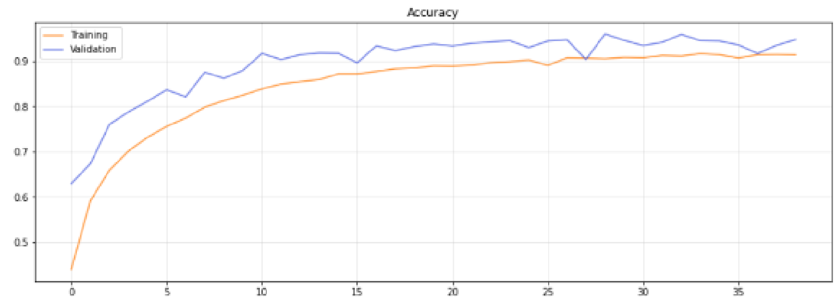
Introduction

The aim of this project is to train the given images with the best possible model and predict the test images with high mean accuracy. The images are divided into fourteen categories. Here, we develop three models in order to cope with the problem: 1. OurCNN 2. Fine-tuning with vgg16 3. Fine-tuning with ResNet50.

The first step in tackling this problem is to actually know that our model is overfitting. After identifying the problem, we can prevent it from happening by applying Dropout technique. Also, one of the most common techniques, which is used in all our models, is Data Augmentation that can help us if our dataset is too small, or we have overfitting.

1.OurCNN

The first model that we have submitted had three convolutional blocks (including convolutional layer and maxpooling layer) followed by one dense layer as classifier. So we decided to increase convolutional blocks to four based on the receptive field concept. In CNN each output only depends on a specific region in the input. This region in the input is the receptive field for that output and the deeper CNN the wider receptive field we have that could help us to get the higher mean accuracy (**58%**).



Here we can see that we prevent overfitting and both training and validation have good accuracy.

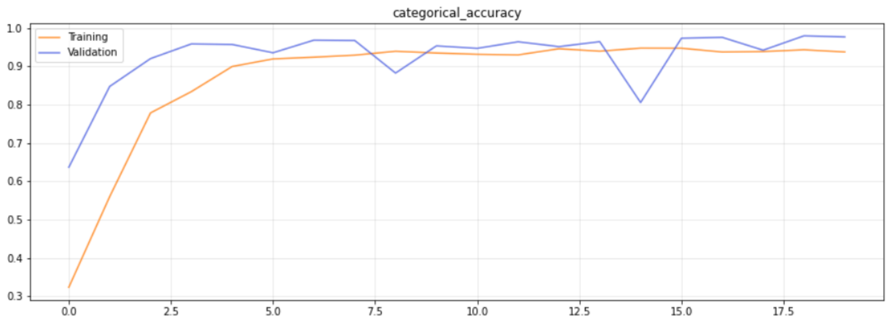
Classification Report

	precision	recall	f1-score	support
Apple	0.92	0.98	0.95	198
Blueberry	0.97	0.96	0.96	94
Cherry	0.98	0.93	0.96	117
Corn	1.00	0.96	0.98	242
Grape	0.99	0.96	0.98	292
Orange	0.99	0.97	0.98	350
Peach	0.99	0.91	0.95	196
Pepper	0.95	0.86	0.90	153
Potato	0.93	0.82	0.87	144
Raspberry	0.94	0.92	0.93	53
Soybean	0.96	0.96	0.96	324
Squash	0.96	0.98	0.97	115
Strawberry	0.88	0.99	0.93	135
Tomato	0.96	0.99	0.97	1139
accuracy			0.96	3552
macro avg	0.96	0.94	0.95	3552
weighted avg	0.96	0.96	0.96	3552

In this section, we use other metrics like f1-score, recall and precision for each metrics because only accuracy cannot satisfy us for this kind of multiclassification problem

2.Fine-Tuning with vgg16

In order to get higher accuracy, we came up with using pretrained models like vgg16 and deployed the Fine-tuning technique. In this part we kept the first four blocks of vgg16 untouched as they can play an edge-detector role and trained the fifth block and classifier. This method helped us to get **87%** accuracy in the test set. Also, we decided to cope with our imbalanced data because we saw that some of the classes had high accuracy and some of them were worse. Therefore, among some techniques like oversampling, downsampling and class weights for dealing with this problem, we chose class weights. In this approach, we can modify the current training algorithm to take into account the bad distribution of the classes. This can be achieved by giving different weights to both the majority and minority classes. The whole purpose is to penalize the misclassification by reducing weight for the majority class.



Also in this model we changed the metric into categorical_accuracy in order to save best model in ModelCheckPoint or Earlystopping callbacks.

Classification Report

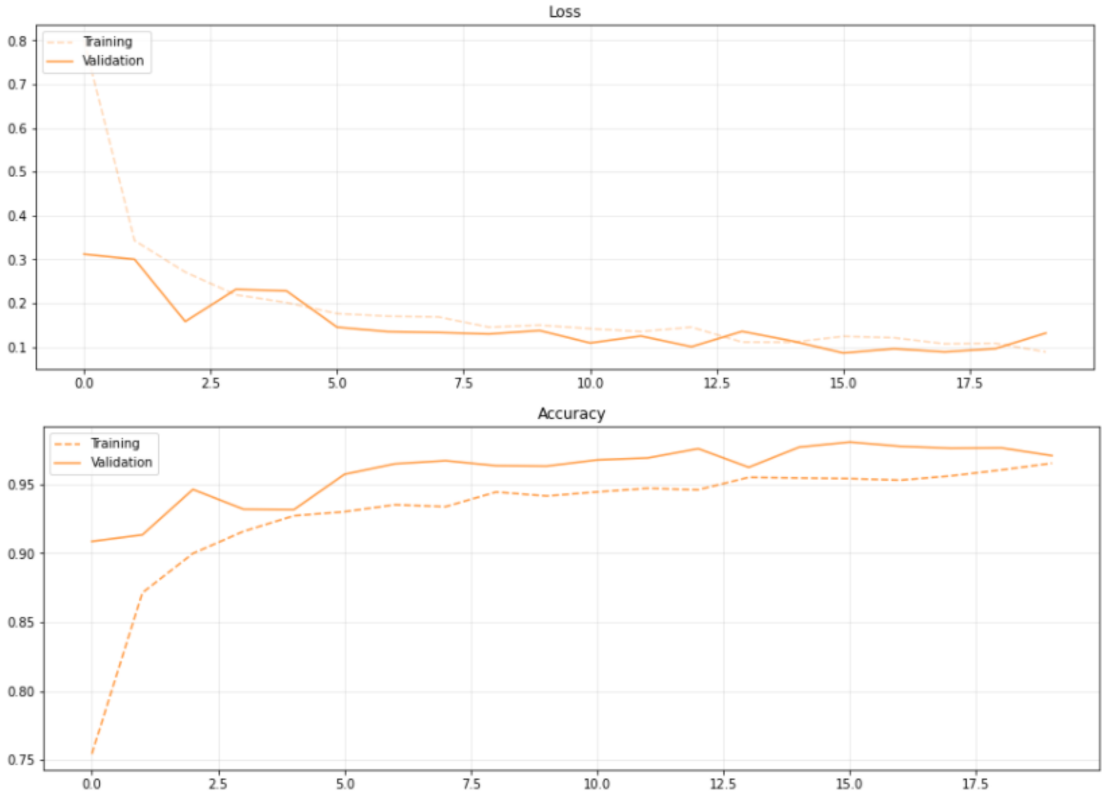
	precision	recall	f1-score	support
Apple	0.97	0.98	0.98	198
Blueberry	0.98	1.00	0.99	94
Cherry	0.98	0.99	0.99	117
Corn	1.00	1.00	1.00	242
Grape	0.99	0.97	0.98	292
Orange	0.99	0.97	0.98	350
Peach	0.99	0.99	0.99	196
Pepper	0.92	1.00	0.96	153
Potato	0.79	0.99	0.88	144
Raspberry	0.98	1.00	0.99	53
Soybean	0.99	0.99	0.99	324
Squash	1.00	0.97	0.99	115
Strawberry	0.95	1.00	0.97	135
Tomato	0.99	0.95	0.97	1139
accuracy			0.98	3552
macro avg	0.97	0.99	0.98	3552
weighted avg	0.98	0.98	0.98	3552

As we can see, all the metrics show the improvement compared to the last model

3.FineTuning with ResNet50

In the third try, we chose to train the model using ResNet50, because it implements skip connection technique solving two problems: 1. Vanishing gradient 2. learning identity function which ensures that the higher layer will perform at least as good as the lower layer

With Fine Tuning we are not limited to retraining only the classifier stage (i.e., the fully connected layers), but we retrain also the feature extraction stage, i.e. the convolutional and pooling layers. The Model we used has 189 layers, we freeze 100 first layers and retrain the others.



It appears that validation loss may be rising toward the end of training, indicating that the model may be overfitting. This model reached 87% on test set on codalab.

Conclusion

Using pre-trained models like VGG16 and ResNet50 we have seen better performance from the 1^{st} epoch of training. This approach is effective because the models were trained on a large corpus of photographs, and we use them as Feature Extractors for our model. Although, in Fine-Tunning we retrained also few last convolutional layers of the models. Analyzing accuracy plots we see major improvements:

- **Higher start**. The initial skill (before refining the model) on the vgg16/ResNet model is higher, e.g. ResNet accuracy starts from 0.75 (see accuracy plot above).
- **Higher slope**. The rate of improvement of skill during training of the vgg16/ResNet is steeper than the 1^{st} model.
- **Higher asymptote**. The converged skill of the vgg16/ResNet trained model (between 0.9 and 1) is better than 1^{st} model (between 0.8 and 0.95).