SAPIENZA
UNIVERSITÀ DI ROMA

Homework 2

Machine Learning

Students:
Flavio Maiorana
2051396

Academic Year 2023/2024

1 Introduction

First of all, it could be useful to gain some insight one how the dataaset is made. The available data is already split into training and testing set. We will consider the test split only in the evaluation phase (to prevent data leakage).

```
[Dataset for Training]
    N Examples: 6369
    N Classes: 5
    Classes: [0 1 2 3 4]
    - Class 0: 1000 (15.701051970482022)
    - Class 1: 1500 (23.551577955723033)
    - Class 2: 1500 (23.551577955723033)
    - Class 3: 2000 (31.402103940964043)
    - Class 4: 369 (5.7936881771078665)
```

Some comments: the dataset is highly imbalanced. That needs to be addressed accordingly. The main two problems to address are:

- Model architecture
- Training process
  - Class imbalance mitigation
  - Optimization choice
  - Performance metrics choice

2 Model Architecture

The Model is a CNN, since we have to classify images. The first and foremost choice regarding cnn model architecture is surely how deep it has to be and how big the kernel need to be. Since the images are not that big, a reasonably small cnn would be enough, since a too deep one would either overfit on training data, or waste computational resources without gaining any improved performance. Although, also a too small model would not be able to adequately catch the features of the image. Also, kernels size is very important. A too big kernel would compromise computational efficiency, but the size has to be adapted to the tiype of features the convolution needs to to capture. To corroborate these concepts we will compare some cnns with a different number of convolutional layers. Each cnn has 2 fully connected layers at the end. Each list entry indicates the dimension for the convolutional layer (from 1 to N). The kernel size 0 indicates the number of channels of the input image. This training results are all captured after 40 epochs.

- N = 3

```
'channels': [3,8,16,32,64],
'kernels': [5,5,5,5],
'strides': [1,1,1,5],
'pool_kernels': [2,2,2,2],
'pool_strides': [2,2,2,2]

precision    recall  f1-score   accuracy
  0.702     0.623     0.644      0.623

Total reward: 903.86
Frames: 894
Training time: 300.85
```

- N = 6

```
'channels': [3,10,16,32,64,64,64,64,64],
'kernels': [5,5,5,3,3,3,64,64],
'strides': [1,1,1,1,1,1,1,1],
'pool_kernels': [2,2,2,2,2,2],
'pool_strides': [2,1,2,1,2,2],

precision  recall  f1-score   accuracy
  0.699    0.638     0.653      0.638

Tile points: 874.69
Training time:
```

2.1 Batch normalization layers

2.2 Dropout layer

2.3 Ensemble

3 Training process

There are some important issues to address, which are to some extent independent from the model we use to classify. First of all, to mitigate class imbalance we could apply:

- Data augmentation (useful in any case to increase generalization)
- Different sampling techniques
  - Upsampling/Downsampling
  - Weighted resampling

3.1 Sampling techniques

In order to mitigate class imbalance, one could try to rebalance the dataset by using weighted resampling techniques.

3.2 Data augmentation

This technique is widely used with images. It exploits the invariance property of images to enhance generalization properties of neural networks. In our case it is particularly useful, because it allows us to "reuse" the same sample multiple times and make it look new to the network by augmenting its features. The transformation I used are these three from the torchvision package, plus a random color jitter implemented separately. I also tried RandomHorizontalFlipping, but I noticed actually worse conditions, maybe because the images are sensitive to the horizontal orientation (for turning left and right), so randomly flipping them would mean introducing noise to the labels.

```
transforms.RandomInvert(),
transforms.RandomErasing(),
transforms.RandomRotation(degrees=5),
```

3.3 Weighted random resampling

To rebalance classes, we could sample differently the training data, maybe according to some weights.

4 Final Considerations

In the end, after different trials I was able to achieve the completion of a full lap through manual tuning of class weights