CNN of Animals DataSet

Davide Saitta DAVIDESAITTA39@GMAIL.COM
Fabio Amoroso FABIOAMOROSOFA98@GMAIL.COM

1. Model Description

In order to accomplish the classification task assigned, we have developed a model made by a convolutional part, whose aim is to extract relevant features from the data, and a fully connected one to classify the images into 8 classes. In the first part, starting from RGB images (three channels), there are 6 convolutional layers. The first convolutional layer provides 64 feature maps after applying 3x3 kernels, with padding and stride equal to 1, followed by a ReLU activation function and a batch normalization. Each of the following convolutional layers double the number of feature maps received as input by the previous layer, applying 3x3 kernels with the same padding and stride. Differently by the first convolutional layer, the others involve also a Max Pooling phase that allow us to reduce the dimensionality of the feature maps using 2x2 kernels.

Figure 1: A simple representation of the first two Convolutional Layers.

In the fully connected part we exploit the power of dense layers for classification. We have reshaped the 4D data (batch size, depth, width, height) obtained by the convolutional part of our model in order to get the 2D data for the fully connected layers. The input received by the first dense layer is 8192 neurons, that we reduced to 1024 for the classifier. Finally, the model will be able to accomplish the classification task. Also in the first dense layer a ReLU activation function and a batch normalization phase are applied.

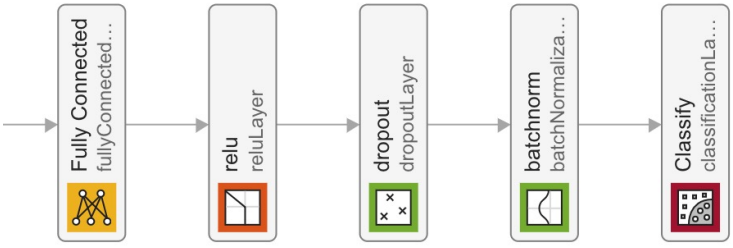

Figure 2: A simple representation of the Fully Connected Layers.

The problem that we had to face is the overfitting of the model to the real data, that lead to an earlier divergence between training and validation/test accuracy during the training. The solutions that we applied to mitigate this problem are the batch normalization, that allowed us also to speed up the training, and the dropout in the first dense layer. Another technique used for reducing the overfitting is the data augmentation, explained in the next section.

2. Dataset

The data-set we used has been taken from the Kaggle website. The name of the data-set is Animals-10, and it consists of 26179 medium quality RGB images divided in 10 classes.

All images were collected by "Google images" and have been checked by human. In the data-set there is the presence of some erroneous data to simulate real conditions.

The classes of the data-set are the following: "cane" (4863), "cavallo" (2623), "elefante" (1446), "farfalla" (2112), "gallina" (3098), "gatto" (1668), "mucca" (1866), "pecora" (1820), "ragno" (4821) and "scoiattolo" (1862).

As is evident from above, the data-set is unbalanced, having as minority class "elefante" which contains 1446 images and majority class "cane" with 4863 images. In order to handle this problem, we decided to apply under-sampling so that the model can learn equally from all the classes; we have also removed the two minority classes ("elefante" and "gatto") to increase the number of images for each class before the under-sampling procedure. The new resized data-set has 8 classes and is made up of 14560 images, 1820 for each class and divided into train, validation and test sets with the following proportion: 80% train, 10% validation, 10% test.

To uniform data shape and reduce memory requirements, each input has been resized to a 64x64 pixels each and, to increase the variability of the data, we also performed data augmentation; specifically we rotated the images randomly by an amount ranging from 0 to 20 degrees. Below, a sample of images of the data-set.

Figure 3: A sample of images of the class cavallo.

As it appears evident in figure 3 the dataset presents images with high variability. This implies that the dataset doesn't need the application of many data augmentation techniques.

3. Training procedure

The training of the models have been performed with the aid of the GPU accelerators provided by CUDA. The batch size chosen for the loaders is 32 images. The loss has been computed with the cross entropy criterion, that calculate a separate loss for each batch and sum the results.

For the backpropagation phase, we chose the Stochastic Gradient Descent as optimizer. Although we have used some procedures to reduce the overfitting, the issues were already present. So, for our main model, we chose the value of 0.005 for the learning rate. Thanks to this choice, the divergence between training and validation/test accuracy have been delayed and we get better results. The main model has been trained for 30 epochs.

We have also performed an ablation study, training other 5 models, in which we have incrementally added convolutional layers starting from the first one. The fully connected part is always the same, but the input it receives from the convolutional part is different for each model. The results can be seen in the Table 1 of the following section.

4. Experimental Results

Several experiments were conducted to test the effectiveness of the proposed Network architecture. The chosen network was trained as described in the previous section.

Below, in Table 1, we show test results of the ablation studies (i.e., different variants of the proposed architecture when adding or removing layers).

Model	Test Accuracy
1 Convolutional Layer	57.07%
2 Convolutional Layer	64.81%
3 Convolutional Layer	69.97%
4 Convolutional Layer	74.05%
5 Convolutional Layer	75.88%
6 Convolutional Layer	76.29%

Table 1: Test performance of the models when adding incremental layers.

As we can see from above, the architecture that give us the best result in terms of test accuracy is the one with 6 convolutional layers, that obtain an accuracy of 76.29%.

Let's have a look to the confusion matrix obtained with the chosen architecture.

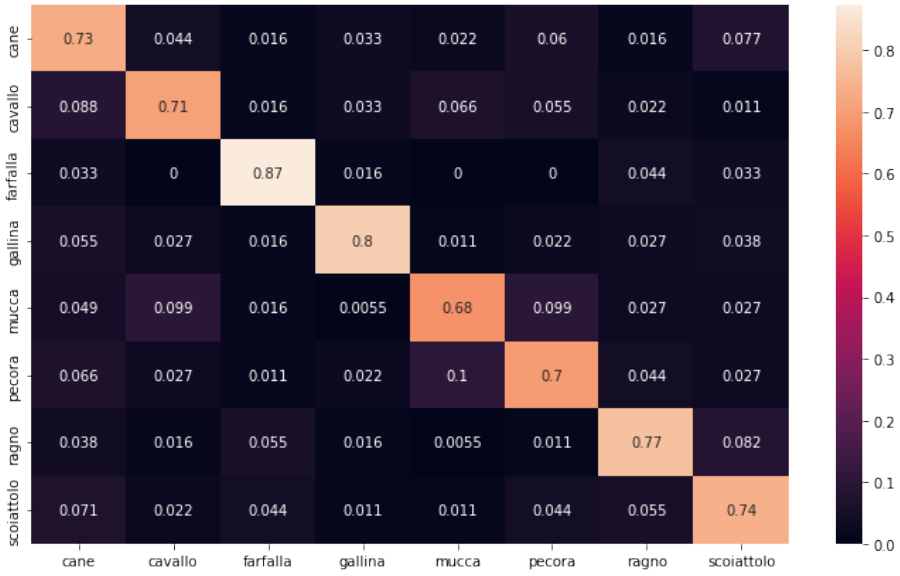

Figure 4: Confusion Matrix of the results obtained with the main model.

As we can see from the confusion matrix, all the classes have values of accuracy greater than the 70%, except for "mucca". The most frequent misclassifications involve "mucca", "pecora" and "cavallo". The similarities between these three animals show how a model with more convolutional layers would be more appropriate for identifying the high level features that distinguish each of these species from the others. With the model used, the accuracy on these three classes is significantly lower than the accuracy of the best classified classes. The classes which present the best accuracy values are "farfalla" and "gallina" with, respectively, 87% and 80%.