Statistical methods for machine learning project

Giuseppe Carugno

July 26, 2022

Contents

0 Disclaimer

I/We declare that this material, which I/We now submit for assessment, is entirely my/our own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of my/our work. I/We understand that plagiarism, collusion, and copying are grave and serious offences in the university and accept the penalties that would be imposed should I engage in plagiarism, collusion or copying. This assignment, or any part of it, has not been previously submitted by me/us or any other person for assessment on this or any other course of study.

1 Introduction

The aim of the project is developing a neural network for binary classification of cats and dogs images. The dataset containing the images to classify is provided.

1.1 Requirements

Images must be transformed from JPG to RGB or grayscale pixel values and scaled down. Risk estimates must be computed using 5-fold cross validation for training and zero-one loss for testing.

1.2 Tools

The coding language is Python and the machine learning toolbox used is TensorFlow 2. The project is developed on Colab to exploit the availability of GPU and TPU.

2 Preprocessing

Before developing a neural network model some preprocessing techniques are applied to the dataset. The aim is to clean and standardize the data before passing it to the model. A clever preprocessing improves training and yields better results.

2.1 Dataset download

The dataset is downloaded from the link to a directory and unzipped. A visual check over the dataset shows that it's composed by a directory containing two subdirectories. The first subdirectory contains cats images, meanwhile the second subdirectory contains dogs images. Both subdirectories contain 12.500 images meaning that the distribution of samples over the two classes is uniform and the dataset is balanced.

2.2 Dataset cleaning

A full pass is made over the dataset to check the images. If they're not JPG, can't be read, scaled or are corrupted, they're removed from the dataset. After this pass the number of images is lowered from 25.000 to 24.734, with 12.385 cats images and 12.349 dogs images. After removing not valid images, since the number of samples in each class is still similar, the dataset remains balanced. Then no balancing technique is required.

2.3 Generation of samples and labels

Using the TensorFlow pipeline a dataset of samples, that can be used by neural network models, is generated from the main directory. The labels are assigned to each image by inferring them from the structure of the directory. Since the pipeline can alter the dataset that is fed to the neural network, different configurations are going to be tested to improve the final model. As starting point labels are chosen to be integers, in this case 0 and 1 since there are only two classes, images are converted to RGB and reshaped to a default size of 256*256. The dataset is shuffled and batches of size 32 are generated.

2.4 Conversion to grayscale and normalization

Using a transformation all images are converted to grayscale and pixel values are normalized in the range of 0 to 1.

2.5 Scaling down

Lastly images are scaled down to a lower size until cats and dogs are still recognizable inside the pictures. After this final step all the requirements regarding the preprocessing of the images are satisfied.

3 Training, validation and test sets

Once the dataset is ready it is split into two subsets using the ratio of 80% to 20%. The larger subset is used to generate training and test sets to use during 5-fold cross validation, meanwhile the smaller subset is the validation set.

3.1 Training and test sets

To execute 5-fold cross validation the subset that has 80% of the total samples is split into 5 folds with an equal number of samples. Each fold is a single test set. The training set of each test set is the union of the remaining 4 folds that are not the fold of the test set. A total of 5 training sets and their corresponding 5 test sets are built. Each couple of sets trains and tests a different model and then the results are averaged.

3.2 Validation set

The validation set corresponds to 20% of the total dataset and it is always the same. It is used during each epoch of each training of the 5-fold cross validation technique.

4 Neural network

This section explains the rationale and the steps behind the development of the model.

4.1 Design choice

The assessment of the task to solve leads to the choice of the neural network architecture to use. Since the task is binary classification of images the most suitable architecture is a convolutional neural network (CNN). Convolutional neural networks architecture can be divided into two sequential smaller neural networks. The first one handles the extraction of features from the images by moving neurons of the input layers over the input image. The second one, built on top of the first one, is a feed forward neural network used as image classifier. Since in binary classification there are only two possible classes, the output layer has only 1 neuron, with a sigmoid activation function to decide whether the image is either a dog or a cat.

4.2 Convolutional neural network

To find the best model various layers with different configurations parameters are trained and tested. Experimental results are collected and then the best convolutional neural network is selected. Before deployment the final model is fine tuned once again by trial and error to find its best configuration parameters for its layers.

4.2.1 Feature extraction

The first part of the convolutional neural network, in charge of extracting features from the images, is initially tested by alternating 2-dimensional convolutional layers and 2-dimensional max pooling layers. After each convolutional layer there is always a max pooling layer. At the end of the feature extraction network a flattening layer flattens the results to one dimension to pass them to the classifier. Initially only one pooling layer is used with a different number of filters. As starting point the number of filters tried is 32, 64 and 128 with maximum kernel size of 3*3. Scaling down images to a size of 64*64 allows to check whether using a number of filters that is half, same or double the image dimensions yields promising results. The exact classifier structure at the moment is irrelevant. It's a dense layer with 128 neurons and the output layer. Training is done over 10 epochs.

1 Conv2D layer	Training loss	Validation loss	Testing loss
32 filters	0.03	2.32	2.07
64 filters	0.03	2.67	2.69
128 filters	0.08	2.02	2.00

1 Conv2D layer	Training accuracy	Validation accuracy	Testing accuracy
32 filters	99.03%	69.07%	70.74%
64 filters	99.34%	68.32%	68.94%
128 filters	97.11%	69.83%	70.15%

All the models are overfitting but a lower number of filters in the first layer leads to overall better results. So the model with 32 filters in the first convolutional layer is used. To reduce overfitting two different strategies are tested: applying further preprocessing on the images or using a regularization technique over the network.

Data augmentation: All images are randomly flipped left or right and up or down. Half of them is also rotated by 90 degrees counter clock-wise. This way the model improves generalization capabilities by being forced to extract and learn rotation invariant features.

1 Conv2D layer	Training loss	Validation loss	Testing loss
32 filters	0.56	0.58	0.60

1 Conv2D layer	Training accuracy	Validation accuracy	Testing accuracy
32 filters	71.00%	71.39%	70.00%

Data augmentation improved the results since, even though the model achieved lower accuracy, no overfitting occurred. So the data augmentation procedure is added as new final step of the data preprocessing pipeline.

Dropout regularization: A dropout layer is added after the pooling layer. With dropout each neuron in the corresponding layer may be discarded from training during each epoch according to a certain probability. The dropout technique makes learning more difficult for the model while reducing the risk of overfitting. The model is tested with different dropout probabilities of 25%, 50% and 75%.

1 Conv2D layer	Training loss	Validation loss	Testing loss
25% dropout	0.13	1.48	1.40
50% dropout	0.27	0.71	0.72
75% dropout	0.57	0.60	0.60

1 Conv2D layer	Training accuracy	Validation accuracy	Testing accuracy
25% dropout	95.65%	67.95%	69.24%
50% dropout	89,22%	70.00%	69.78%
75% dropout	71.63%	69.00%	68.76%

The collected data exhibits the benefits of dropout regularization. However a relevant reduction of overfitting happens only when a 75% dropout rate is used. Given the 32 neurons of the convolutional layer only 8 are active and trained during each epoch, which is an extremely small number overall. Since data augmentation provides the same results as a 75% dropout rate, but while training all 32 neurons simultaneously, at the moment only the former is applied. Now it's necessary to raise accuracy values.

Deep neural network: To check for further improvements regarding accuracy new convolutional layers followed by max pooling layers are added. Two new models are evaluated: the first one has a second convolutional layer with twice the filters of the first layer, 64, meanwhile the second one is exactly the previous model but with the addiction of a third convolutional layer with twice the filters of the second convolutional layers, so 128. Filters in the deeper layers are always an increasing number to extract higher level features from the results of the previous one. Models are then trained and tested.

N Conv2D layers	Training loss	Validation loss	Testing loss
1 (32)	0.56	0.58	0.6
2 (32, 64)	0.6	0.59	0.59
3 (32, 64, 128)	0.69	0.69	0.69

N Conv2D layers	Training accuracy	Validation accuracy	Testing accuracy
1 (32)	71.00%	71.39%	70.00%
2 (32, 64)	68.04%	67.87%	68.21%
3 (32, 64, 128)	49.77%	49.90%	49.58%

A deeper model doesn't show improvements over a simpler one but actually all accuracy metrics decreases drastically. A new test is made over the models by adding back dropout regularization to each layer. The aim is to study the benefits of dropout training over deeper networks. The dropout percentage chosen is 25% for each layer. Results are collected.

N Conv2D layers	Training loss	Validation loss	Testing loss
1 (32)	0.57	0.56	0.56
2 (32, 64)	0.56	0.54	0.54
3 (32, 64, 128)	0.61	0.60	0.61

N Conv2D layers	Training accuracy	Validation accuracy	Testing accuracy
1 (32)	71.15%	70.68%	70.13%
2 (32, 64)	71.47%	73.57%	72.18%
3 (32, 64, 128)	67.00%	67.00%	66.21%

On one hand the simpler model yields results similar to the test without dropout regularization, while on the other the two deeper models achieve higher accuracy levels than without dropout. The improvement is more significant when the model is deeper. While the model with two convolutional layers improved accuracy by around 4%, the one with three by a relevant 17%. The three models must be now trained over a larger number of epochs to check whether the deeper ones, with the benefit of longer training, can outperform the simple one. However, before doing so, some trials are run over the classifier, which is doing the actual classification after the convolutional layers. The aim is to find if there is a better number of neurons to use or dropout can further improve accuracy by being applied also to the classifier.

4.2.2 Classification

The second part of the architecture is a feed forward neural network in charge of providing the actual prediction. After extensive tests over the feature extraction network the classification network has its configuration changed. By trying some adjustments it's possible to discover if the last layers are responsible for the low accuracy. The model with just one convolutional layer is tested with different numbers of neurons in the last hidden layer. Instead of 128 neurons 64 and 256 are used and then the results are reported.

Neurons	Training loss	Validation loss	Testing loss
64	0.61	0.60	0.59
128	0.57	0.56	0.56
256	0.60	0.58	0.58

Neurons	Training accuracy	Validation accuracy	Testing accuracy
64	66.18%	68.40%	68.61%
128	71.15%	70.68%	70.13%
256	67.49%	69.85%	70.33%

There aren't significant differences regarding the number of neurons in the last hidden layer so 128 neurons are kept. Now dropout regularization is also applied to the classifier to see if there are improvements. The neural network receives dropout on the classifier's layer. The dropout rates testes are 25%, 50% and 75%.

Dropout rate	Training loss	Validation loss	Testing loss
25%	0.50	0.48	0.47
50%	0.54	0.51	0.50
75%	0.59	0.55	0.55

Dropout	Training accuracy	Validation accuracy	Testing accuracy
25%	75.48%	77.27%	77.87%
50%	72.24%	75.28%	75.97%
75%	68.88%	72.12%	72.33%

A 25% dropout rate applied to the classifier improves accuracy by around 8%. So far applying dropout to each layer yields the best performing models.

4.2.3 Training and testing

Dropout has been proven beneficial both during feature extraction and classification so all the layers receive a 25% dropout rate. Also the deepest model now receives longer training to discover if it's capable of outperforming the simpler ones. Instead of 10 epochs training is performed during 25, 50 and 100 epochs.

Epochs	Training loss	Validation loss	Testing loss
25	0.41	0.35	0.35
50	0.38	0.32	0.33
100	0.31	0.25	0.24

Dropout	Training accuracy	Validation accuracy	Testing accuracy
25	81.20%	84.09%	84.81%
50	82.68%	86.18%	85.80%
100	85.93%	90.46%	90.09%

The model benefits greatly from longer training. With 100 epochs it achieves a 90% accuracy and shows no overfitting. This is the best result achieved so far. The final model configuration is reported below.

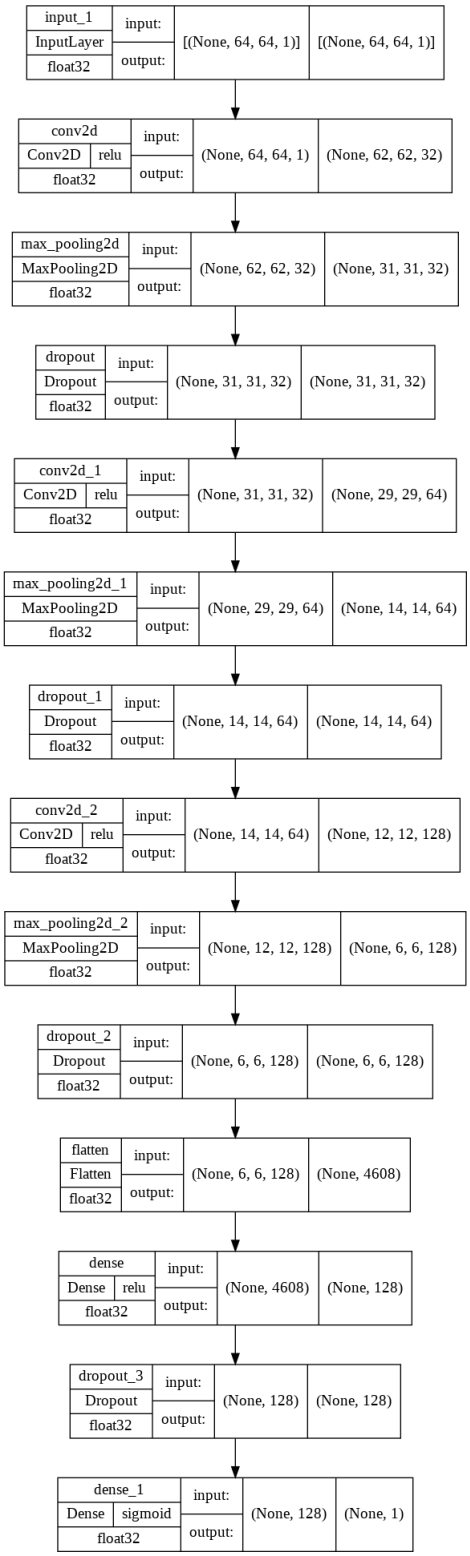

Figure 1: Convolutional neural network architecture

5 Results

The experimental data collected exhibits the behaviour of convolutional neural networks used for binary classification of dogs and cats images. Simpler models with fewer layers tend to overfit. They yields high accuracy values during training but lower accuracy when evaluated over validation and test sets. Data augmentation and regularization techniques are effective improvements to avoid overfitting, at the cost of lowering training accuracy. Making the model deeper is not enough to achieve better results since accuracy gets even lower than a simpler model. However this statement loses validity when a regularization technique is applied, like dropout regularization. It seems that, while dropout is still beneficial for simple models, the benefits of similar techniques are amplified even more for complex models. Another trade-off regarding deeper models shows up during training. Over a low number of epochs simple models have higher accuracy values than deeper ones. This fact can be misleading because their accuracy during training starts plateauing early. Over a larger number of epochs the deepest model accuracy keeps increasing for longer time, until it starts outperforming the simpler models. At the end it also becomes the only model to reach an accuracy value above 90% regarding the test set. So, once a good yet complex model that doesn't overfit has been found, the trade-off to handle is between accuracy and training time.