Programming Exercise 6: Image Classification of Cats

Thiago Raulino Dal Pont
Automation and Systems Department
Federal University of Santa Catarina

July 4, 2022

Abstract

1 Introduction

In this sixth programming exercise, we are going to focus on image classification of cats based on Convolutional Neural Networks (CNN), Logistic Regression (LR) and Simple Neural Networks (NN) with one hidden layer.

Our goal is to maximize the accuracy on the test set. To do so, based on the Keras and Scikit Learn libraries, we implement a grid search for the best parameters for bothLR and NN. We could not apply the grid search for CNN, but we tried some hyperparameter tweaks.

This work is divided as follows:

- Section 2 presents the required materials, methods and studies to finish the exercise;
- Section 3 presents the implementations
- Section 4 presents the results and discussions.
- Section 5 presents the conclusions

2 Methods and Materials

For model evaluation, we applied accuracy to estimate how well the models fit the data:

$$Acc = 100 \cdot \frac{|y_{act} - y_{pred}|}{y_{act}} \tag{1}$$

In terms of tools versions, we used:

- Python 3.9.12
- Numpy 1.21.5
- Matplotlib 3.5.1
- Scipy 1.7.3
- Keras 2.4.3

3 Experiments

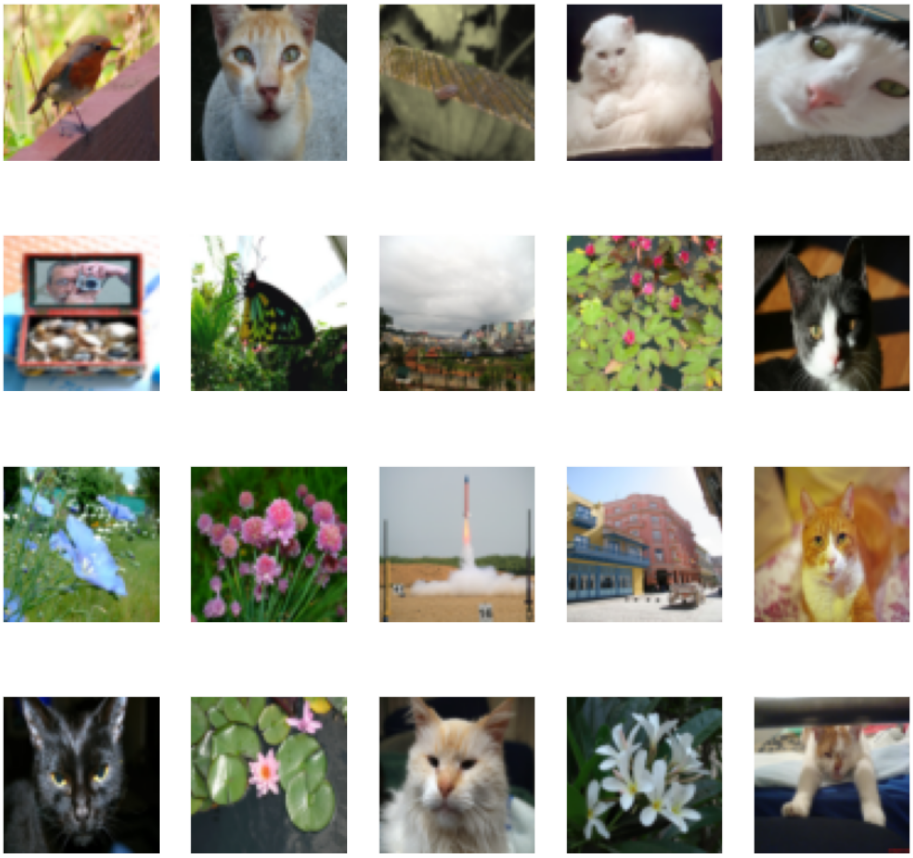
3.1 Data visualization

The experiments from this assignment relate to image classifications with or without cats. Figure 1 shows a sample of the training set.

The dataset has 209 and 50 images in the train and test sets respectively, where the images have the size 64x64 and the RGB channels.

The dataset has two classes, Cat vs Not Cat and there is some imbalance, as shown in Figure 2.

Figure 1: Sample of the training set



3.2 Data preparation

In this step, we loaded the data from a h5 file and run some steps to prepare the data for the models. Then, for

First, we calculated the class weights. Then, based on the input type accepted by each technique, we follow two paths: just use the raw image in the case of CNN or convert the image to gray scale[1]. By doing this, we keep most of the visual features and reduce the input size. In the next step, we flatten the image into a vector of 4096 elements, and finally, we normalized the array using `MinMaxScaler`

One can note that using CNN is easier to apply as it does not required the described preparation steps.

3.3 Grid search of hyperparameters

For each of the classification techniques, we implemented a class to handle the hyper-parameter search. When applying NN and CNN, we implemented a special function called `create_model` which wraps the creation of keras models based on the hyperparameters set. In addition, we used the class `KerasClassifier`, that uses such function and masks the keras model as if it was a Scikit-Learn model. By doing so, we could use the standard `GridSearchCV` class.

Based on the documentations[2] for each of the aforementioned libraries, we set up a wide range of parameters, trying to maximize the space search.

After search for the best hyperparameters using the train set, we test the best model in the test set to get the final accuracy.

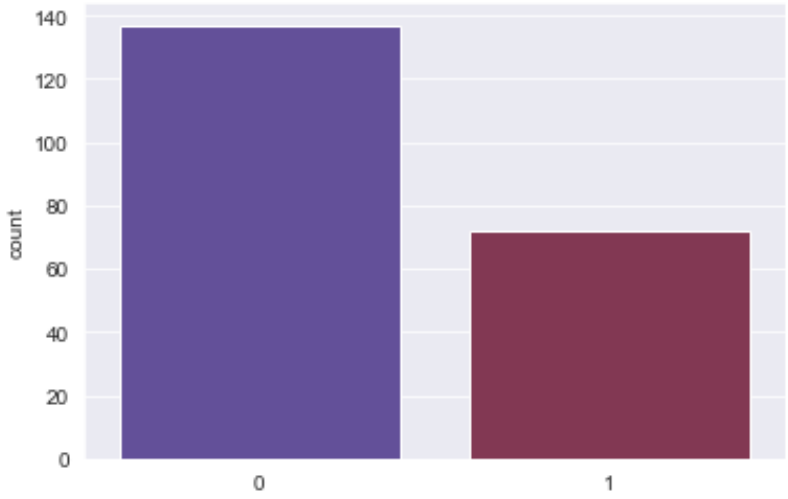
3.4 Classification models and hyperparameters

In this section, we present the the models and architectures

[1] Based on the REC601 Luma

[2] https://keras.io/api/
https://scikit-learn.org/stable/modules/classes.html

Figure 2: Class imbalancing in the train set

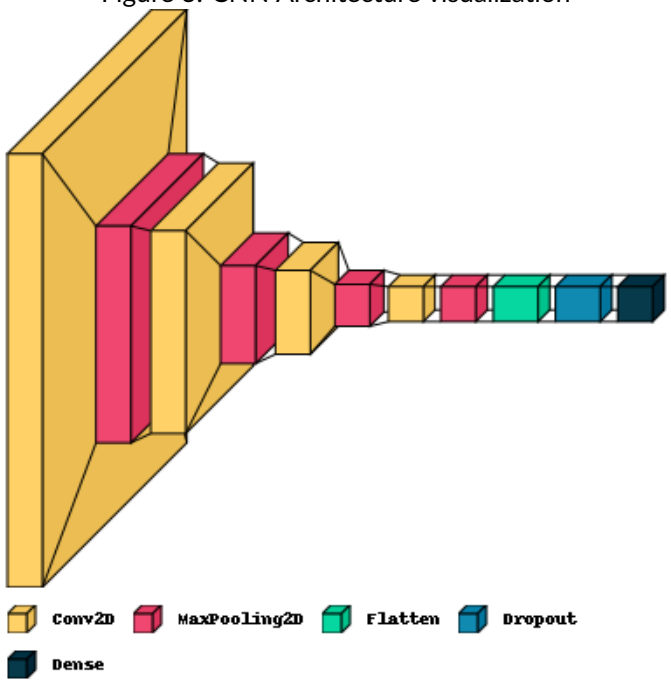


3.4.1 Convolutional Neural Network

For the CNN, we used the architecture from the original exercise. Although we tried some adjustments in the hyper-parameters, we could not find any combinations that leveraged improvements in accuracy.

In Figure 3, we present the CNN architecture.

Figure 3: CNN Architecture visualization



3.4.2 Logistic Regression

In terms of Logistic Regression, we used the implementation from Scikit Learn. The input is the flatten image in gray scale.

For the grid search, we used the parameters described in Table 1.

Table 1: Logistic Regression Params

Hyperparam	Range of values
max_iter	32, 64, 128, 256, 512, 1024
penalty	L2, None
C	0, 1e-10, 1e-9, 1e-8, 1e-7, 1e-6, 1e-5
solver	"lbfgs","liblinear","sag", "saga"
l1_ratio	0, 0.25, 0.50, 0.75, 1

3.4.3 Neural Network with one hidden layer

We also used a neural network with one hidden layer. The input is the flatten image in gray scale. The used hyper-params for grid search are described in Table 2

Table 2: Neural Network params

Hyperparam	Range of values
epochs	16, 32, 64
dropout	0.5, None
lr	1e-4, 1e-3
batch_size	32,64
hidden_layer_size	256,512,1024
optimizer	Adam, SGD
hidden layer activation	tanh, sigmoid, softmax, selu

4 Results

4.1 Convolutional Neural network

After running the training, we check the performance of the CNN in the test set, and the confusion matrix is show in table 3.

Table 3: Confusion Matrix for CNN in test set

		Predicted	
		Non Cat	Cat
Actual	Non Cat	16	1
	Cat	7	26

The performance achieved is shown in Table 4.

One can note that CNN is a good model to detect cats in images. In general, most failures happen when the model can not detect the cat in the image.

Although we could not achieve accuracy higher than 90%, the results are acceptable considering the small dataset.

4.2 LR

After running the grid search using the hyperparameters from Table 1, the following parameters were elected as the best.

- C = 1e-10
- l1 ratio = 0
- max_iter = 32
- penalty = L2
- solver = LBFGS

Table 4: Metrics for CNN in the test set

Metric	Value
Accuracy	84%
Macro F1-Score	0.83

Table 5: Confusion Matrix for LR in test set

		Predicted	
		Non Cat	Cat
Actual	Non Cat	17	0
	Cat	33	0

And the confusion matrix in the test set for this setup is shown in Table 5

The metrics achieved are shown in Table 6

It is clear that even after running a grid search, the Logistic Regression did not learn how to detect cats in images. Considering it is a "single neuron network", it can not handle all the visual complexity from cats. It is too simple for that task.

4.3 NN

After running grid search in the hyper-parameters from Table 2, the following parameters were selected:

- batch_size= 32
- dropout = None
- epochs = 16
- hidden activation = SeLu
- hidden layer size = 1024
- LR = 0.001
- Optimizer = Adam

Based on these parameters, the predictions on test set resulted in the Confusion matrix from Table 7.

And the metrics achieved are presented in Table 8.

Although the performance from simple NN is better than the Logistic Regression, it could not predict cats correctly. This result may arise from two factors: the input structure and the network complexity. From the former, we can see that flatten image is not a good input and from the latter, we see that a single hidden layer is not enough to handle the complexity of cats visual features.

5 Conclusions

In this assignment, we applied models for the image classification task: Convolutional Neural Network, Logistic Regression and Neural Network with one hidden layer.

The best results came from the CNN, as it can handle the visual features based on the convolutional filters. However, we could not improve the results based on hyperparameter tweak. Still, one can note that using CNN is easier to apply as it does not require data preparation steps.

The LR and NN could not classify the image with reasonable performance. They do not have enough "space" to handle the complexity of the image classification task.

Table 6: Metrics for LR in the test set

Metric	Value
Accuracy	34%
Macro F1-Score	0.25

Table 7: Confusion Matrix for NN

		Predicted	
		Non Cat	**Cat**
Actual	**Non Cat**	8	9
	Cat	13	20

Table 8: Metrics for NN in the test set

Metric	Value
Accuracy	56%
Macro F1-Score	0.53

References

[1] S. Haykin, *Neural networks and Learning Machines*. 2008.

[2] "Mathematical functions - numpy." https://numpy.org/doc/stable/reference/routines.math.html. Accessed 30 Apr 2022.

[3] "Matplotlib 3.5.1 documentation." https://matplotlib.org/3.5.1/index.html. Accessed 30 Apr 2022.

[4] "Scipy api - scipy v1.8.0." https://docs.scipy.org/doc/scipy/reference/. Accessed 12 May 2022.

[5] "Complete python numpy tutorial (creating arrays, indexing, math, statistics, reshaping)." https://www.youtube.com/watch?v=GB9ByFAIAH4. Accessed 30 Apr 2022.

[6] "Intro to data visualization in python with matplotlib! (line graph, bar chart, title, labels, size)." https://www.youtube.com/watch?v=DAQNHzOcO5A. Accessed 30 Apr 2022.