Convolutional Neural Network (CNN) to classify CIFAR100.

M.Sc. Sergio Chavez Lazo

Access to Python project here

SUMMARY

This project presents the creation of a convolutional neural network (CNN) to classify images belonging to the CIFAR100 database. The results obtained from 7 models with progressive complexity are compared. After a process of transfer learning and two-stage optimization of the hyperparameters in model 7, a model with an accuracy of 59.5% was built. Although the result can still be improved, there is a significant improvement compared to the initial results with the first model - Simple Neural Network - (18.5%). Models 1 and 7 were also tested for the super classes of the database with even more significant improvements (37.7% and 65.6% respectively).

INTRODUCTION.

The objective of this project was to create a model that classifies images belonging to CIFAR100, a dataset consisting of 100 classes with 600 images each. A cumulative approach was chosen to progressively build the final model. It started with a simple model with a single hidden layer (Model 1) and ended with a hyperparameter-optimised model using transfer learning from MobileNet (Model 7). The same seed value was set to guarantee the same starting point for the gradient descent process in all models and ensure the comparability of models. Although the central objective was to improve the classification accuracy for the 100 classes, the evolution of the improvement was also evaluated for the 20 super-classes into which the classes are grouped.

I. PREPROCESSING

Downloaded CIFAR100 database created 2 datasets: training and test. As their names suggest, the first set is used to create and fit the model while the second is used for model evaluation. However, for reasons of completeness and correct assessment of the models, the test segment should only be used for the final evaluation and not during the training process. Therefore, 20% of the training data was randomly selected in a stratified way to build a new data set called validation. The training process of the models was evaluated based on the results in this group.

Table1. Data segmentation

Training Dataset		**Test Dataset**
Training	**Validation**	
100 classes 400 images per class	100 classes 100 images per class	100 classes 100 images per class

The images were converted into matrices and then normalized by dividing the values by the maximum value found (255). This processing is standard and is recommended because it helps efficiency and improves the generalisability of CNN.

II. DESIGNS

The final model was built under a comparative and progressive approach based on the previous models. Thus, each model contains the previous model in its topology plus some added aspects. This means that the hyperparameters that were added are maintained until model 7 where they were optimized.

Table2. Models

Model	Description
1	Simple CNN with 10 filters and kernel 2, MaxPooling function and without extra hidden layers
2	Inclusion of extra hidden layer with 128 neurons
3	Inclusion of a new extra hidden layer with 64 neurons
4	Inclusion of dropout rate of 20% between each hidden layer
5	Inclusion of previous stage for augmentation (random flip in vertical and horizontal position)
6	Inclusion of MobileNet using transfer learning
7	Optimization of hyperparameters using RandomSearch and GridSearch

All models were trained under the same parameters in terms of number of epochs (50) and batch-size (50). To avoid inefficient training periods, a stop parameter was included in case the loss of validation data stops decreasing for 3 consecutive epochs. As the true labels were not hot-encoded, 'sparse_categorical_crossentropy' was used as a loss measure and the softmax activation function was used in the last layer for all models. Finally, ReLU was used in all hidden layers. No change was contemplated during the hyperparameter optimisation process due to the computational advantages of the efficiency of this function - only involves simple mathematical operations that are faster than other functions such as sigmoid or tanh.

Reasoning for modifications

Model 2&3: Inclusion of intermediate layers generally increases the accuracy of classification models compared to simple neural models.

Model 4: Inclusion of dropout helps to avoid overfitting since it 'forces' all neurons to optimise their individual performance for the final objective of the model.

Model 5: Data augmentation helps to generate variants of the inputs in such a way as to obtain more cases with which to train the model.

Model 6: Through a Transfer Learning, all MobileNet parameters were used except for the last layer (which was modified to include the architecture of the previous models and the final classification of 100/20 classes/superclasses). MobileNet is a type of CNN developed by Google in 2017 that is designed to be computationally efficient. This model was preferred over others because of its time efficiency and because it takes as inputs images of the same dimensions as the CIFAR100 images (32,32). Thus, image resizing and the risk of distorting the content of the images was avoided.

Model 7: Hyperparameters were optimized after a two-stage process. First, RandomSearch was used to apply 3-layer cross validation for each of the 30 random combinations of candidate hyperparameters. For efficiency, epochs and batchsize were modified to 20 and 250 respectively. The 3 best combinations (in terms of accuracy) were analyzed to select common and even variant hyperparameters for the next stage of optimization with GridSearch. There, the models were created and retrained in a greater number of epochs and a smaller batch-size considering all possible combinations of variant hyperparameters for the previous top 3 models

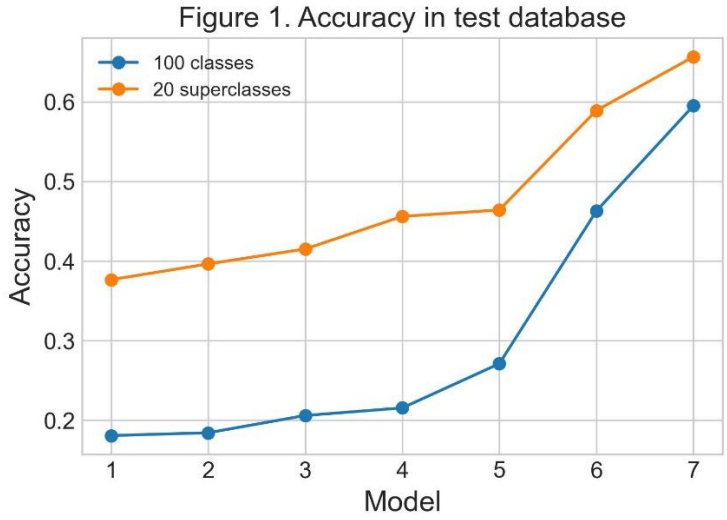
Table3. Tested hyperparameters

	Neurons	Optimizer	Learning-Rate	Number of HL	DropoutRate	Filters	Kernel
Range	64 to 650	Adam/SGD	0.001-0.01-0.0001	1 to 3	0.1 to 0.3	5 to 20	2 to 5

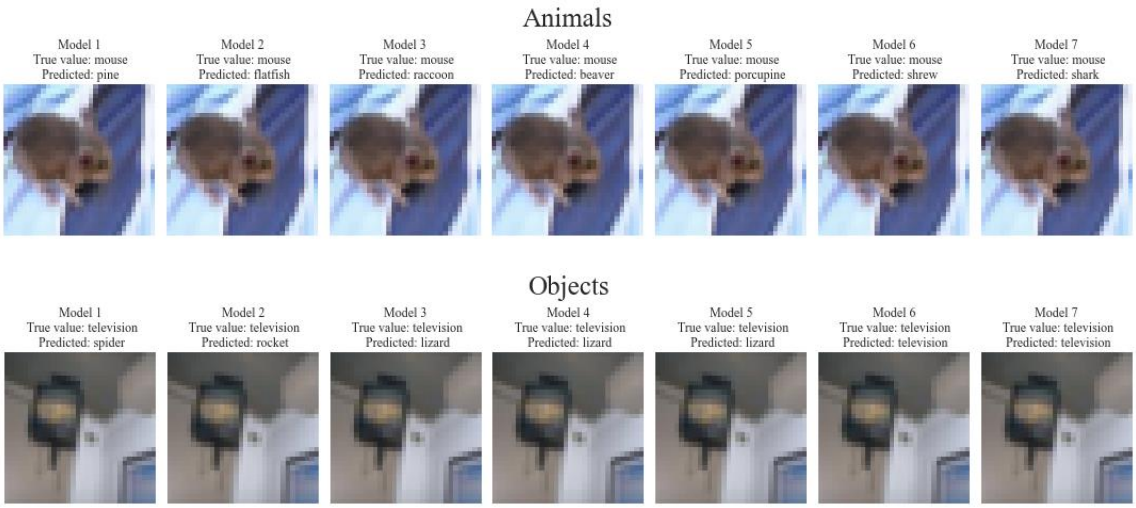
RandomSearch allowed to contemplate a greater range of possibilities for hyperparameters, while GridSearch exhaustively explored the combinations of the still unclear parameters. In this sense, this process allowed experimenting with wide ranges for the hyperparameters and then ensuring that the subsequent selection was the optimal within what was considered.

III. RESULTS

Figure 1 shows the progression in accuracy for the test database for the classes and super-classes



Evaluation of results for each of the 100 classes revealed that the model is especially good at identifying objects such as cupboards and roads (above 80% in accuracy, recall and f1-score). However, it is poor at identifying animals and people (below 20% on the same metrics). This is confirmed by looking at the confusion matrix for super-classes (Annex1). This would suggest that MobileNet may not be efficient at detecting living beings and other CNN could be used.

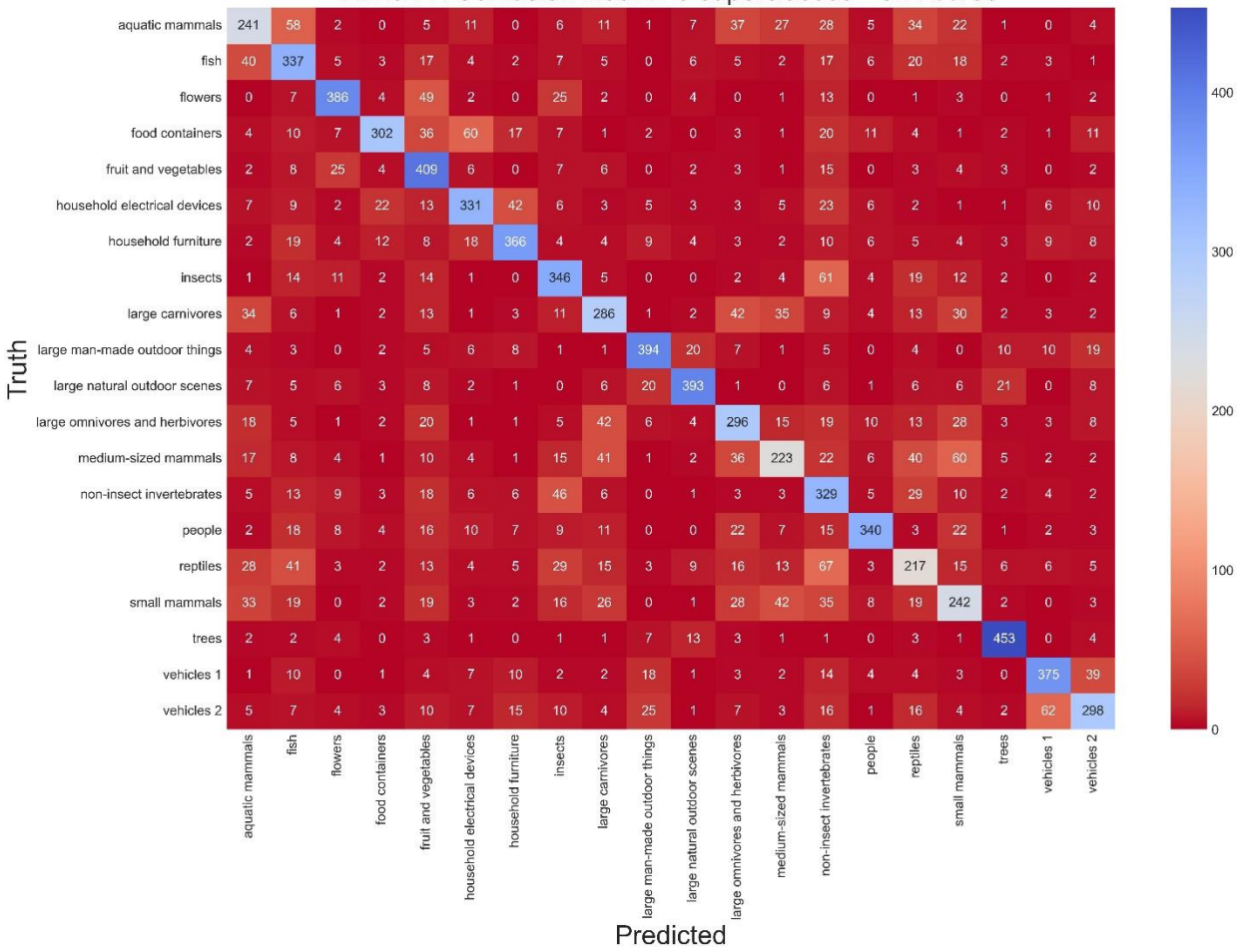


IV. CONCLUSIONS

The final model built was significantly improved compared to the initial model. However, a precision of 59% is far from ideal and further improvements should be implemented. The major constraint in the construction of the model was time. Due to the impossibility of accessing GPUs, training was performed with CPUs, leading to long waiting times (Annex2). Opting for a model other than MobileNet for the transfer would have been even more time-consuming. For this reason, the present work can be further improved by (i) exploring other, more optimal prior models for animal and people classification, (ii) expanding the range of possible hyperparameters, and specially (iii) accessing GPUs for faster and more efficient training.

ANNEX.

Annex 1. Confusion Matrix 20 superclasses - CIFAR100



Annex 2. Time spent on training