Convolutional Neural Network (CNN) to classify CIFAR100 images.

M.Sc. Sergio Chavez Lazo

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).

Access to Python project here

INTRODUCTION.

This project aimed to create a model that classifies images belonging to CIFAR100, a dataset that has 100 classes containing 600 images each. While the central focus was to improve the classification accuracy for the 100 classes, we also evaluated the evolution of the improvement for the 20 super classes into which the classes are grouped.

A comparative and progressive approach to the models tested was chosen to construct the 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 the MobileNet model (Model 7). The same seed value was set for the Keras environment and the notebook in general to ensure the same starting point for the gradient descent process in all models. In this way, the comparability of the models was ensured.

I. PREPROCESSING

The download of the CIFAR100 database created 2 large 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 model 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[1]. The training process of the models was evaluated based on the results in this group.

Figure 1. Data segmentation

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

[1] Stratified sampling was used to ensure the presence of all classes while maintaining the proportionality of the number of images for each 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 convergence, efficiency and improves the generalisability of CNN.

II. DESIGN

The final model was built under a comparative and progressive approach based on the previous models. Thus, it can be interpreted that 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 optimised after a two-stage search process. Table 1 presents a summary of the model's characteristics.

Model	Desciption
1	Simple CNN with 10 filters and kernel 2, MaxPooling function and without extra hidden layers
2	Inclusion of a extra hidden layer with 128 neurons and relu as activation function
3	Inclusion of a new extra hidden layer with 64 neurons and relu as activation function
4	Inclusion of dropout rate of 20% between each hidden layer
5	Inclusion of a previous stage of 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 batchsize (50). To avoid inefficient training periods, a stop parameter was included in case the loss in the validation data stops decreasing for 3 consecutive epochs. In such cases, the CNN kept the parameters that obtained the best results in the validation segment.
- As a no one hot encoded classification case, 'sparse_categorical_crossentropy' was used as a loss measure and the softmax activation function was used in the last layer for all models.
- ReLU is used in all the layers included. No change was contemplated during the hyperparameter optimisation process due to the computational advantages of the efficiency of this function (because ReLU only involves simple mathematical operations (max(x,0)), it is faster to compute than other functions such as sigmoid or tanh). Testing other measures would have implied further extending a test period that already exceeded 24 hours.

<u>Reasoning for modifications</u>

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

Model 4: The inclusion of dropout helps to avoid the problem of 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 process, 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 images). MobileNet is a type of convolutional neural network (CNN) developed by Google in 2017 that is designed to be computationally efficient, particularly for mobile and embedded applications. 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.

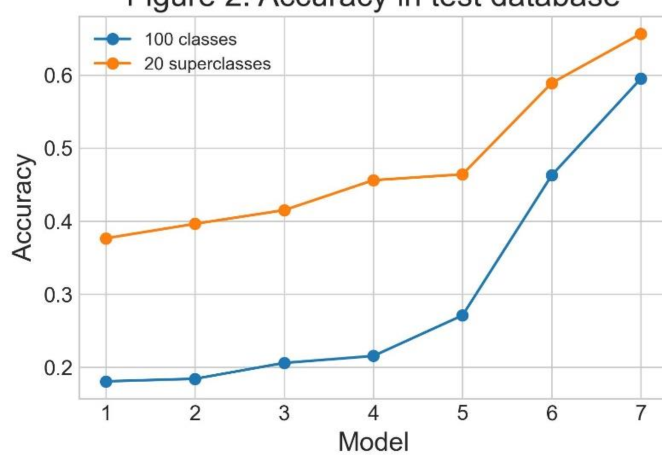
Model 7: The hyperparameters in Figure 1 were optimized after a two-stage process. First, RandomSearch was used to apply 3-layer cross validation for each of the 30 random combinations of candidate hyperparameters. For efficiency, in each case 20 epochs with a batch of 250 were used. The 3 best combinations were used 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 batchsize, all the possible combinations of the hyperparameters that varied in the top 3 of the previous stage.

RandomSearch is an approach that allows to contemplate a greater range of possibilities for hyperparameters, while GridSearch is a more exhaustive approach that uses all possible combinations. In this sense, this process allowed experimenting with wide ranges for the hyperparameters and then ensuring that the subsequent selection is the most optimal within what was considered.

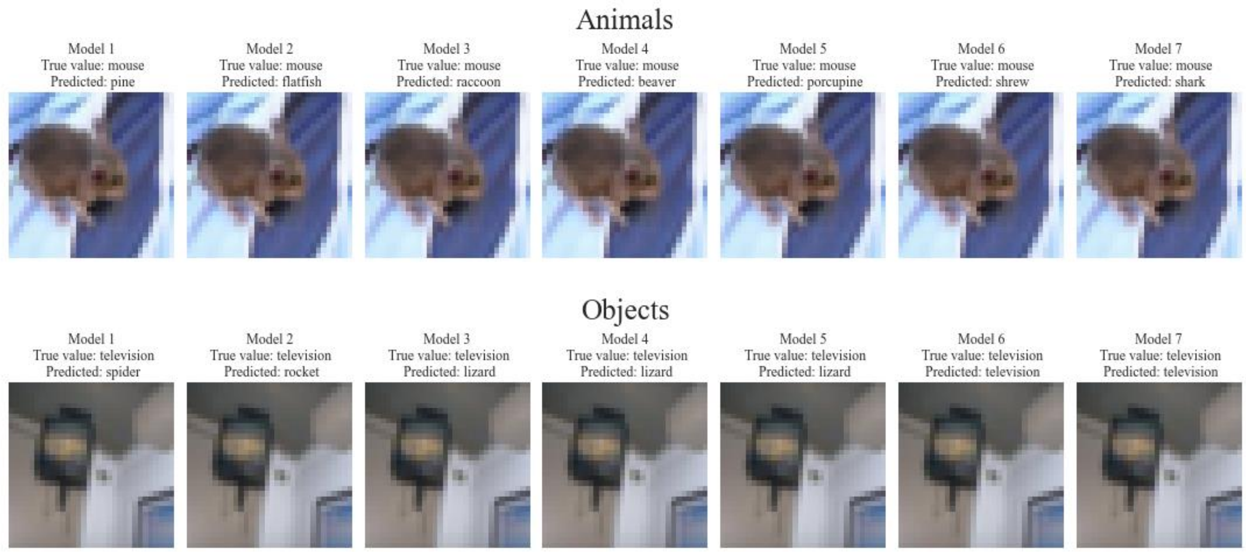
III. RESULTS

Figure 2 shows the progression in model accuracy for the test database for the 100 classes and 20 super-classes.

Figure 2. Accuracy in test database



The evaluation of the ranking metrics for each of the 100 categories revealed that the model is especially good at identifying objects such as cupboards and roads (both above 80% in accuracy, recall and f1 score). However, it is poor at identifying animals and people (all below 20% on the same metrics). This is confirmed by looking at the confusion matrix for the superclasses (Annex 1). This would suggest that MobileNet may not be very efficient at detecting animals and another CNN could be used for transfer learning.

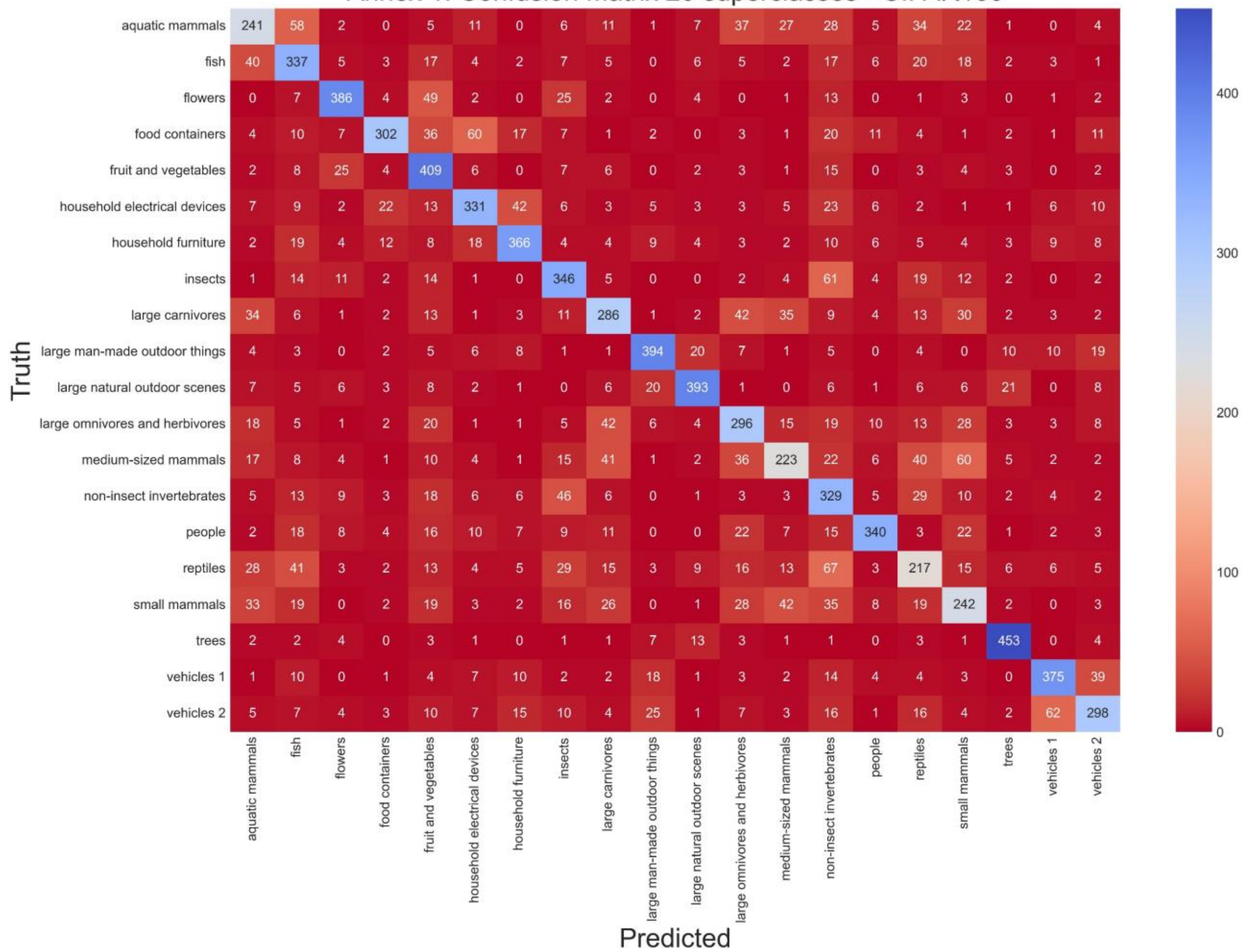


IV. CONCLUSIONS

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 very long waiting times (Annex 2). 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 image 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