Model memory optimisation using MNIST data

Omar El Nahhas, 214316IV

I. Introduction

The goal of this assignment is to optimise a deep-learning model with respect to its memory size. The model is trained on the MNIST data set, containing 60000 training images and 10000 validation images (28x28; grayscale). The memory size in this report is measured in total model parameters and the file size of the model converted to a TensorFlow Lite model.

II. Methodology

Using the neural network architecture presented in Fig. 2, optimisations were performed in order to reduce the memory size of the model, while maintaining as much of the accuracy provided by the initial model. Considered optimisations techniques were experimenting with the learning rate, optimizer algorithms, number of epochs, batch size, activation functions, neurons per layer, types of layers and number of layers. Besides the 10000 validation images, another 10 images were created as external test set for validation purposes, see Fig. 1.

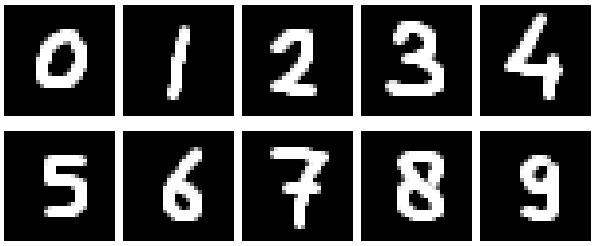

Fig. 1: External test set created for model validation.

Using the new model optimised on memory size, the results are compared with two existing architectures to evaluate the new model's effectiveness. The chosen models are SqueezeNet (2016) and Efficient-CapsNet (2021), based on their relatively low size compared to alternative models.

III. Results

The baseline model, i.e. initial model, can be seen in Fig. 2. Through trial and error, the baseline model structure is altered by changing above-mentioned optimisation measures. The goal of the optimisation is to reduce the size of the model as much as possible, while minimising the loss in accuracy compared to the baseline model. The initial hypothesis is that reducing the parameters of the model will decrease the accuracy of that model significantly when classifying MNIST.

Throughout the series of experiments, the changes to the model have been documented. Table I shows the used optimisation methods and whether the optimisation have a positive (↑) or negative (↓) impact on the model's parameters, .tflite file size, validation accuracy (on 10000 MNIST images) and test accuracy (on 10 external test images from Fig. 1).

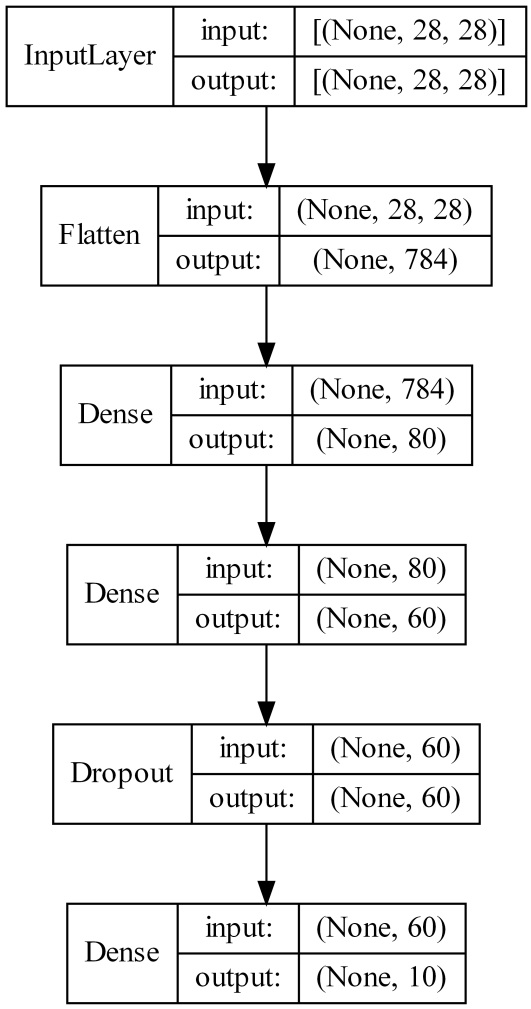

Fig. 2: The baseline model to be optimised.

TABLE I: Experiment results for model size optimisation.

↑/↓	Optim. method	Params.	Size	Valid.	Test
–	Initial	68,270	269 KB	0.9784	0.7
↑	CNN + MP (8,8)	10,994	45,5 KB	0.9753	0.9
↓	Drop Dense layer	13,610	55,5 KB	0.9793	0.8
↓	Change neurons (2,2)	728	5,5 KB	0.3830	0.2
↓	Dropout layer (20%)	10,994	45,5 KB	0.9812	0.8
↑	2x CNN + MP (3,3,4)	468	5 KB	0.9279	0.7
↑	Change neurons (3,4,5)	707	6 KB	0.9483	0.9
↑	**Leaky ReLu (0.1x)**	**707**	**6,5 KB**	**0.9636**	**1.0**

All the experiments were, with exception of the mentioned changes, conducted under the same circumstances. The used loss function is sparse categorical entropy, using the Adam algorithm for optimization. Softmax is used as classifier, and the training process is run for 20 epochs for every experiment.

The newly proposed model architecture is seen in Fig. 3, which has significantly lower neurons per layer compared to the baseline model in Fig. 2.

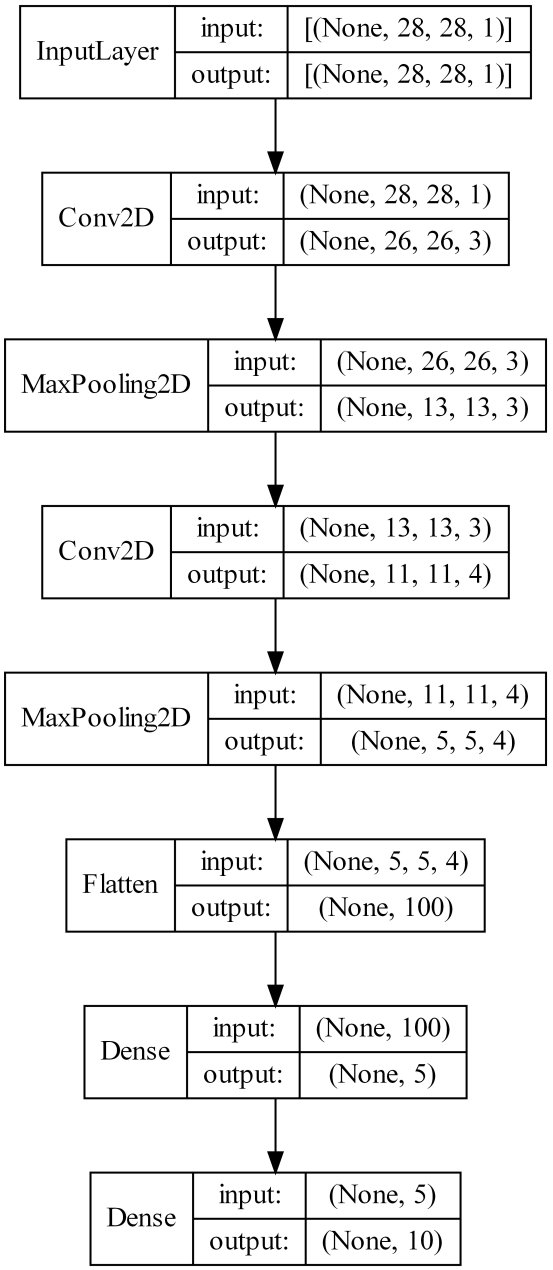

Fig. 3: The proposed model for memory optimisation.

Table II shows the comparison of the proposed model with SqueezeNet[1] and Efficient-CapsNet[2]. Here, the models again run for only 20 epochs to maintain similar conditions across testing cases. However, things like cost function and optimization algorithms are not changed to match, as they shape part of the definition of that particular architecture. Observing the results, the proposed model is unique in its extremely small size, but exchanges size for approximately 3% accuracy on the MNIST validation set compared to the Efficient-CapsNet implementation. Squeezenet only scores approximately 0.3% extra accuracy, but at significantly higher memory requirements.

TABLE II: Comparison between models optimised on size.

Architecture	Params.	Size	Valid.	Test
Proposed model	707	6,5 KB	0.9636	1.0
SqueezeNet	45,088	185 KB	0.9671	0.8
Efficient-CapsNet	161,824	656 KB	0.9978	1.0

Efficient-Capsnet has been ranked as one of the top architectures to classify MNIST as of 2021[3], where SqueezeNet is a relatively old implementation which was published in 2016. The newly proposed model in this report is a feasible alternative for low-memory embedded applications, classifying 10 out of 10 test images correctly. The hypothesis of this research can therefore be rejected, as the model with only 707 parameters scores equally (within margin) or even more accurate on selected data sets than counterparts with a large architecture.

Finally, the TensorFlow native solutions for network pruning and quantization have been tested on the proposed model. This resulted in a significantly lower model accuracy while having more parameters and therefore a larger .tflite file size. As a result, these optimisations are not applied on the final model.

IV. Discussion

The choice of only 20 epochs for model comparison could have a negative impact on larger models, as they would potentially require more epochs before correctly tuning all the weights. At this choice of epochs, the proposed model seemed as good as fully converged, while the bigger architectures were still making significant reductions on the loss in the final epochs. Further research should consider this and experiment with a varying number of epochs.

Pruning the model was expected to make the model smaller while retaining accuracy, but resulted in significantly lowering the accuracy while the size went up. This phenomenon can be explained because the optimizations were applied on an already highly-optimised model. The TensorFlow pruning and quantization functions apply generalistic optimisations, which removed vital pieces from an already 'skinny' model. Further research should investigate under which circumstances such optimisation methods can be applied after already manually optimising the model.

During these experiments, the MNIST data set was used, containing 60000 training images, 10000 validation images and 10 test images. Having this amount of reference data reduces the need of a complex model immensely, as even a basic model can define a relation between input and output given a 'perfect' data set. Further research should be performed on more complex data to test if a small model (e.g. 707 parameters) can still compete with its much larger counterparts, checking if this research's hypothesis conclusion still holds in situations with more complex data.

[1]https://www.kaggle.com/somshubramajumdar/squeezenet-for-mnist
[2]https://github.com/EscVM/Efficient-CapsNet
[3]https://paperswithcode.com/sota/image-classification-on-mnist