IL2230 Hardware Architecture for Deep Learning
Lab 3C-Group 3

Andreas Lezdins, Patrik Johansson, Jiayi Feng, Lihao Yan, Ruobing Li

December 12, 2020

Contents

1 Introduction

One of the most common ways to evaluate pattern recognition in machine learning is through the MNIST dataset. The MNIST dataset contains handwritten digits representing numbers 0-9 in ten classes. The simplicity of MNIST makes it a good baseline dataset for evaluating common neural networks (NN).

In this lab, we compare the inference performance using the MNIST dataset on the commonly known LeNet-5 convolutional neural network (CNN) and two implementations of multilayer perceptron (MLP) in terms of classification accuracy, run-time and memory consumption.

An MLP consists of at least three layers, one input layer and one output layer with an arbitrary number of hidden layers in between. All layers in an MLP are fully connected layers and the hidden layers contain an arbitrary number of neurons. Figure 1 shows an example of an MLP with three inputs, two outputs and one hidden layer with four neurons.

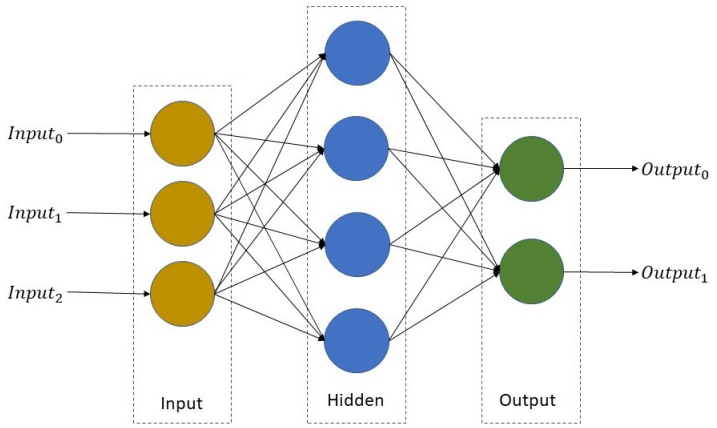

Figure 1: Structure of an MLP with three inputs, two outputs and one hidden layer.

LeNet-5 structure was developed in late 20th century [1] and is one of the first successful CNNs that automatically extract features from real-life problems. This proved that manually-made feature extractors might not be the best solution for extracting features in machine learning problems. The structure can be seen in Figure 2.

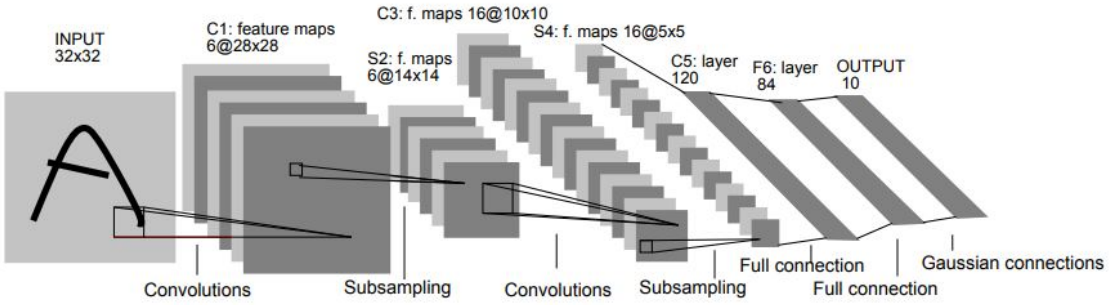

Figure 2: LeNet-5 structure.

2 Method

In this report we evaluate LeNet-5 and two MLPs, using one hidden layer and two hidden layers. The structure and implementation of the MLPs are described in Section 2.1. The full LeNet-5 structure is described in Section 2.2. The MNIST dataset and preparation of the data is described in Section 2.3. Lastly Section 2.4 describes the training and validation steps. The Deep-leaning framework used was PyTorch [2].

2.1 MLP

Since the MNIST dataset consists of $28 \times 28 \times 1$ grey scale images, both MLPs have $N_{input} = 28 \times 28$ neurons in the input layer. Both also have $N_{output} = 10$ neurons in the output layer, representing the ten classes that are to be determined. The number of neurons for the hidden layers is alternated according to Table 1. The outputs from all hidden layers are passed through the non-linear Rectified Linear Unit (ReLU) function.

Table 1: Tested MLP Neural Networks.

number of hidden layers	layer 1	layer 2	layer 1	layer 2
1 Hidden	30		270	
	60		300	
	90		400	
	100		512	
	120		1024	
	150		2058	
	180		4096	
	200		8192	
	210		16384	
	240		32768	
2 Hidden	100	30	1024	512
	200	70	2048	1024
	300	100	4096	2048
	400	130	8192	4096
	512	256	16384	8192

2.2 LeNet-5

The structure of LeNet-5 presented in Figure 2, shows two convolutional layers each followed by a pooling layer. The last three layers are fully connected layers used for classification. The convolutional layers uses 5×5 kernel for convolution with the first convolutional layer having depth 6 and the second depth 16. LeNet-5 uses max-pooling layers with a 2×2 kernel. The fully connected layers have 120, 84 and 10 neurons respectively. Numbers for each layer is presented in Table 2. The activation function used from the two convolutional layers and the first two fully connected layers is the tanh function.

2.3 Data preparation

The images in MNIST are of size $28 \times 28 \times 1$, i.e. gray scale images of size 28×28. The dataset is divided into 60,000 images for training and 10,000 images for validation, making it a total of 70,000 images. For all three networks the data is partitioned into batches of 4 images. The images is normalized with mean 0 and standard deviation 1, since training will be faster according to [1].

LeNet-5 is constructed for input data of size 32×32 and the MNIST dataset consists of images of size

28×28 we pad the data with size 2. This adds two zero value pixels around each image increasing the image size to match the LeNet-5 architecture.

Table 2: LeNet-5 structure.

Layer	Nerons	Input	Output	Kernel size
Conv 1		1	6	5x5
Conv 2		6	16	5x5
Max-pooling				2x2
Fully-connected 1	120			
Fully-connected 2	84			
Fully-connected 3	10			

2.4 Training and validation

The networks were trained utilizing a Nvidia Geforce 1080 GPU. The performance of the networks were measured on both the GPU and the CPU, an AMD Ryzen 3700X. Stochastic gradient decent was used to update the internal parameters of the networks, and they were trained with 20 epochs. The learning rates were varied according to the same schedule used by [1], decreasing the learning rate depending on the epoch. Table 3 shows the learning rate for each epoch. To measure inference time consumption, CPU usage and GPU usage, we used the built-in PyTorch profiler [2]. This gives average time and memory consumption for one batch of images.

Table 3: Learning rate depending on epoch.

Epoch	Learning rate
1 - 2	0.0005
3 - 5	0.0002
6 - 8	0.0001
9 - 12	0.00005
$>$12	0.00001

3 Results

The accuracy of the neural networks was compared with the memory usage of said network shown in Figures 3 and 4. The accuracy was also compared to the inference time shown in Figure 5 and 6.

The complete performance data for the different NN tested is presented in Appendix A.

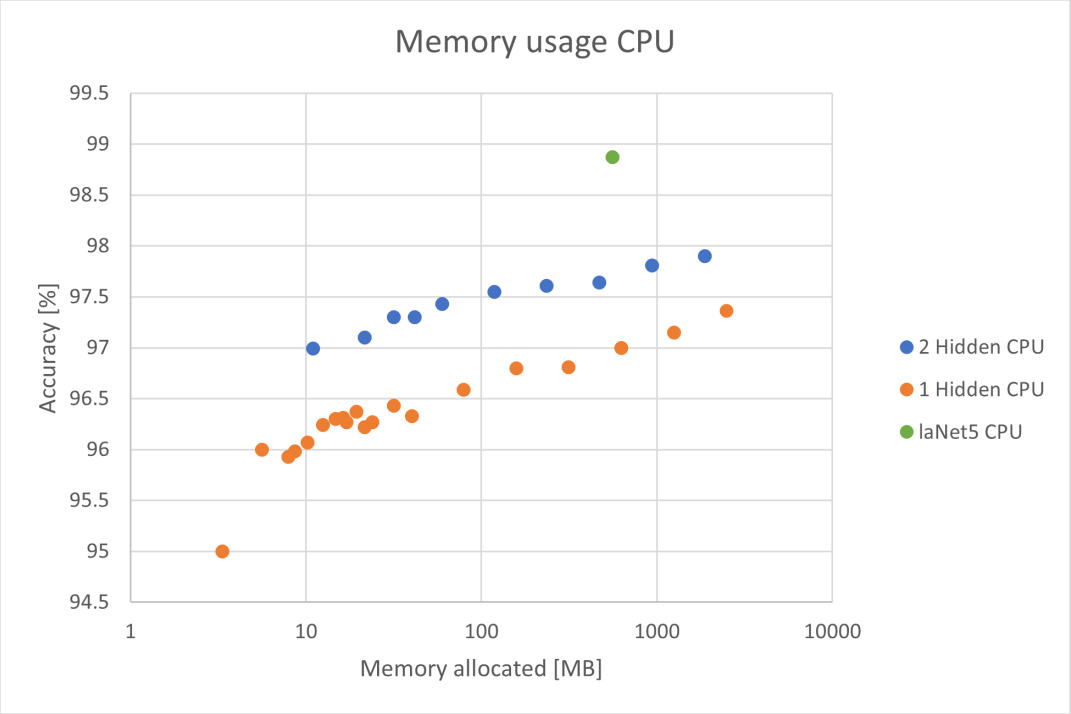

Figure 3: The accuracy of the tested NN as a function of the memory usage when running on the CPU.

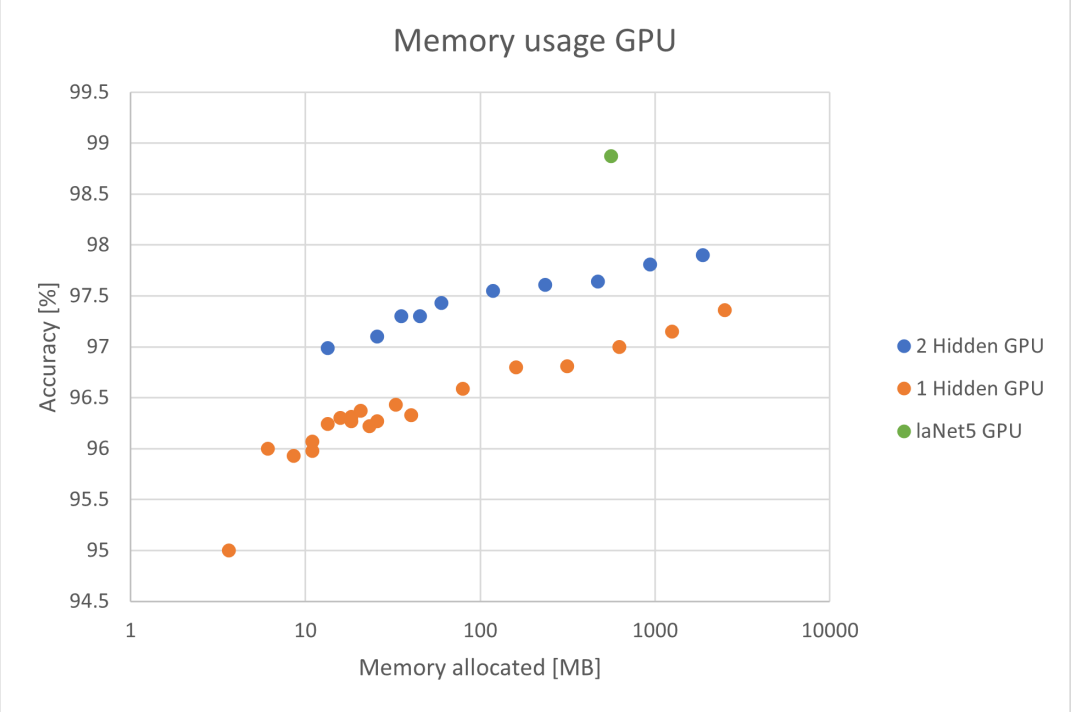

Figure 4: The accuracy of the tested NN as a function of the memory usage when running on the GPU.

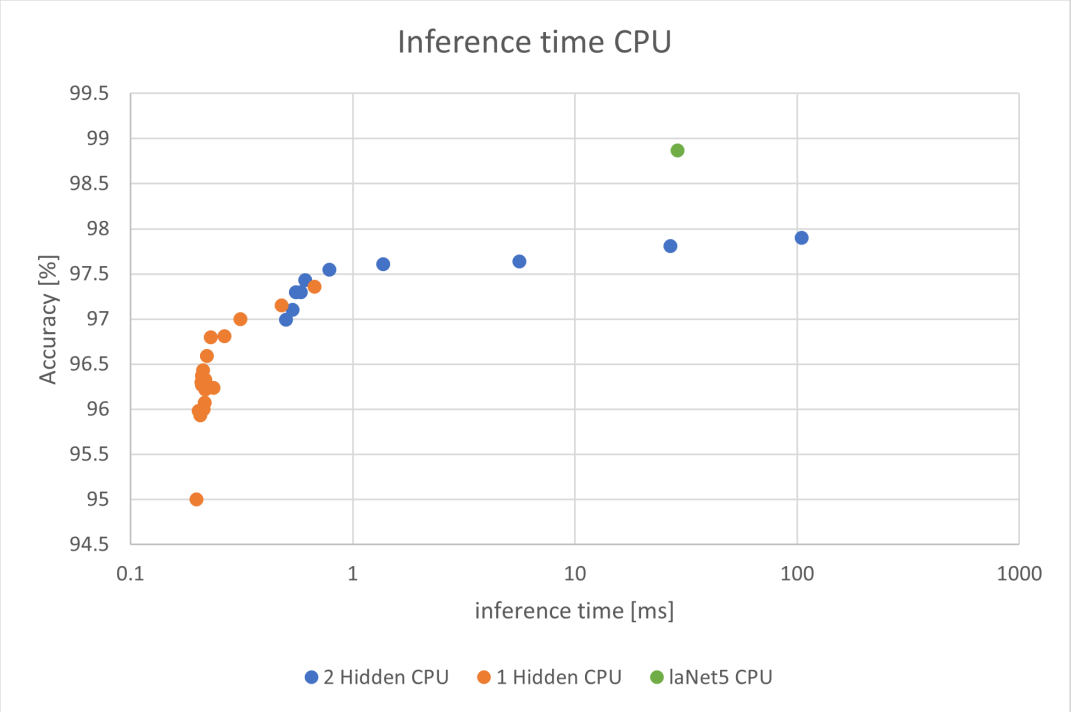

Figure 5: The accuracy of the tested NN as a function of inference time when running on the CPU.

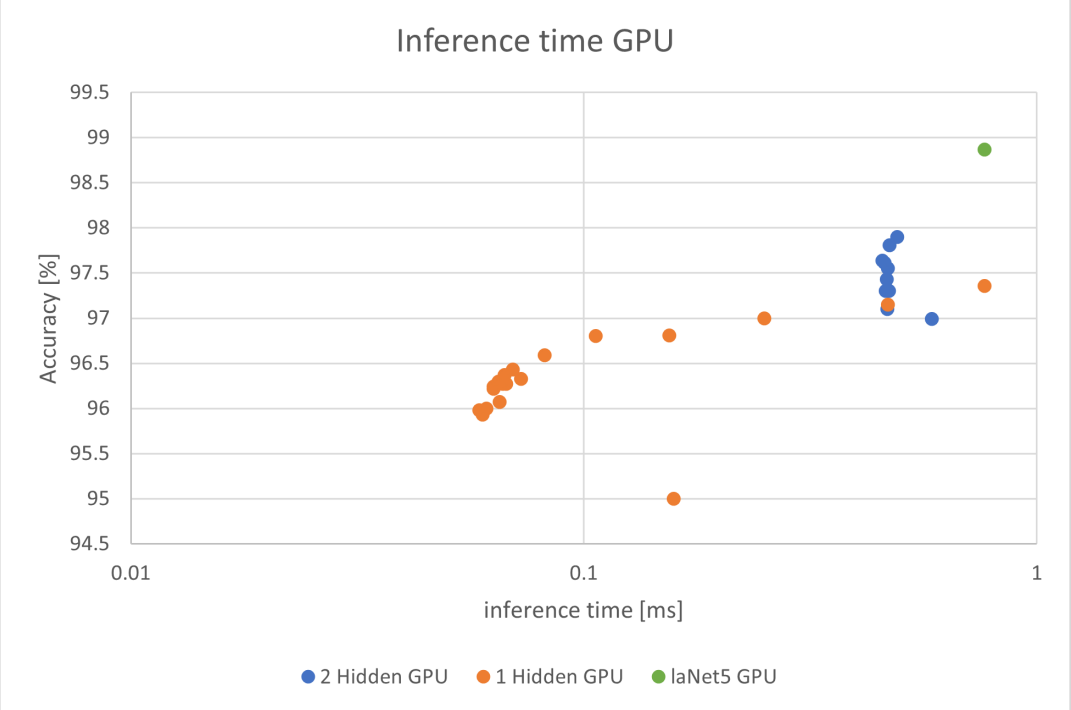

Figure 6: The accuracy of the tested NN as a function of inference time when running on the GPU.

4 Conclusion

LeNet-5 stands out as having a significantly higher accuracy then either of the two MLPs,on average LeNet-5 consumes more memory resources then the two MLPs, something that could be a limitation. However, the inference times for LaNet-5 can be as much as 10x the inference times for smaller MLPs. The inference times still being quite short in absolute terms, aprox 1 ms.

When comparing the two MLP solutions, it appears that for the same memory utilisation the two hidden layer MLP gives approximate 1% higher accuracy then the one hidden layer MLP. This is true when validating on the CPU Figure 3 and the GPU Figure 4.

For the inference times it is a clear difference depending on whether the neural network is validated on the CPU (Figure 5) or the GPU (Figure 6). Using the GPU the inference times for the two hidden layer MLP is roughly constant when increasing the amount of neurons in the hidden layers. However for the single layer structure increasing the amount of neurons increases the inference time. When the networks are validated on the CPU, the total amount of neurons in the network has a direct impact on the inference time. It does not matter if the increasing neurons are structured in single layers or multiple layers.

LeNet-5 outperforms both the MLP implementations in terms of accuracy, regarding how much the number of neurons are increased. However, since both MLP have fewer neurons i.e. fewer connections than LeNet-5 they both have lower inference time and memory usage.

References

[1] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[2] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

A Data

This appendix contains data produced from testing the implemented networks. The naming convention of the networks is as follows. LeNet5 denotes our implementation of the LeNet-5 described in section 2.2, and MLP 30 0 denotes an MLP with only a single hidden layer with 30 neurons. MLP 100 30 denotes an MLP with two hidden layers, 100 neurons in the first hidden layer and 30 in the second hidden layer.

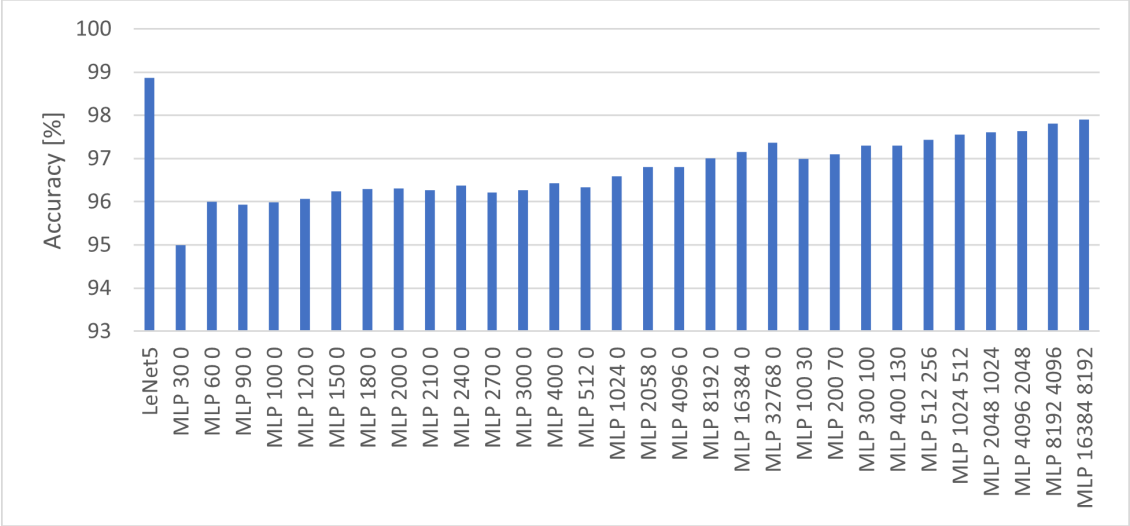

Figure 7: Accuracy of the tested NN implementations.

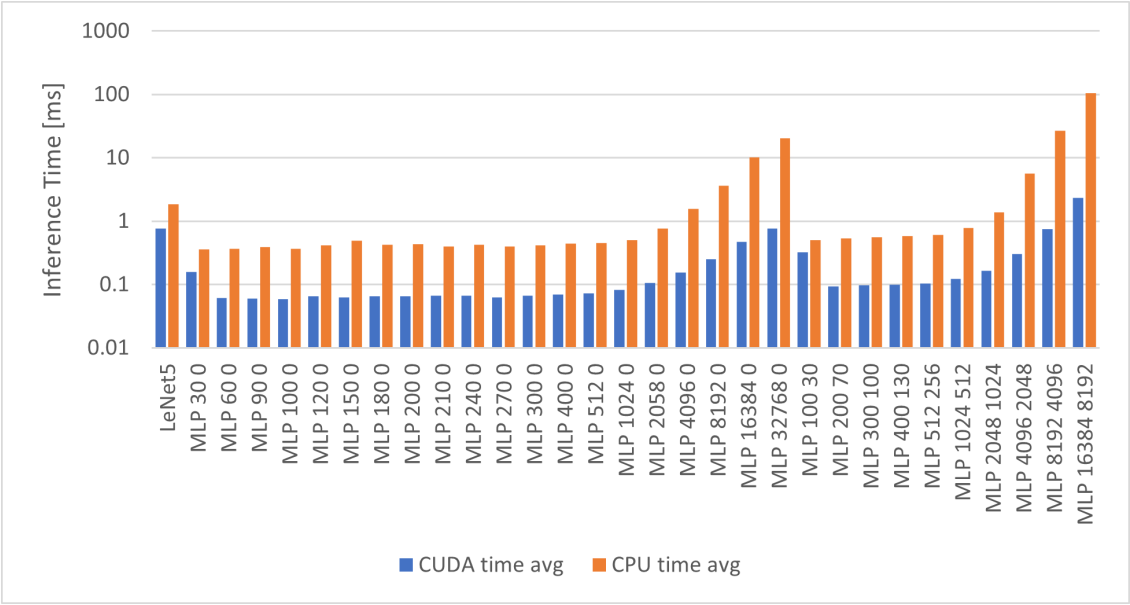

Figure 8: Inference time of the tested NN implementations.

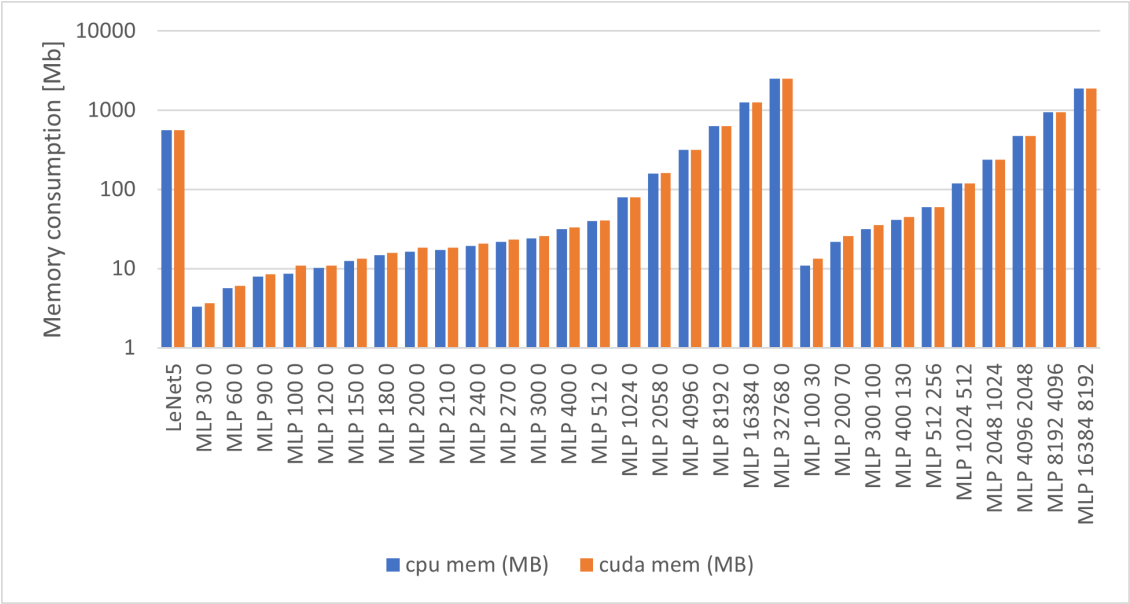

Figure 9: Memory usage of the tested NN implementations.