SAPIENZA
UNIVERSITÀ DI ROMA

DEPARTMENT OF COMPUTER SCIENCE

Training on Parameters Subspaces

DEEP LEARNING & APPLIED AI

Professor:
E. Rodolà

Assistants:
L. Moschella
D. Crisostomi

Student:
Minut R. Adrian,
1942806

Academic Year 2021/2022

1 Experiments

1.1 Exploring Subspaces

The following experiments were performed using the Gaussian Random Projection method. The aim is to confirm the reference paper results and to get some insights on the architectures and difficulty of the respective tasks.

In the plots there is a horizontal dotted line which represents the 90% accuracy baseline of the models trained without projection.

1.1.1 MNIST MLP

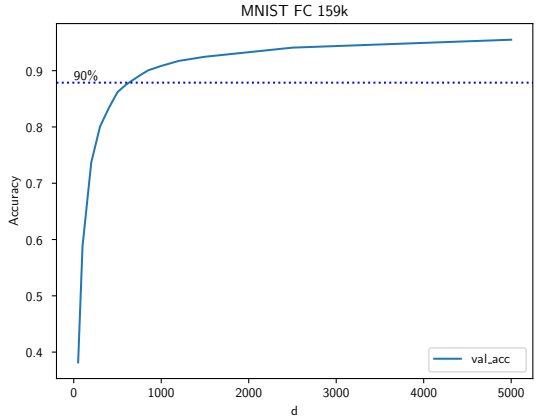

Figure 1: MLP MNIST validation accuracy depending on d.

Considering the MNIST dataset and an MLP with a single FC layer, we have: 784 dimensions for our input, 200 hidden units, 10 output dimensions, so $159,010$ parameters in total.

1.1.2 CIFAR MLP

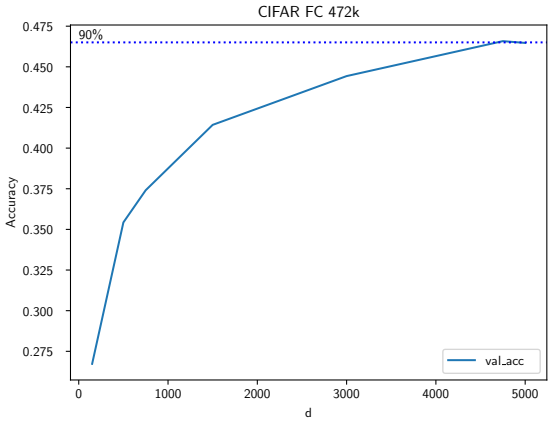

Figure 2: MLP CIFAR validation accuracy depending on d.

Considering the CIFAR-10 dataset and the MLP, we have: 784×3 dimensions for our input, 200 hidden units, 10 output dimensions, so $472,610$ parameters in total.

1.1.3 MNIST CNN

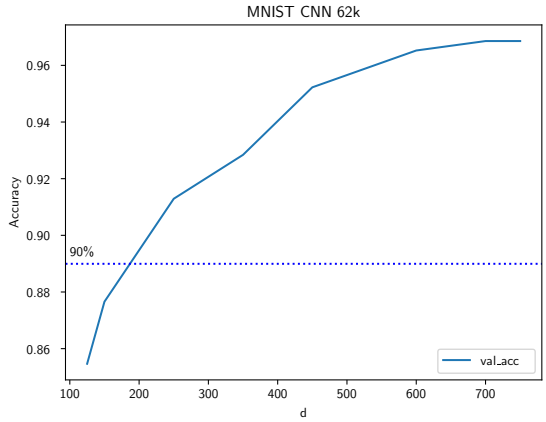

Figure 3: LeNet MNIST validation accuracy depending on d.

Considering the MNIST dataset and LeNet, we have: 784 input dims and 1 channel, 6 Kernels (5×5) – Max Pooling (2×2) – 16 Kernels (5×5) – Max Pooling (2×2) – 120 FC – 84 FC – 10 FC. So $61,706$ parameters in total.

1.1.4 CIFAR CNN

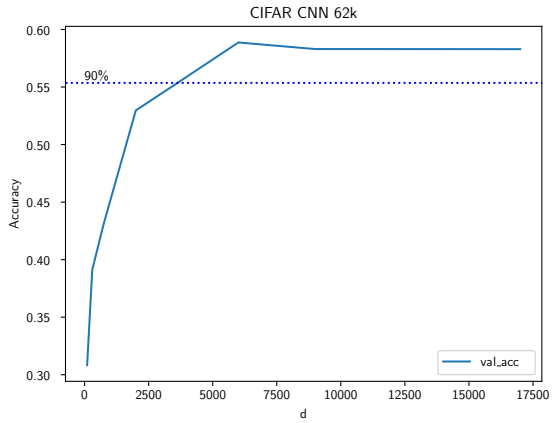

Figure 4: LeNet CIFAR validation accuracy depending on d.

Considering the CIFAR-10 dataset and LeNet, we have: 784 input dims and 3 channels, 6 Kernels (5×5) – Max Pooling (2×2) – 16 Kernels (5×5) – Max Pooling (2×2) – 120 FC – 84 FC – 10 FC. So $62,006$ parameters in total.

1.2 Sparse Matrix Implementation

Representations considered:

- Dense (sparse): used as a baseline.
- Coordinate Format (sparse_coo): lower memory usage but high computational cost.
- Compressed Sparse Row (sparse_csr): low memory usage but high computational cost.
- Compressed Sparse Column (sparse_csc): low memory usage and lower computational cost compared to sparse_csr, the best performing sparse representation for our use case.

In Fig. 5 we can see the benchmarks for multiple random projection types, all of which were tested using the model with most parameters (Section 1.1.2), and $d = 4.5$k.

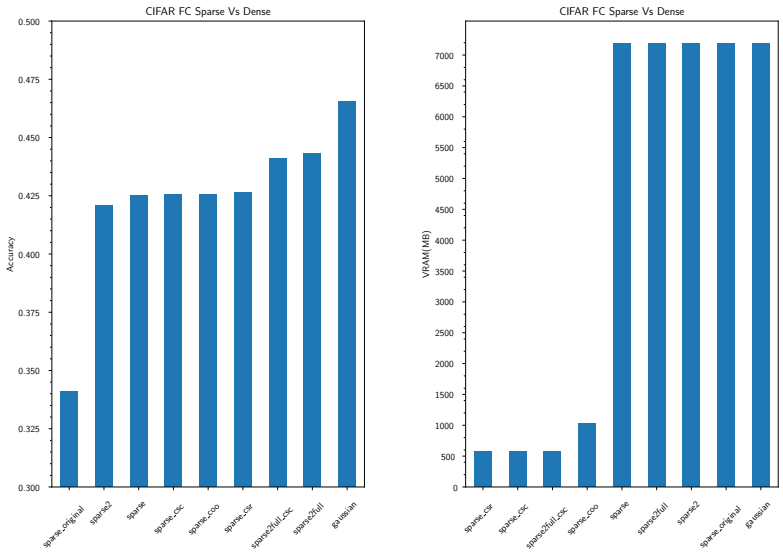

Figure 5: Accuracy and VRAM usage for MLP CIFAR

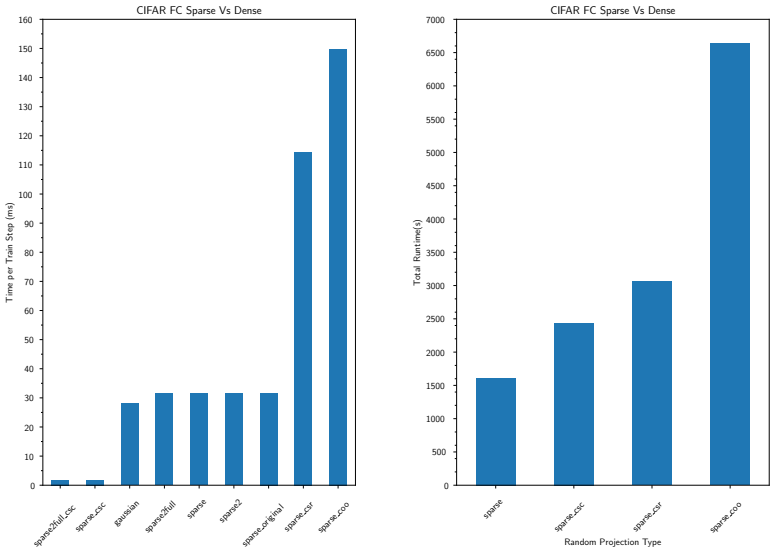

Figure 6: Train Time Step and Total Runtime for MLP CIFAR

1.3 Orthogonal Random Projection

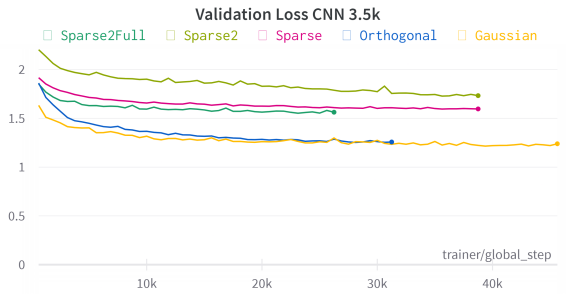

Figure 7: Validation Loss for MLP CIFAR

1.4 Smaller models

1.4.1 CNN

In the following experiments the models using projection are LeNet with a fixed architecture, same as (Section 1.1), while the smaller models are LeNets with different hyper-parameters: kernel size, output channels, stride, hidden size.

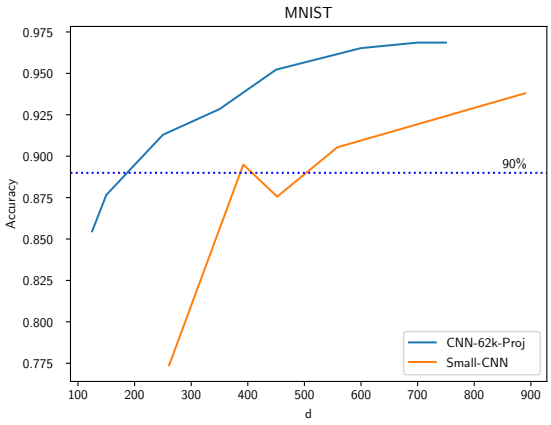

Figure 8: MNIST LeNets Small vs Projection.

The gap in the case of CNNs is lower, but it's still there. As we consider less parameters, we can see that the model using projection performs significantly better.

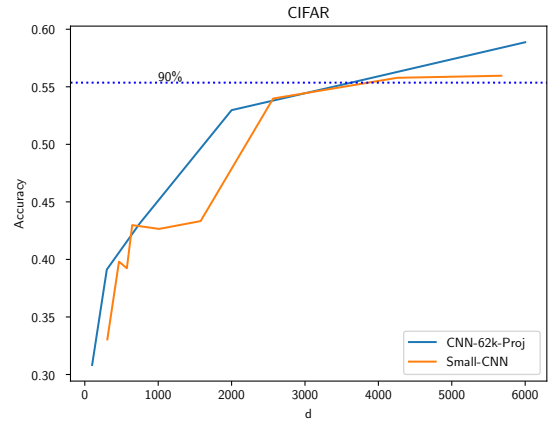

Figure 9: CIFAR LeNets Small vs Projection.

Interestingly, if we consider a more difficult task and dataset, in this case CIFAR-10, the gap pretty much closes. My hypothesis is that in this case the Fully Connected layers of the LeNet, which are used for predicting the output based on the high-level features of the images, extracted through the Convolutional layers, are actually fully exploited for this task.

2 Training Runs

All model optimizations were logged on Weights & Biases, and the data can be accessed at the following link: Weights And Biases Report. There is also an additional experiment about how changing the hyper-parameters of the projections may bring better predictions.