Heidelberg University
Institute of Computer Engineering
Computing Systems Group

Holger Fröning
Hendrik Borras

Embedded Machine Learning
Summer Term 2023

Exercise 4

- **Return electronically until Wednesday, June 7, 2023, 09:00**
- **Include your names on the top sheet. Hand in only one PDF.**
- **A maximum of nine students are allowed to work jointly on the exercises.**
- **When you include plots add a short explanation of what you expected and observed.**
- **Hand in source code if the exercise required programming. You can bundle the source code along with the PDF in a .zip file.**
- **Programming exercises can only be graded if they run on the cluster in the provided conda enviroment. Make sure to document additional steps, which you might have taken to run the exercises.**

4.1 Regularization: Dropout

On this exercise sheet we will be taking a look at regularization and data augmentation techniques to improve the training behaviour of networks without changing the architecture significantly.
In the first step we will implement a more complex convolutional network, which is more fitting for the SVHN task. This network now also strictly requires that one uses a GPU to train it, as CPU training becomes prohibitively slow.
With this network we will then explore: Dropout, L2 loss regularization and data augmentation.

- Implement the smallest version of VGG, colloquially called VGG11, as published in the original paper (ConvNet Version A in Table 1): `https://arxiv.org/pdf/1409.1556.pdf`
  - `conv3-64` Denotes a 3x3 convolution with 64 output channels and 1 pixel padding.
  - `maxpool` Denotes a maximum pooling layer with 2x2 receptive field and 2x2 stride. This effectively halves the feature map size in height and width.
  - `FC-4096` Denotes a fully connected layer with 4096 neurons.
  - Each Conv and FC layer is followed by a ReLU activation layer, except for the very last FC layer, which uses the softmax activation function.
  - Note: To run the network with the SVHN dataset instead of the ImageNet dataset the last layer will have to be adapted to the number of classes in SVHN, from the 1000 classes found in ImageNet.
  - In its original form this network does not use global average pooling.
- Note: The network is very sensitive to the batch size and LR. However, it should work out of the box with the default parameters from the template.
- Train the VGG11 network for 30 epochs and plot how the train loss, test loss and test accuracy develop over the epochs.
- Compare the training time for one epoch on the GPU compared to the CPU and discuss why this differs from the previous time CPU and GPU were compared in exercise 3.1. Note: Don't run more than one epoch on the CPU.
- Implement dropout layers for the network. They should be positioned in front of the ReLU activations. Make sure to use the dropout implementation built into PyTorch.
- Train the network with at least three different dropout values between 0.1 and 0.9. Plot how the test accuracy differs between the dropout implementations and the plain implementation. Note: You may zoom in to the region of the final accuracy to make the differences more visible.
- Discuss the results obtained in the previous plots.

(30 points)

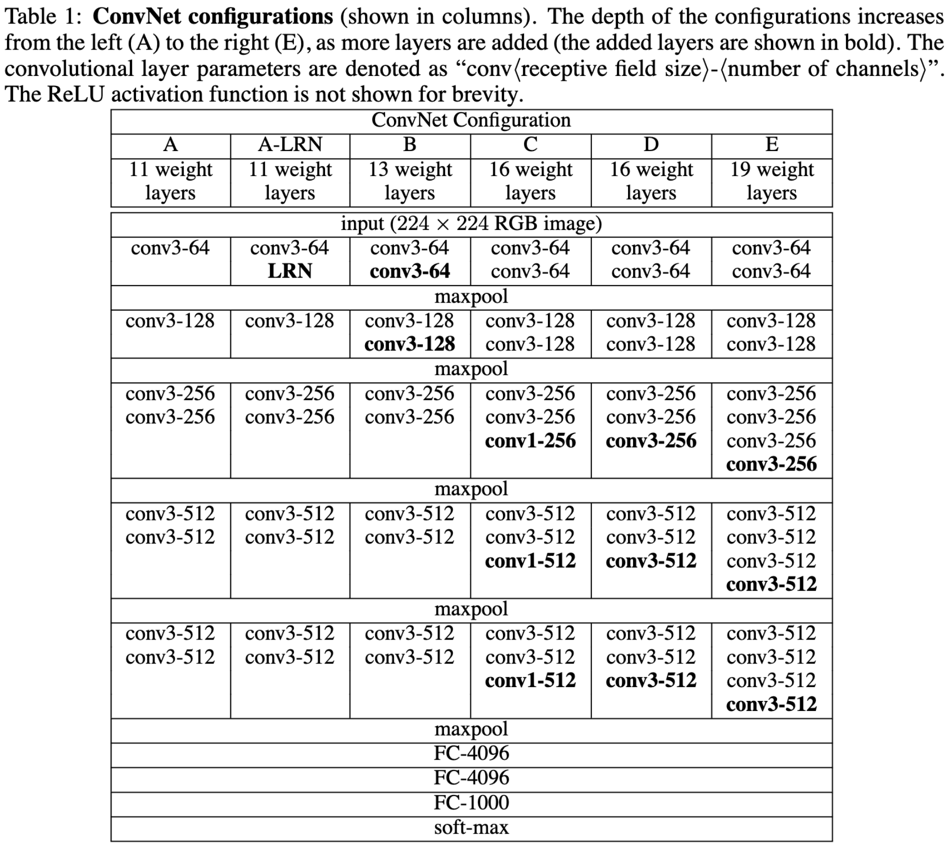

Table 1: **ConvNet configurations** (shown in columns). The depth of the configurations increases from the left (A) to the right (E), as more layers are added (the added layers are shown in bold). The convolutional layer parameters are denoted as "conv⟨receptive field size⟩-⟨number of channels⟩". The ReLU activation function is not shown for brevity.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

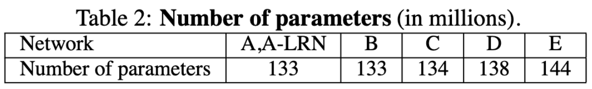

Table 2: **Number of parameters** (in millions).

Network	A,A-LRN	B	C	D	E
Number of parameters	133	133	134	138	144

4.2 L2 regularization / weight decay

In this section we will look at L2 regularization, which directly affects how weights are learned in the network.

- Implement L2 regularization, by utilizing the weight decay parameter of the ADAM optimizer.
- Vary the strength of the L2 regularization between 0.001 and $1e-06$, choose at least three measurement points.
- Plot a histogram of the weights of the last convolutional layer of the VGG at the end of the training. Compare the histograms of the weights at different L2 regularization strengths.
- Plot how the networks test accuracies develops for different regularization strengths.
- Discuss the results obtained in the previous plots.

(20 points)

4.3 Data Augmentation

Finally, we will look at data augmentation, which can be applied to the dataset using the transformations provided with torchvision[1].

- Apply different image data augmentations to the SVHN dataset (test at least three different ones or a combination thereof).
- Plot how the different transformations impact training accuracies.

[1] `https://pytorch.org/vision/stable/transforms.html`

- Plot how the different transformations impact training times.
- Discuss the results obtained in the previous plots.
- Finally, discuss the trade-offs observed between the three techniques explored in this exercise: Dropout, L2 loss regularization and data augmentation. What are their advantages/disadvantages and when should they be used?

(10 points)

4.4 Willingness to present

Please declare whether you are willing to present any of the exercises from this sheet. The declaration should happen on the PDF which you hand in.
The declaration can be made on a per-exercise basis. Each declaration corresponds to 50% of the exercise points. You can only declare your willingness to present exercises for which you also hand in a solution. If no willingness to present is declared you may still be required to present an exercise for which your group has handed in a solution. This may happen if as example nobody else has declared their willingness to present.

- Regularization: Dropout and DropBlock (Section: 4.1)
- L1 regularization and L2 regularization / weight decay (Section: 4.2)
- Data Augmentation (Section: 4.3)

(30 points)

Total: 90 points