Introduction to Deep Learning (I2DL)
Mock Exam

IN2346 - SoSe 2020

Technical University of Munich

Problem		Full Points	Your Score
1	Multiple Choice	10	
2	Short Questions	12	
3	Backpropagation	9	
Total		**31**	

Total Time: **31 Minutes**

Allowed Ressources: **None**

The purpose of this mock exam is to give you an idea of the type of problems and the structure of the final exam. The mock exam is not graded. The final exam will most probably be composed of 90 graded points with a total time of 90 minutes.

Multiple Choice Questions:

- For all multiple choice questions any number of answers, i.e. either zero (!) or one or multiple answers can be correct.
- For each question, you'll receive 2 points if all boxes are answered correctly (i.e. correct answers are checked, wrong answers are not checked) and 0 otherwise.

How to Check a Box:

- Please **cross** the respective box:  (interpreted as **checked**)
- If you change your mind, please **fill** the box:  (interpreted as **not checked**)
- If you change your mind again, please **circle** the box:  (interpreted as **checked**)

Part I: Multiple Choice (10 points)

1. (2 points) To avoid overfitting, you can...

   ☐ increase the size of the network.

   ☒ use data augmentation.

   ☐ use Xavier initialization.

   ☒ stop training earlier.

避免过拟合的常用方法有：
使用更大的训练集
使用正则化
使用 dropout
使用 early stopping
使用 k-fold cross-validation
使用数据增强
使用网络结构约束
使用更小的网络结构
使用更小的超参数
使用随机性，如随机化初始权重

2. (2 points) What is true about Dropout?

   ☐ The training process is faster and more stable to initialization when using Dropout.

   ☐ You should not use weaky ReLu as non-linearity when using Dropout.

   ☒ Dropout acts as regularization.

   ☒ Dropout is applied differently during training and testing.

3. (2 points) What is true about Batch Normalization?

   ☒ Batch Normalization uses two trainable parameters that allow the network to undo the normalization effect of this layer if needed.

   ☒ Batch Normalization makes the gradients more stable so that we can train deeper networks.

   ☒ At test time, Batch Normalization uses a mean and variance computed on training samples to normalize the data.

   ☒ Batch Normalization has learnable parameters.

调整每一层的数据分布
解决梯度消失或爆炸

4. (2 points) Which of the following optimization methods use first order momentum?

   ☐ Stochastic Gradient Descent

   ☒ Adam

   ☒ RMSProp

   ☐ Gauss-Newton x

5. (2 points) Making your network deeper by adding more parametrized layers will always...

   ☒ slow down training and inference speed.

   ☐ reduce the training loss.

   ☒ improve the performance on unseen data.

   ☒ (Optional: make your model sound cooler when bragging about it at parties.)

Part II: Short Questions (12 points)

1. (2 points) You're training a neural network and notice that the validation error is significantly lower than the training error. Name two possible reasons for this to happen.

- Data leakage: The validation data may have been used in the preprocessing of the training data, leading to a bias in the validation error.
- Data imbalance: The distribution of the classes in the training and validation sets may be different, leading to a bias in the validation error.
- Hyperparameter tuning: The model may have been fine-tuned to perform well on the validation set, leading to poor performance on unseen data.
- Randomness: The training process is random, and sometimes the validation error is just lower than the training error by chance.
- Dropout or other regularization techniques applied on training but not on validation/testing, this will make the training error larger than validation error.
- Programme error

2. (2 points) You're working for a cool tech startup that receives thousands of job applications every day, so you train a neural network to automate the entire hiring process. Your model automatically classifies resumes of candidates, and rejects or sends job offers to all candidates accordingly. Which of the following measures is more important for your model? Explain.

Recall $= \frac{\textit{True Positives}}{\textit{Total Positive Samples}}$

Precision $= \frac{\textit{True Positives}}{\textit{Total Predicted Positive Samples}}$

Precision is more important

NN is working for classify unseen job application.

So the

3. (2 points) You're training a neural network for image classification with a very large dataset. Your friend who studies mathematics suggests: "If you would use Newton-Method for optimization, your neural network would converge much faster than with gradient descent!". Explain whether this statement is true (1p) and discuss potential downsides of following his suggestion (1p).

That's true
Netwon-Method is a second-order optimization method which use Hessian Matrix instead of learning rate to update the weight
⇒ + better to find the optimum beacuse of the second-order deritaie Hessian matrix
− Compute Hessian Matrix (second-order deritave and inverse) are compute costly

4. (2 points) Your colleague trained a neural network using standard stochastic gradient descent and L2 weight regularization with four different learning rates (shown below) and plotted the corresponding loss curves (also shown shown below). Unfortunately he forgot which curve belongs to which learning rate. Please assign each of the learning rate values below to the curve (A/B/C/D) it probably belongs to and explain your thoughts.

```
learning_rates = [3e-4,  4e-1,  2e-5,  8e-3]
```

B D   C A   A B   D C

5. (1 point) Explain why we need activation functions.

Usually the Network is linear classif

Because the Linear Model can only handle linear data

But most data are non-linearity

Add non-linearity to the NN, in case gradient vanish when back propaga..

6. (3 points) When implementing a neural network layer from scratch, we usually implement a 'forward' and a 'backward' function for each layer. Explain what these functions do, potential variables that they need to save, which arguments they take, and what they return.

$\sum wx = w_1 x_1 + w_2 x_2 + w_3 x_3 + \cdots w_n x_n$

Forward, a neuron sum all the input multiple weight, and activation func will add non-linearity to the layer via $f(\Sigma)$, then these output become the input of next layer

Back propagation via Backprop NN can update the weight via chainrule in Backp

$$\frac{\partial L}{\partial f} \cdot \frac{\partial f}{\partial z} \cdot \frac{\partial z}{\partial x}$$

L

Forward:
- Accept the output from the previous layer and do some operation, return result
- cache values which will be used in Back propagation

Backward:
- take upstream gradient, return partial derivaties

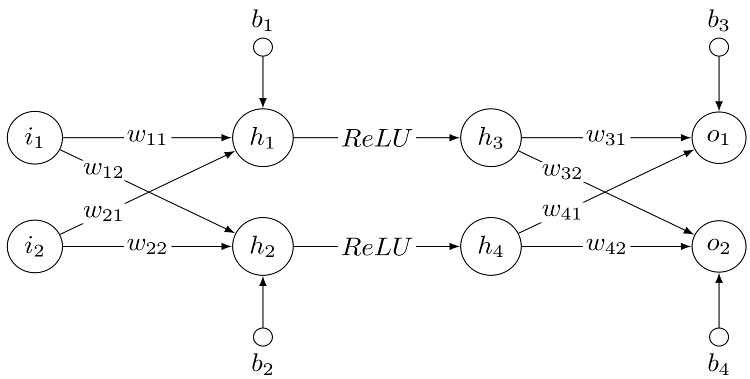
7. (0 points) Optional: Given a Convolution Layer with 8 filters, a filter size of 6, a stride of 2, and a padding of 1. For an input feature map of $32 \times 32 \times 32$, what is the output dimensionality after applying the Convolution Layer to the input?

$$\frac{N+2P-F}{S} + 1 = \frac{32+2\times1-6}{2} + 1 = 15$$

output: $15 \times 15 \times 8$

Part III: Backpropagation (9 points)

1. (9 points) Given the following neural network with fully connection layer and ReLU activations, including two input units (i_1, i_2), four hidden units (h_1, h_2) and (h_3, h_4). The output units are indicated as (o_1, o_2) and their targets are indicated as (t_1, t_2). The weights and bias of fully connected layer are called w and b with specific sub-descriptors.



The values of variables are given in the following table:

Variable	i_1	i_2	w_{11}	w_{12}	w_{21}	w_{22}	w_{31}	w_{32}	w_{41}	w_{42}	b_1	b_2	b_3	b_4	t_1	t_2
Value	2.0	-1.0	1.0	-0.5	0.5	-1.0	0.5	-1.0	-0.5	1.0	0.5	-0.5	-1.0	0.5	1.0	0.5

(a) (3 points) Compute the output (o_1, o_2) with the input (i_1, i_2) and network paramters as specified above. Write down all calculations, including intermediate layer results.

$h_1 = w_{11} \cdot i_1 + w_{21} \cdot i_2 + b_1 = 1.0 \times 2.0 + 0.5 \times (-1.0) + 0.5 = 2.0$

$h_2 = w_{22} \cdot i_2 + w_{12} \cdot i_1 + b_2 = (-1.0) \times (-1.0) + (-0.5) \times (2.0) + (-0.5) = -0.5$

$h_3 = relu(h_1) = 2.0$

$h_4 = relu(h_2) = 0.0$

$o_1 = w_{31} \cdot h_3 + w_{41} \cdot h_4 + b_3 = 0.5 \times 2.0 + (-0.5) \times 0.0 + (-1.0) = 0$

$o_2 = w_{32} \cdot h_3 + w_{42} \cdot h_4 + b_4 = (-1.0) \times 2.0 + (1.0) \times 0.0 + (0.5) = -1.5$

(b) (1 point) Compute the mean squared error of the output (o_1, o_2) calculated above and the target (t_1, t_2).

$$MSE = \frac{1}{n} \sum_{i=1}^{N} \| y_i - \hat{y}_i \|_2^2 = \frac{1}{2}\left((t_1 - o_1)^2 + (t_2 - o_2)^2 \right)$$
$$= \frac{1}{2}(1 + 4) = 2.5$$

(c) (5 points) Update the weight w_{21} using gradient descent with learning rate 0.1 as well as the loss computed previously. (Please write down all your computations.)

$$\frac{\partial ReLu(x)}{\partial x} = \begin{cases} 1, & x > 0 \\ 0, & x \le 0 \end{cases}$$

$$W_{i+1} = W_i - \alpha \nabla_w L(w)$$

$$\frac{\partial MSE}{\partial W_{21}} = \frac{\partial \frac{1}{2}(t_1 - o_1)^2}{\partial o_1} \cdot \frac{\partial o_1}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_1} \cdot \frac{\partial h_1}{\partial W_{21}} + \frac{\partial \frac{1}{2}(t_2 - o_2)^2}{\partial o_2} \cdot \frac{\partial o_2}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_1} \cdot \frac{\partial h_1}{\partial W_{21}}$$

$$= (t_1 - o_1) \cdot (-1) \cdot W_{31} \cdot 1 \cdot i_2 + (t_2 - o_2) \cdot (-1) \cdot W_{32} \cdot 1 \cdot i_2$$

$$= 1 \cdot -1 \cdot 0.5 \cdot 1 \cdot (-1.0) + (2.0)(-1) \cdot (-1.0) \cdot 1 \cdot (-1.0)$$

$$= 0.5 - 2$$

$$= -1.5$$

$$W_{21} = W_{21} - (0.1) \times (-1.5)$$

$$= 0.5 + 0.15$$

$$= 0.65$$

Additional Space for solutions. Clearly mark the problem your answers are related to and strike out invalid solutions.