Chair of Visual Computing
Department of Informatics
Technical University of Munich

Esolution
Place student sticker here

Note:

- During the attendance check a sticker containing a unique code will be put on this exam.
- This code contains a unique number that associates this exam with your registration number.
- This number is printed both next to the code and to the signature field in the attendance check list.

Introduction to Deep Learning

Exam:	IN2346 / endterm	**Date:**	Tuesday 8th February, 2022
Examiner:	Prof. Dr. Matthias Nießner	**Time:**	15:00 – 11:30

- The blackened exam has the same layout as the non-blackened exam with the acutal questions, which is going to be released once the working time starts.
- Only submit your personalized blackened exam. **DO NOT submit the non-blackened/non-personalized exam** (clearly indicated with “DO NOT SCAN/UPLOAD”).
- This final exam consists of **16 pages** with a total of **7 problems**.
  Please make sure now that you received a complete copy of the exam.
- The total amount of achievable credits in this simulation is **90 credits**.
- No additional resources are allowed.

Problem 1 Multiple Choice (18 credits)

Mark correct answers with a cross ☒
To undo a cross, completely fill out the answer option ■
To re-mark an option, use a human-readable marking ×■

Please note:

- For all multiple choice questions any number of answers, i.e. either zero (!), 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.

1.1 You are training a network to classify images of handwritten digits in the range of [0,...,9] on the MNIST dataset. Which of the following data augmentation techniques are suitable to use for this task?

- ☒ Add Gaussian noise to the images
- ☐ Vertically flip the images
- ☒ Rotation of the images by 10 degrees
- ☒ Change the contrast of the images

1.2 What is true about Residual Blocks?

- ☐ Reduce the number of computations in the forward pass
- ☒ Act as a highway for gradient flow
- ☒ Enable a more stable training of larger networks
- ☐ Act as a regularizer

1.3 For a fully-convolutional 2D CNN, if we double the spatial dimensions of input images, ...

- ☐ ... the number of network parameters doubles
- ☒ ... the number of network parameters stays the same
- ☐ ... the receptive field of an arbitrary pixel in an intermediate activation map can decrease
- ☐ ... the dropout coefficient p must be corrected to $\sqrt{p}$ in test time

1.4 What is true about Generative Adversarial Networks?

- ☒ The Generator minimizes the probability that the Discriminator is correct
- ☐ The Generator provides supervision for the Discriminator
- ☒ The Discriminator acts as a classifier
- ☐ The Discriminator samples from a latent space

1.5 Given input x , which of the following statements are always true? Note: For dropout, assume the same set of neurons are chosen.

- ☐ BatchNorm(ReLU(x)) $\equiv$ ReLU(BatchNorm(x))
- ☒ Dropout(ReLU(x)) $\equiv$ ReLU(Dropout(x))
- ☒ MaxPool(ReLU(x)) $\equiv$ ReLU(MaxPool(x))
- ☒ ReLU(Sigmoid(x)) $\equiv$ Sigmoid(ReLU(x))

1.6 When you are using a deep CNN to train a semantic segmentation model, which of the following can be chosen to help with overfitting issues?

- ☐ Decrease the weight decay parameter
- ☒ Increase the probability of switching off neurons in dropout
- ☒ Apply random Gaussian noise to the input images
- ☐ Remove parts of the validation set

1.7 In terms of (full-batch) gradient descent (GD) and (mini-batch) stochastic gradient descent (SGD), which of the following statements are true?

- ☐ The computed gradient of the loss w.r.t model parameters in SGD is equal to the computed gradient in GD
- ☒ The expected gradient of the loss w.r.t model parameters in SGD is equal to the expected gradient in GD
- ☒ There exists some batch size, for which the gradient of the loss w.r.t model parameters in SGD is equal to the gradient in GD
- ☐ SGD and GD will converge to the same model parameters, but SGD requires less memory at the expense of more iterations

1.8 What is true about batch normalization assuming your train and test set are sampled from the same distribution?

- ☐ Batch normalization cannot not be used together with dropout
- ☒ Batch normalization makes the gradients more stable, so we can train deeper networks
- ☐ At test time, Batch normalization uses a mean and variance computed on test set samples to normalize the data
- ☒ Batch normalization has learnable parameters

1.9 What is true for common architectures like VGG-16 or LeNet? (check all that apply)

- ☒ The number of filters tends to increase as we go deeper into the network
- ☐ The width and height of the activation maps tends to increase as we go deeper into the network
- ☐ The input can be an image of any size as long as its width and height are equal
- ☒ They follow the paradigm: Conv $\rightarrow$ Pool ... $\rightarrow$ Conv $\rightarrow$ Pool $\rightarrow$ FC ... $\rightarrow$ FC

  (Conv = Conv + activation)

Problem 2 Short Questions (18 credits)

0 ☒ 1 ☐ 2 ☐

2.1 In k -fold cross validation, choosing a larger value for k increases our confidence in the validation score. What could be a practical disadvantage in doing so? Explain how it arises.

If k is equal to the number of input data, we will just memorize the input data. We will not learn anything, it will just overfit. (no generalizes)

Increases training time or more computations. Making use of more folds, will present the model with more data to train on, but will require way more time as it has to train and validate k separate times

0 ☒ 1 ☐ 2 ☐ 3 ☐ 4 ☐

2.2 Consider the activation function $f : \mathbb{R} \to \mathbb{R}$ and $f(x) = \ln(1 + e^x)$. Which one of the following activation functions is most closely approximated by f ? Briefly justify your answer (2 points). What is the benefit of f over the activation function it closely approximates (2 points)?

- Tanh
- ReLU
- Sigmoid

Sigmoid. → $\text{sigm}(x) = \frac{1}{1+e^{-x}}$ ✗

We can express the gradient as a function of itself $y = \sigma(x)$, $y' = y(1-y)$ ✗

ReLU. ReLU is the only function which is unbounded up

Unlike ReLU, this function is smooth everywhere in $\mathbb{R}$, so its differentiable everywhere in $\mathbb{R}$. ReLU is not differentiable at 0

0 ☐ 1 ☒ 2 ☐

2.3 Explain the difference between the validation set and the test set. In your answer, explain the role of each subset and how they are used differently.

Validation set is used for testing generalization with different hyperparameters.

Test set is only used at the end ! Not touched during training. to test generalization on unseen data once.

0 ☐ 1 ☐ 2 ☒

2.4 You notice vanishing/exploding gradients in a deep network using the tanh activation function. Suggest two possible changes you can make to the network in order to diminish this issue, without changing the number of trainable parameters. Explain how each of these changes helps.

Changing the activation function to a Leaky ReLU, it does not saturate ReLU activation → does not saturate, large consistent gradients

Or with Xavier Initialization → improved weight initialization targets active area of the activation function

2.5 Can two consecutive dropout layers with probabilities q and p be replaced with one dropout operation? Explain.

0 1 2 — 2

Yes. We just have to add a dropout with q·p probability

Neurons zero out independently, so dropout layer with propability p+q-2pq. If dropout layer one already zeros out neurons those can't be considered by second layer anymore. Dropping probability of pq

2.6 Can one encounter overfitting in an unsupervised learning setting? If your answer is *no*, provide a mathematical reasoning. If your answer is *yes*, provide an example.

0 1 2 3 — 1

Yes. (Kmeans with K= N)

Example → clustering N datapoints with N clusters, PCA with many components, autoencoder with large bottleneck lowerfitted on one image

2.7 For each of the following functions, describe one common problem when choosing them as the activation function for your deep neural network: (a) Sigmoid, (b) ReLU, (c) Identity

0 1 2 3 — 3

Sigmoid → Saturation // non-zero centered

ReLU → dead values for values < 0

Identity → ~~Learning the same~~ does not introduce non-linearity

Problem 3 **Autoencoder (11 credits)**

Consider a given **unlabeled** image dataset consisting of 10 distinct classes of animals.

0 [x] 1 [ ] 2 [ ]

3.1 To train an Autoencoder on images, which type of losses you would use? Name two suitable losses.

~~Cross Entropy~~

~~Hinge Loss~~

Image reconstruction losses (L1, L2, MSE...)

0 [ ] 1 [ ] 2 [x]

3.2 Explain the effect of choosing a bottleneck dimension which is too small, and the effect of a too large bottleneck dimension in Autoencoders.

Too small → we will lose features of the input data, it will be too compressed (Poor reconstruction, underfitting)

Too large → it will overfit, it will not be compressed

0 [ ] 1 [x] 2 [ ]

3.3 Having trained an Autoencoder on this dataset, how would you use the trained Autoencoder (without further training/fine-tuning) to partition the dataset into 10 subsets, where each subset consists only of images of a distinct type of animal?

to get latent embedding for each unlabeled image

Using the Encoder to extract the features / ~~and replacing the decoder with a classifier that outputs the class of the image~~ Do clustering (K-means with K=10), assign each cluster to it's cluster image

3.4 We want to use the same network architecture for de-noising and colorizing old, degraded, gray-scale images of animals. Given the dataset you already have, explain the steps you would take to train your model. In your answer, elaborate on your model's inputs, outputs, and losses.

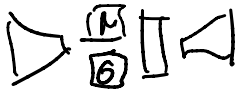
0 1 2 3

Augment input image by adding noise.
Transform input images by converting to grayscale
Use original image as target, use L1 / L2 as loss

3.5 Explain the differences between Autoencoders and Variational Autoencoders. How do they differ during training?

0 1 2

Autoencoder → the decoder just compress the input and the decoder uncompress it. The Variational Autoencoder learns a gaussian distribution in the latent space to recreate new images



Having a constraint in the latent space

Variational Autoencoders will constrain the bottleneck distribution into a probability distribution but encoders don't constrain the latent space

Problem 4 CNNs (10 credits)

You are given the following network that classifies RGB images into one of 4 classes.
All Conv2d layers use `kernel` = 3 , `padding` = 1 , `stride` = 1 , `bias` = `True` and are defined as `Conv2d`(< $\texttt{channels}_{\text{in}}$ >, < $\texttt{channels}_{\text{out}}$ >) .
All MaxPool2d layers use `stride` = 2 , `padding` = 0 , and are defined as `MaxPool`(< `kernel` >) .
The input dimension x of the Linear layer is unknown.
The network's architecture is as follows:

- Conv2d(3, 8) → MaxPool2d(2) → BatchNorm2d() → ReLU() →
- Conv2d(8, 16) → MaxPool2d(2) → BatchNorm2d() → ReLU() →
- Conv2d(16, 32) → MaxPool2d(2) → BatchNorm2d() → ReLU() →
- Flatten() →
- Linear(x , 4) → Softmax()

0 1 2 ☒

4.1 In terms of x , what is the total number of trainable parameters of the last linear layer? Include a bias term in your calculation.

4x+4 ✓

0 1 ☒ 2

4.2 Given RGB input images of size 80×80 pixels, what should the value of x in the Linear layer be? Explain your calculation.

Input: 80×80×3 → $\left\lfloor \frac{80+2(1)-3}{1} \right\rfloor + 1 = 80 \times 80 \times 8$ →

$\frac{80-2}{2} + 1 = 40 \times 40 \times 8 =$ $40+2(1)-3+1 = 40 \times 40 \times 16$

$20 \times 20 \times 16 = 20 \times 20 \times 32 = 10 \times 10 \times 32$

x → 10 × 10 × 32 = 3200

Each conv2d preserves spatial dimensions. Each maxpool reduces spatial dimensions by 2. Height and width for the shape 80/2/2/2=10 at linear layer. Depth is 32 as given by final conv2d.

4.3 Explain the main difference between the usage of a BatchNorm layer in a convolutional network in comparison to a fully connected network.

0 1 2 ✗

In a convolutional is it apply in different channels and in a fully connected network is applied to features weights

Conv → Normalization acts on channel dimension instead of per feature normalization

4.4 Compute the total number of trainable parameters of the first convolutional layer, Conv2d(3,8).

0 ✗ 1 2

$80 \times 80 \times 8 \times 3 = 6400 \times 24 = 153600$

64
×24
256
128
1536

$3 \times 3 \times 3 \times 8 + 8 = 216 + 8 = 224$

Bias = # channels

4.5 Compute the total number of trainable parameters in all of the BatchNorm layers.

0 1 2 ✗

2 · Channels → $2 \cdot 8 = 16$
32
64
} 112

Each BatchNorm2d layer has two weights per channel

Problem 5 Optimization and Gradients (16 credits)

You are training a large fully-connected neural network and select as an initial choice an SGD optimizer. In order to overcome the limitations of SGD, your colleague suggests adding momentum.

0 1 2 3

5.1 Name two limitations of SGD that momentum can potentially solve. Explain how momentum solves them.

- Overcome latent space / local minima
- Faster

keeps direction of gradient to get out of local minimum

1) Slow learning / small steps, can't escape local minima, SGD is noisy,
2) Speeds up learning if gradient keeps pointing in the same direction

0 1 2

5.2 One can apply momentum, as shown in the formula:

$$\nu^{k+1} = \beta \cdot \nu^k - \alpha \cdot \nabla_\theta L(\theta^k)$$

What do the hyperparameters α and β represent?

β → velocity (accumulation rate of velocity)

α → learning rate

0 1 2

5.3 How does Nesterov Momentum differ from standard momentum? Explain.

Gradient term computed from position calculated with previous gradient (look ahead step).

Gradient corrects potential overshooting of momentum already in the same step

0 1 2 3

5.4 Is RMSProp considered a first or second order method (1p)? What is the main difference between RMSProp and SGD+Momentum?

First

RMSProp dampens oscillation (Uses second moment)

SGD + Momentum accumulates gradients (first moment)

For the following questions, consider the convex optimization objective:

$$\min_{x \in \mathbb{R}} x^2$$

5.5 What is the optimal solution of this optimization problem?

0 / 1

0

5.6 You are working with an initialization of $x_0 = 5$ and a learning rate of `lr` = 1 . How many iterations would gradient descent (without momentum) need in order to converge to the optimal solution? Explain.

0 / 1

$2x = 0 \rightarrow 2 \cdot 5 = 10$ It will overshoot.

$5 - 1(10) = -5$ Never reaches the min.

$2 \cdot -5 = -10 \quad -5 - 1(-10) = -10$

5.7 Assuming you instead start with a random initialization of x_0 , how could you speed up the convergence of the gradient descent optimizer (without adding momentum) in this case?

0 / 1

↓ lr (Reduce learning rate)

5.8 What is the main advantage of using a second order method such as Newton's Method? Why are second order methods not used often in practice for training deep neural networks?

0 / 1 / 2

Faster. Complex to calculate

Less iterations → converges faster

Hessian costly to compute

5.9 How many iterations would Newton's method need to converge (using the same initialization $x_0 = 5$, `lr` = 1)? Explain.

0 / 1

1. Jumps to minimum right away

Problem 6 Derivatives (9 credits)

Consider the formula of the Sigmoid function $\sigma(x) : \mathbb{R} \to \mathbb{R}$:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

0 1 ☒

6.1 Compute the derivative $\frac{d\sigma(x)}{dx}$ in terms of x .

✓

$$\frac{d\sigma(x)}{dx} = \frac{0 \cdot (1 + e^{-x}) - 1 \cdot (-e^{-x})}{(1 + e^{-x})^2} = \frac{e^{-x}}{(1 + e^{-x})^2}$$

0 1 ☒

6.2 A special property of this function is that its derivative can be expressed in terms of the Sigmoid function itself. Denote $y = \sigma(x)$, and show how the derivative you computed can be re-written in terms of y , the output of the Sigmoid function. *Hint:* Your answer should only depend on y .

✓

$$y = \sigma(x) = y(1 - y)$$

$$\frac{e^{-x}}{(1 + e^{-x})^2} = \frac{e^{-x}}{(1 + e^{-x})} \cdot \frac{1}{(1 + e^{-x})} = y(1 - y)$$

An affine Layer is described by $\mathbf{z} = XW + b$.
Consider the following affine layer, which has 2 input neurons and 1 output neuron:

$$W = \begin{bmatrix} 1 \\ 2 \end{bmatrix}_{2\times 1}$$

$$b = 2 \in R^1$$

and input:

$$X = \begin{bmatrix} 1 & 1 \\ 0 & -1 \end{bmatrix}_{2\times 2}$$

The forward pass of the network would be:

$$\sigma(\mathbf{z}) = \sigma(XW + b) = \sigma(\begin{bmatrix} 1 & 1 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \end{bmatrix} + 2) = \sigma(\begin{bmatrix} 3 \\ -2 \end{bmatrix} + \begin{bmatrix} 2 \\ 2 \end{bmatrix}) = \sigma(\begin{bmatrix} 5 \\ 0 \end{bmatrix}) = \begin{bmatrix} 1 \\ 0.5 \end{bmatrix} \text{ (rounded up).}$$

Let's compute the backward pass of the network.

6.3 If $\mathbf{y} = \sigma(\mathbf{z}) = \begin{bmatrix} 1 \\ 0.5 \end{bmatrix}$, calculate the gradient of the output after the Sigmoid activation function w.r.t $\mathbf{z}$, $\frac{d\mathbf{y}}{d\mathbf{z}}$:

0
1
2
3

$\frac{dy}{dz} = \frac{\sigma(z)}{dz} = z(1-z) = \begin{bmatrix} 1 \\ 0.5 \end{bmatrix} \begin{bmatrix} 0 & 0.5 \end{bmatrix} = \begin{bmatrix} 0 \\ 0.25 \end{bmatrix}$

$y \circ (1-y) = \begin{bmatrix} 1 \\ 0.5 \end{bmatrix} \circ 1 - \begin{bmatrix} 1 \\ 0.5 \end{bmatrix} = \begin{bmatrix} 0 \\ 0.25 \end{bmatrix}$

6.4 We will use the computed gradient to perform back-propagation through the affine layer to the network's parameters.
Let *dout* be the upstream derivative of the Sigmoid that you have calculated in question 3. Calculate the derivatives $\frac{d\mathbf{y}}{dW}$ and $\frac{d\mathbf{y}}{db}$.
Hint: Pay attention to the shapes of the results; they should be compatible for a gradient update.
Note: In case you skipped the previous question, you can get partial points by writing the correct formulas using *dout* symbolically.

0
1
2
3
4

$\frac{dy}{dw} = \frac{dy}{dz} \cdot \frac{dz}{dw} = \begin{bmatrix} 0 \\ +0.25 \end{bmatrix} \cdot X = \begin{bmatrix} 0 \\ +0.25 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 0 & -1 \end{bmatrix} = \begin{bmatrix} 0 \\ -0.25 \end{bmatrix}$

$\begin{bmatrix} 0 & +0.25 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 0 & -1 \end{bmatrix} = \begin{bmatrix} 0 & -0.25 \end{bmatrix}$

$\frac{dy}{db} = \frac{dy}{dz} \cdot \frac{dz}{db} = \begin{bmatrix} 0 \\ +0.25 \end{bmatrix} \cdot 1 = \begin{bmatrix} 0 \\ 0.25 \end{bmatrix} = 0.25$

$\begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0.25 \end{bmatrix} = 0.25$

Problem 7 **Model Evaluation (8 credits)**

Two students, *Erika* and *Max* train a neural network for the task of image classification. They use a dataset which is divided into train and validation sets. They each train their own network for 25 epochs.

0 1 2

7.1 Erika selects a model and obtains the following curves. Interpret the model's behaviour from the curves. Then, suggest what could Erika do in order to improve its performance?

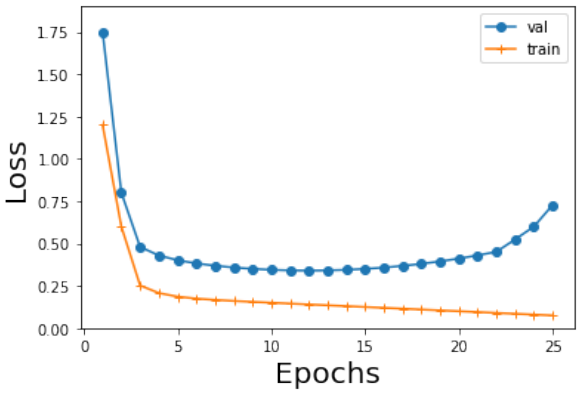


Figure 7.1: Training curves for Erika's model.

Overfitting

Needs more generalization (drop out/ data augmentation)

0 1 2

7.2 *Max* selects a different model and obtains the following curves. Interpret the model's behaviour from the curves. Then, suggest what change could Max make to his model in order to improve its performance?

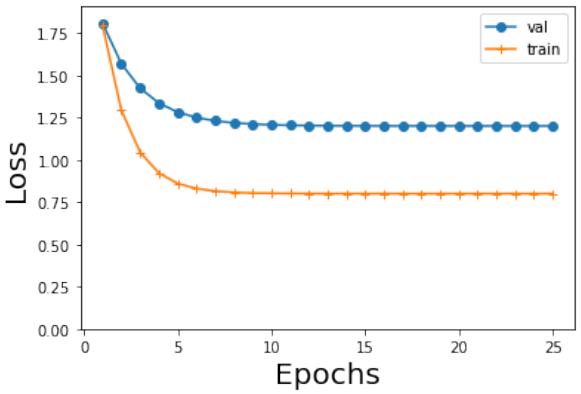


Figure 7.2: Training curves for Max's model.

Underfitting, increase model capacity

7.3 Both Max and Erika are able to agree on a model architecture and obtain the following curves. However, when deployed in real world, their model seems to perform poorly. What is a possible reason for such an observation and what should they do?

0 1 2

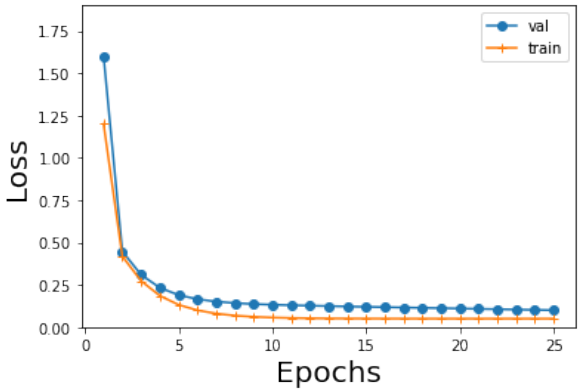


Figure 7.3: Training curves for the new model.

Test and train/val data is sampled from different distribution! domain gap
Fix by trying to make test and train data more similar or from same distribution

After adapting the new network architecture, Max and Erika are training their own model, using the same architecture, with identical initial weights, using exactly the same hyperparameters. They also use the same SGD optimizer (no momentum), batch size, and learning rates. The only difference is that Max normalizes the loss by $1/N$ (where N is the number of training samples in the dataset) while Erika does not.

7.4 How does this affect the optimal model weights that minimize this optimization objective? (1p) After 10 optimizer steps, will they arrive at the same model parameters? Explain.(2p)

0 1 2 3

It doesn't affect the optimal model's optima. The weights will be different after 10 steps.

Additional space for solutions–clearly mark the (sub)problem your answers are related to and strike out invalid solutions.