UNIVERSITY OF OSLO

Faculty of mathematics and natural sciences

Exam in: INF5860/INF9860 — Machine Learning for Image Analysis

Day of examination: 11th June 2018

Examination hours: 14:30 – 18:30

This exercise set consists of 7 pages.

Appendices: None

Permitted aids: Certified calculator

Read the entire exercise text before you start solving the exercises. Please check that the exam paper is complete. If you lack information in the exam text or think that some information is missing, you may make your own assumptions, as long as they are not contradictory to the "spirit" of the exercise. In such a case, you should make it clear what assumptions you have made.

You should spend your time in such a manner that you get to answer all exercises shortly. If you get stuck on one question, move on to the next question.

Your answers should be short, typically a few sentences and / or a sketch should be sufficient.

Every subtask has equal weight in the evaluation.

(Continued on page 2.)

Exercise 1 Dense neural networks

Suppose we have a small dense neural network as is shown in fig. 1. The

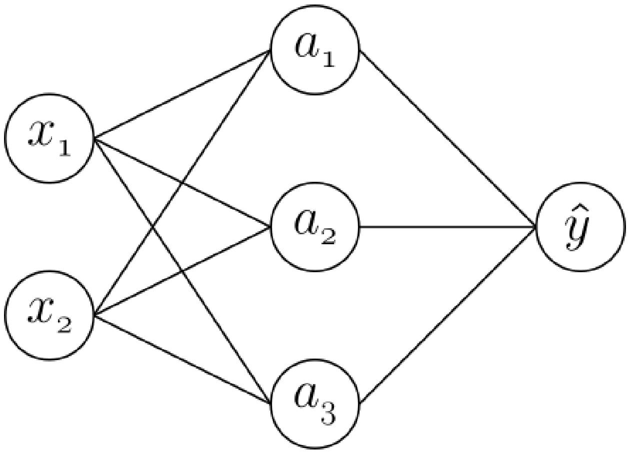

Figure 1: A small dense neural network

input vector $x = [x_1, x_2]^\intercal$ and the associated ground truth y, are

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 3 \end{pmatrix}, \quad y = 32.$$

In the first layer we have the following weight and bias parameters

$$\begin{pmatrix} w_{11}^{[1]} & w_{12}^{[1]} & w_{13}^{[1]} \\ w_{21}^{[1]} & w_{22}^{[1]} & w_{23}^{[1]} \end{pmatrix} = \begin{pmatrix} 2 & 1 & 3 \\ 2 & -1 & 1 \end{pmatrix}, \quad \begin{pmatrix} b_1^{[1]} \\ b_2^{[1]} \\ b_3^{[1]} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix}.$$

In the second layer we have the following weight and bias parameters

$$\begin{pmatrix} w_{11}^{[2]} \\ w_{21}^{[2]} \\ w_{31}^{[2]} \end{pmatrix} = \begin{pmatrix} 3 \\ 1 \\ 2 \end{pmatrix}, \quad b_1^{[2]} = 1.$$

For clarity,

- $w_{jk}^{[l]}$ is the value of the weight parameter from node j in layer $l-1$ to node k in layer l.
- $b_k^{[l]}$ is the value of the bias parameter at node k in layer l.

(Continued on page 3.)

1a

Compute the value of the network output, $\hat{y}$, when the activation functions in the first and second layer are identity functions.

1b

Compute the value of the network output, $\hat{y}$, when the activation functions in the first and the second layer are ReLU functions.

1c

Given a squared error loss function $J = (\hat{y} - y)^2$, and *identity functions* as activations in all nodes (as in task **1a** above), compute the value of the following four partial derivatives

$$\frac{\partial J}{\partial b_1^{[2]}},$$
$$\frac{\partial J}{\partial w_{21}^{[2]}},$$
$$\frac{\partial J}{\partial b_2^{[1]}},$$
$$\frac{\partial J}{\partial w_{13}^{[1]}}.$$

1d

Given everything as in task **1c**, perform one step with gradient descent with learning rate 2, and compute the updated values of the two parameters

$$b_2^{[1]},$$
$$w_{13}^{[1]}.$$

(Continued on page 4.)

Exercise 2 Convolutional neural networks

2a

You are given an input image (fig. 2a) and a kernel (fig. 2b). Your task is to evaluate the shaded pixel in the image after the convolution. The origin of the kernel is the shaded pixel. Assume we use zero padding.

3	4	5	1	0
2	0	7	3	4
1	3	1	0	8
5	5	3	8	1
1	2	5	9	9

(a) Input image

1	5	1
2	3	2
1	5	1

(b) Kernel

Figure 2

2b

Explain the dimensions of the filter bank (learnable convolution kernel weights) in a typical convolutional layer.

2c

What is the receptive field after the following layers?

Layer #	filter size	stride
Layer 1	3×3	1
Layer 2	5×5	1
Layer 3	2×2	2
Layer 4	3×3	1

(Continued on page 5.)

2d

Given two models:

- Model A has a single 7×7 convolutional layer.
- Model B has multiple subsequent 3×3 convolutional layers. The number of layers is such that the size of the receptive field is the same as in model A.

What is the ratio of the number of parameters involved in the two different models? Assume that the number of input and output feature maps are equal in the two models (and for all layers). Ignore bias parameters and use stride = 1.

Exercise 3 Generalization

3a

Explain briefly the effect of the regularization loss on the hypothesis set.

3b

Splitting the available dataset is a common way to estimate the out-of-sample error. If you check multiple models on the test set and select the best performing model, will this be a good indicator of the out-of-sample error? Justify your answer.

Exercise 4 Object localization

You are to design an image classifier that in addition predicts where in the image the (single) object of interest is. You can assume that there is only one interesting object in the image, if any.

4a

Explain briefly how you would set up the target vector, y, (and the output vector), in this case.

4b

Explain briefly how an associated loss function could look like.

(Continued on page 6.)

Exercise 5 Recurrent neural networks

Recurrent neural networks are powerful models for processing sequential data.

5a

Show and describe the most general recurrence formula for a recurrent neural network.

5b

Why are long range dependencies difficult to learn in a recurrent neural network?

5c

Why is Gated Recurrent Units (GRU) more efficient in preserving long range dependencies than vanilla RNNs?

5d

What is the advantage and disadvantage with using Truncated Back-propagation Through Time (TBTT)?

5e

Explain *briefly* how a convolutional neural network and a recurrent neural network can be used to generate descriptive natural language from an image?

Exercise 6 Unsupervised learning

6a

Explain briefly the difference between supervised learning, unsupervised learning and reinforcement learning.

6b

Describe how you could use an autoencoder to denoise images with gaussian white noise.

(Continued on page 7.)

6c (PhD candidates only)

Describe the concepts of a variational autoencoder and what it is most commonly used for.

Exercise 7 Miscellaneous

7a

Why will the TensorFlow code in fig. 3 throw an error?

```
import tensorflow as tf

x = tf.Variable(initial_value=1)
y = tf.Variable(initial_value=2)
z = tf.add(x,y)

with tf.Session() as sess:
    z_val = sess.run(z)
```

Figure 3: TensorFlow code snippet

7b

Explain briefly the concept of Policy gradient methods in reinforcement learning.

7c (PhD candidates only)

Explain briefly the normalization done by the batch-norm algorithm during training and testing.