Examination questions for FYTN14 / EXTQ40 / NTF005F 2020-01-15 (14.00-19.00)

Approximately 17.5 points are required for passing the exam.

1. No derivations or explanations are needed for the following 5 questions!

a) How many **bias-nodes** do you have in an MLP with the N hidden layers? (1p)

b) Compute the derivative of the $1/(1+\exp(-x))$ activation function and express the derivative in terms of the function value. (1p)

c) True of false? Bagging is a method to generate datasets used to train networks in an ensemble? (1p)

d) What is the natural output activation function for binary classification problems? (1p)

e) Give an example of how to normalize continuous input values before training a neural network. (1p)

2. Figure 1 shows an MLP with one hidden node and a single output. The hidden activation function is φ_h and the output activation function is φ_o. This network have direct input to output weights.

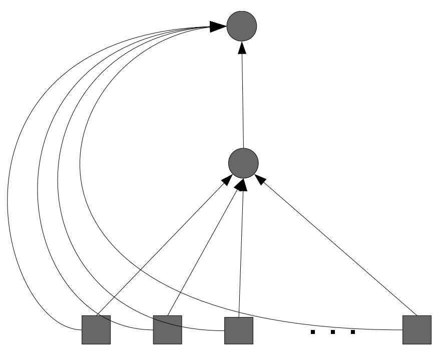

Figure 1:

a) Introduce symbols for the different weights and write down the function this network implements, i.e. the output as a function of inputs and the weights. Note! You should explicitly add the bias weights even though are not visible in the figure. (2p)

b) Write down the mean squared error function and the cross-entropy error function, and describe what kind of problems they are suitable for. (1p)

c) Many weight update approaches are based on the gradient of the error function. Using the symbols you introduced in (a), write down the derivative of the error function with respect to an input to hidden weight. (1p)

d) Some gradient descent methods (such as: momentum terms; "Rprop"; "Adam") include information about earlier gradients to hopefully improve weight updates. Describe one such algorithm in more detail than this brief introduction. (1p)

3.

a) What is over-training and why can it be a problem in machine learning? (2p)

b) "Early stopping", "L2 (weight decay) regularization" and "drop-out" can be used to reduce over-training. Describe each of them and explain why they can prevent over-training. (3p)

4. Model selection is an important part of machine learning! We can say that model selection finds suitable values for hyper-parameters such as regularization parameters or network design parameters. When doing model selection, the goal is to find a network (with all hyper-parameters) that has optimal *generalization* performance.

a) We can define hyper-parameters to be all "settings" (except the weights) that define the network and how to train it. Assume you should construct a MLP for a binary classification task. List different kinds of hyper-parameters when defining and training your MLP. (0.5 points per hyper-parameter, max 2p)

b) When doing model selection we typically need three different data sets. Let's call them D1, D2 and D3. Explain how to use the different datasets to find your optimal network and estimate the generalization performance. (3p)

5.

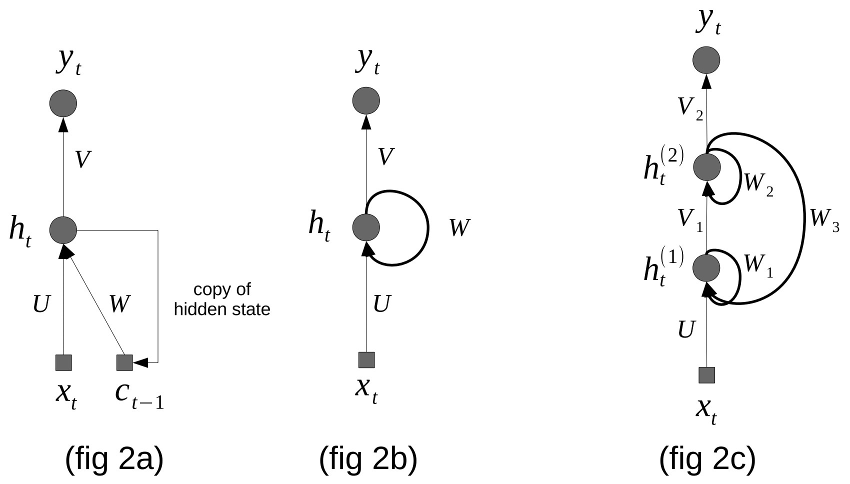

Figure 2:

a) Figure 2 (2a and 2b) shows two simple recurrent networks. Fig. 2a is the Elman type (ET) and fig. 2b is the general type (GT). Here an input sequence $x_1, x_2, ..., x_T$ is producing an output sequence $y_1, y_2, ..., y_T$ using a single hidden node with a feedback weight (W). The hidden activation function is φ_h and the output activation function is φ_o. The ET network works as follows:

$$\begin{aligned} y(t) &= g_{\text{o}}(Vh(t)) \\ h(t) &= g_{\text{h}}(Ux(t) + Wc(t)) \\ c(t) &= h(t-1) \end{aligned}$$

And the GT network have these update equations:

$$\begin{aligned} y(t) &= g_{\text{o}}(Vh(t)) \\ h(t) &= g_{\text{h}}(Ux(t) + Wh(t-1)) \end{aligned}$$

Although they appear to be same there is a difference between the two networks when evaluating the derivative,

$$\frac{\partial h(t)}{\partial W}$$

Explain the difference between ET and GT when evaluating this derivative (2p)

b) Figure 2 (2c) shows another recurrent network with two hidden nodes. Again the input sequence $x_1, x_2, ..., x_T$ is producing an output sequence $y_1, y_2, ..., y_T$. Unfold this network two time steps, i.e. show the unfolded network for inputs x_1, x_2, x_3 (mark all weights in the unfolded network with the labels $U, V_1, V_2, W_1, W_2, W_3$). (2p)

c) The GT network type can be used to model sequence data. When dealing with sequences with long time dependencies it can however be tricky to train the unfolded GT network because of the vanishing gradient problem. What is that? (1p)

6. Autoencoder: Figure 3 shows an example of an autoencoder.

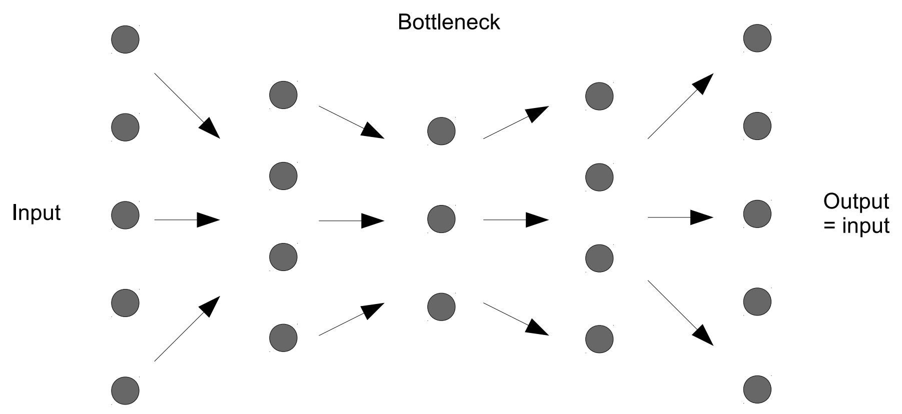

Figure 3:

a) Describe one usage of the autoencoder? (1p)

b) Assume that all input data for the autoencoder are binary (0/1) values. Write down a suitable error function for this autoencoder and explain why this error function. (2p).

CNN:

a) You have the following setup for a binary image classification task using CNNs. Image size: 11x11. One convolutional layer: 1 kernel, size 5x5 (stride = 2, no zero-padding) + max-pooling 2x2 (stride = 1). Dense layer: 3 hidden nodes, Output layer: 1 single output node. How many weights (including bias weights) does this CNN have? (2p)

7. Remember the Hopfield model (see figure 4)!

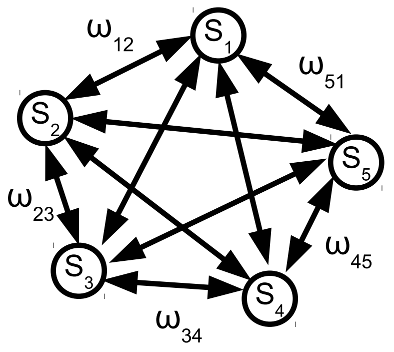

Figure 4:

It had the following properties,

- The nodes s_i in the network are binary with $s_i \in \{-1, 1\}$.
- We have symmetrical weights, i.e. $w_{ij} = w_{ji} \;\; \forall\, i, j$.
- No self feedback weights, i.e. $w_{ii} = 0 \;\; \forall\, i$.

Assume the nodes are updated by the following equations,

$$\begin{aligned} s_i &= \operatorname{sgn}(h_i) && (1) \\ \operatorname{sgn}(x) &= \begin{cases} 1 & \text{if} \quad x \geq 0 \\ -1 & \text{if} \quad x < 0 \end{cases} && (2) \end{aligned}$$

where $h_i = \sum_j w_{ij} s_j$. For the Hopfield model we can define an *energy* function,

$$E = -\frac{1}{2} \sum_i \sum_j w_{ij} s_i s_j$$

The idea of calling E an energy function is that it should decrease when updating the Hopfield model.

This is now your task, show that $\Delta E \leq 0$ when the nodes of the Hopfield model are updated according to eqn. 1 (5p).

Hint: You can assume serial updating of the nodes, meaning that you could analyze what happens to E when a given node e.g. s_k is updated to the value s'_k given by eqn. 1.

Good Luck!

Mattias & Patrik