Prof. T. Hofmann

Final Exam

February 7, 2017

First and Last name: ____________________

Student ID (Legi) Nr: ____________________

Signature: ____________________

General Remarks

- Please check that you have all 24 pages of this exam.
- There are 120 points, and the exam is 120 minutes. **Don't spend too much time on a single question!** The maximum of points is not required for the best grade!
- Remove all material from your desk which is not permitted by the examination regulations.
- Write your answers directly on the exam sheets. If you need more space, make sure you put your **student-ID**-number on top of each supplementary sheet.
- Immediately inform an assistant in case you are not able to take the exam under regular conditions. Later complaints are not accepted.
- Attempts to cheat/defraud lead to immediate exclusion from the exam and can have judicial consequences.
- Please use a black or blue pen to answer the questions.
- Provide only one solution to each exercise. Cancel invalid solutions clearly.

	Topic	Max. Points	Points Achieved	Visum
1	Feedforward networks, RNNs and CNNs	30		
2	Learning & Optimization	30		
3	Factor models, autoencoders, latent representation	30		
4	Undirected Deep Models & Generative models	30		
Total		120		

Grade: ....

1 Feedforward networks, RNNs and CNNs (30 pts)

1.1 Activation Functions

1. Show that the *hyperbolic tangent* function $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ is related to the *sigmoid function* $\sigma(x) = \frac{1}{1+e^{-x}}$ by $\tanh(x) = 2\sigma(2x) - 1$, where $x \in \mathbb{R}$.

   3 pts ☐

   ....

   ....

   ....

   ....

   ....

2. Show that the inverse of the *sigmoid function* is given by: $\sigma^{-1}(y) = \ln\left(\frac{y}{1-y}\right)$, where $y = \sigma(x)$.

   3 pts ☐

   ....

   ....

   ....

   ....

   ....

1.2 Training/Back-propagation

Given a training dataset $D = \{(\mathbf{x}_n, t_n)\}_{i=1}^{N}$, where $\mathbf{x}_n \in \mathbb{R}^D$ and $t_n \in \{0, 1\}$. We consider a binary classification problem with a neural network consisting of one single hidden layer and two hidden neurons parametrized by $\theta = \{\mathbf{w}_1, \mathbf{w}_2, b_1, b_2, v_0, v_1, v_2\}$. The class conditional is given by:

$$p(t_n|\mathbf{x}_n) = \sigma\Bigg(v_o + v_1 g(\mathbf{w}_1^T \mathbf{x}_n + b_1) + v_2 g(\mathbf{w}_2^T \mathbf{x}_n + b_2)\Bigg),$$

where $g(a_n) = e^{-\frac{1}{2}a_n^2}$ and $\sigma(a_n) = \frac{1}{1+e^{-a_n}}$, variable $a_n \in \mathbb{R}$.

1. Based on the assumption that the training data is i.i.d. , write down the log likelihood for the class conditioned on the inputs (cross-entropy error function):

   2 pts ☐

   ....

   ....

   ....

2. Calculate the partial derivatives of the cross-entropy function with respect to the model parameters w_1, b_1 and v_0 by using chain-rule. You can write the derivatives for just one datapoint $(\mathbf{x}_n, t_n)$.

   5 pts ☐

   ....

   ....

   ....

   ....

   ....

   ....

   ....

   ....

   ....

   ....

1.3 Loss Functions

Given a linear model $f(\mathbf{x}, \mathbf{w})$, where $\mathbf{x}, \mathbf{w} \in \mathbb{R}^D$, defined as:

$$f(\mathbf{x}, \mathbf{w}) = w_0 + \sum_{i=1}^{D} w_i x_i$$

and the sum-of-squares error loss given by:

$$L(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \left\{ f(x_n, \mathbf{w}) - t_n \right\}^2$$

where t_n is the target class for variable x_n. Suppose that Gaussian noise $\epsilon_i \sim \mathcal{N}(\mu = 0, \sigma^2)$ (with zero mean and variance σ^2) is added to each input variable x_i independently, i.e.

$$\hat{f}(\mathbf{x}, \mathbf{w}) = w_0 + \sum_{i=1}^{D} w_i (x_i + \epsilon_i)$$

By using $\mathbb{E}[\epsilon_i] = 0$ and $\mathbb{E}[\epsilon_i \epsilon_j] = \delta_{ij}\sigma^2$, show that minimizing the error loss over the noise distributions:

$$\hat{L}(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \left\{ \hat{f}(x_n, \mathbf{w}) - t_n \right\}^2$$

is equivalent to minimizing the sum-of-squares loss for noise-free variables with the addition of a weight-decay regularization term, in which the bias parameter (w_0) is omitted from the regularizer, i.e. prove that $\mathbb{E}[\hat{L}(\mathbf{w})] = L(\mathbf{w}) + \frac{1}{2} \sum_{i=1}^{D} \mathbf{w}_i^2 \sigma^2$

4 pts ☐

....

....

....

....

....

....

....

....

....

....

....

1.4 Backpropagation through time

1. Consider a bidirectional recurrent neural network with inputs $x = x_{1:T}$, forward hidden states $h_f = h_{f,1:T}$, backward hidden states $h_b = h_{b,1:T}$ and outputs $y = y_{1:T}$.

$$
\begin{aligned}
\bar{h}_{f,t} &= F_f(x_t, h_{f,t-1}; \theta) \\
h_{f,t} &= \sigma(\bar{h}_{f,t}) \\
\bar{h}_{b,t} &= F_b(x_t, h_{b,t+1}; \theta) \\
h_{b,t} &= \sigma(\bar{h}_{b,t}) \\
y_t &= G(h_{f,t}, h_{b,t}; \kappa) \\
L_{\text{total}} &:= L(y)
\end{aligned}
$$

where F_f, F_b and G are smooth vector valued functions, L is a smooth function into the reals, σ is an elementwise nonlinearity, θ and κ are parameter vectors and the initial states $h_{f,0}$ and $h_{b,T+1}$ are given. Note that L depends on all outputs y_1 through y_T.

Calculate $\frac{\partial L_{\text{total}}}{\partial \theta}$, the derivative of the total loss with respect to the parameters θ.
The final answer should only contain partial derivatives of functions with respect to their *direct* parameters, meaning it should not be possible to expand them further using the chain rule.

4 pts

....

....

....

....

....

....

1.5 True / False

Which of the following claims are true/false? *(1 point per correct answer, -1 point per incorrect answer, non-negative total points in any case)*

4 pts

1. A recursive neural network will generate - for the same input of length n - only $\mathcal{O}\log(n)$ hidden states, compared to the $\mathcal{O}(n)$ hidden states of a recurrent neural network.
   [ ] True [ ] False

2. The GRU and LSTM architectures seen in class violate the Markov assumption of basic RNNs.
   [ ] True [ ] False

Consider the Seq2Seq framework, which maps arbitrary length input sequences to arbitrary length output sequences.

3. The *encoder* of a seq2seq network does not have to be an RNN. It could also consist of only convolutional, pooling and fully connected layers.
   [ ] True [ ] False

4. The *decoder* of a seq2seq network does not have to be an RNN. It could also consist of only convolutional, pooling and fully connected layers.
   [ ] True [ ] False

1.6 CNNs

Not all images are two-dimensional. Medical devices such as MRI and CT scans produce 3D image data, which is made of *voxels* instead of pixels. An $N \times M \times K$ image I with C channels is thus an element of $\mathbb{R}^N \times \mathbb{R}^M \times \mathbb{R}^K \times \mathbb{R}^C$. It contains C values (e.g. density, heat, resonance, etc.) for each voxel location (n, m, k).
Analogous to 2D-image convolutions, we would like to create a convolution that employs translation invariant feature detection by convolving filters over the spatial domain at each location for a desired number L of output channels. Further, our filters should have equal side length F in each spatial dimension.

1. What will be the size of one of our filters (call it ∇_i)?

   1 pts ☐

   ....

2. How many of these filters do we need for such a convolution operation?

   1 pts ☐

   ....

3. Using a stride of S and zero-padding of P, which is the *width* and *height* of the feature map after applying the filter with equal side length of F?

   1 pts ☐

   ....

   ....

4. Write down the formula for the convolution. How do we obtain the value of a location (n, m, k, l) of the output O?

   2 pts ☐

   ....

   ....

2 Learning & Optimization (30 pts)

2.1 Optimization

A) Consider the function $f(\mathbf{x})$ where $\mathbf{x} = [x_1\ x_2] \in \mathbb{R}^2$ defined as

$$f(x_1, x_2) = (x_1^2 - x_2^2).$$

1. Find a critical point of $f(x)$. Is this a local maximum, a local minimum or a saddle point? Justify your answer.

2 pts ☐

. . . .

. . . .

. . . .

2. Initialize Newton's method at time step 0 at $\mathbf{x}^0 = [x_1^0\ x_2^0] = [10\ 10]$. Write down $\mathbf{x}^1 = [x_1^1\ x_2^1]$ after one step of Newton's method. Is $\mathbf{x}_1$ a local minimum?

4 pts ☐

. . . .

. . . .

. . . .

B) Consider the function $f(\mathbf{x}) = \mathbf{x}^\top \mathbf{A}\mathbf{x} + \mathbf{b}^\top \mathbf{x} + c$, where $\mathbf{x}, \mathbf{b} \in \mathbb{R}^d, c \in \mathbb{R}$ and $\mathbf{A} \in \mathbb{R}^{d \times d}$.

1. Write down the update of Newton's method after one step. What do you observe?

4 pts ☐

. . . .

. . . .

. . . .

. . . .

2. Show that if $\mathbf{A}$ is not positive definite (i.e. some eigenvalues are negative) then $f(\mathbf{x})$ is unbounded from below (i.e. goes to $-\infty$).
   Hint: Recall that $\mathbf{A}$ is positive definite if there exists $\mathbf{x} \neq \mathbf{0}$ such that $\mathbf{x}^\top \mathbf{A}\mathbf{x} \geq 0$.

3 pts ☐

. . . .

. . . .

. . . .

2.2 Activation function

Consider the function $f(\mathbf{x};\mathbf{W}) = \text{ReLU}(\mathbf{W}\mathbf{x})$, where $\mathbf{x} \in \mathbb{R}^d$ corresponds to some input features and $\mathbf{W} \in \mathbb{R}^{n \times d}$ is a matrix of weight parameters.

1. Write-down the gradient and the Hessian of the function $f(\mathbf{x};\mathbf{W})$ with respect to $\mathbf{W}$.

3 pts ☐

. . . .

. . . .

. . . .

2. Write-down the Taylor expansion of $f(\mathbf{x};\mathbf{W})$ around some $\mathbf{W}x > 0$ ($>$ element-wise).

2 pts ☐

. . . .

. . . .

. . . .

2.3 Critical points

1. Let $\mathbf{x} \in \mathbb{R}^n$ and $f(\mathbf{x}) \in \mathbb{R}$ be of class C^2 (i.e. first two derivatives exist and are continuous). Show that if $\mathbf{x}^* \in \mathbb{R}^n$ is a local minimum for f then $\nabla f(\mathbf{x}^*) = 0$ **and** $\nabla^2 f(\mathbf{x}^*)$ positive semi-definite.
   No points if you only show $\nabla f(\mathbf{x}^*) = 0$.

5 pts [ ]

. . . .

. . . .

. . . .

. . . .

. . . .

2. If $\nabla f(\mathbf{x}^*) = 0$ and $\nabla^2 f(\mathbf{x}^*)$ positive definite, then $\mathbf{x}^*$ is a *strict* local minimum.
   Recall: $\mathbf{x}^*$ is a strict local minimum point if there exists some $\epsilon > 0$ such that, for all $\mathbf{x} \in \mathbb{R}^n$ within distance ϵ of $\mathbf{x}^*$ with $\mathbf{x} \neq \mathbf{x}^*$, we have $f(\mathbf{x}^*) < f(\mathbf{x})$.

4 pts [ ]

. . . .

. . . .

. . . .

. . . .

. . . .

2.4 Regularization

Which of the following claims are true/false? *(1 point per correct answer, -1 point per incorrect answer, non-negative total points in any case)*

3 pts [ ]

a) The regularization imposed via constrained optimization always forces the weight vector to decay during the course of the optimization.
   [ ] True [ ] False

b) When regularizing via early stopping, parameter values corresponding to directions of significant curvature (of the objective function) are regularized more than directions of less curvature.

[ ] True [ ] False

c) Dropout aims to approximate bagging but with an exponentially large number of neural networks.

[ ] True [ ] False

3 Factor models, autoencoders, latent representation

(30 pts)

3.1 Sufficient statistics

Let's denote the examples by $\mathbf{x} \in \mathbb{R}^d$, where d is the number of features. The labels are denoted by $y \in \{1, \ldots, K\}$, where K is the number of classes. Logistic regression considers the following *discriminative model* for classification:

$$p(y|\mathbf{x}, \mathbf{w}) = \frac{1}{Z(\mathbf{x}, \mathbf{w})} \exp(\langle \mathbf{w}_y, \mathbf{x} \rangle), \tag{1}$$

where $\mathbf{w}_y \in \mathbb{R}^d$, for each of the labels $y \in \{1, \ldots, K\}$, are the parameters of the model, i.e. in total we have dK parameters. For simplicity we use $\mathbf{w}$ to denote all the parameters of the model, we could do this for example by concatenation of the individual $\mathbf{w}_y$'s. $Z(\mathbf{w}, \mathbf{x}) = \sum_y \exp(\langle \mathbf{w}_y, \mathbf{x} \rangle)$ denotes the *partition sum*; we will find it convenient to denote the log partition sum by $A(\mathbf{w}, \mathbf{x}) := \log Z(\mathbf{w}, \mathbf{x})$, it is sometimes also called *cumulant generating function* or *moment generating function*.

The distribution in equation (1) can be cast as an *exponential family distribution*, which for example also contains the Gaussian distribution. In general an exponential family distribution has the following form:

$$p(\mathbf{x}|\theta) = \exp(\langle \theta, s(\mathbf{x}) \rangle - A(\theta)), \quad A(\theta) = \log \int_{\mathbf{x}} \exp(\langle \theta, s(\mathbf{x}) \rangle) \mathbf{dx}$$

where θ are called the natural parameters and $s(\mathbf{x})$ the sufficient statistics.

One can derive a number of important properties about exponential family distributions:

1. Show that the derivatives of the cumulant generating function $A(\theta)$ are given as the moments of the sufficient statistics, i.e.

$$\frac{\partial A(\theta)}{\partial \theta} = \mathbb{E}_{p(\mathbf{x}|\theta)}[s(\mathbf{x})]$$
$$\frac{\partial^2 A(\theta)}{\partial \theta^2} = \mathsf{Cov}_{p(\mathbf{x}|\theta)}[s(\mathbf{x})].$$

6 pts ☐

. . . .

. . . .

. . . .

. . . .

. . . .

2. Show that the cumulant generating function $A(\theta)$ is convex w.r.t. the parameters θ.

3 pts [ ]

. . . .

. . . .

. . . .

3.2 Autoencoders

Let a two layer linear auto-encoder with m units and no biases be parametrized by weights $\mathbf{C} \in \mathbb{R}^{m \times d}$ (encoder) and $\mathbf{D} \in \mathbb{R}^{d \times m}$ (decoder).

1. Given datapoints $\mathbf{x}_1, \ldots, \mathbf{x}_n$ write down the objective of the reconstruction problem.

2 pts [ ]

. . . .

. . . .

2. After training, you are given the decoder $\mathbf{D}$ but only a corrupted version of the encoder $\tilde{\mathbf{C}}$. *(1/2 point per correct answer, -1/2 point per incorrect answer, non-negative total points in any case)*

2 pts [ ]

Assume $\tilde{\mathbf{C}}$ is a sheared version of $\mathbf{C}$ and you know the exact shearing matrix. (Hint: A shear matrix is invertible but non-orthogonal.) True or false:

We can adjust $\mathbf{D}$ so that:

(a) The mapping $\mathbf{x} \to \mathbf{z}$ stays the same [ ] True [ ] False

(b) The mapping $\mathbf{x} \to \hat{\mathbf{x}}$ stays the same [ ] True [ ] False

Assume $\tilde{\mathbf{C}}$ is a rotated version of $\mathbf{C}$ and you know the exact shearing matrix. True or false:

We can adjust $\mathbf{D}$ so that:

(a) The mapping $\mathbf{x} \to \mathbf{z}$ stays the same [ ] True [ ] False

(b) The mapping $\mathbf{x} \to \hat{\mathbf{x}}$ stays the same [ ] True [ ] False

3. Now let's assume that the autoencoder has tied weights $\mathbf{D} = \mathbf{C}^\top$ and we actually know that $\mathbf{C}$ has been subject to a rotation described by a rotation matrix $\mathbf{R}$. Give an adapted version $\tilde{\mathbf{D}}$ (as a function of $\mathbf{D}$) so that the mapping $\mathbf{x} \to \hat{\mathbf{x}}$ stays the same.

2 pts ☐

. . . .

. . . .

3.3 Regularized Autoencoders

Let us consider a regularized variant of an autoencoders called "contractive autoencoder". Given a dataset $\mathcal{D} = \{\mathbf{x}^{(k)}\}_{k=1}^n, \mathbf{x}^{(k)} \in \mathbb{R}^d$ and an encoder function f as well as decoder g, we want to minimize the following objective,

$$\mathcal{L} = \sum_{\mathbf{x} \in \mathcal{D}} \|\mathbf{x} - g(f(\mathbf{x}))\|^2 + \lambda \|J_f(\mathbf{x})\|_F^2,$$

where $\|J_f(\mathbf{x})\|_F^2 = \sum_{ij} \left(\frac{\partial (f(\mathbf{x}))_j}{\partial \mathbf{x}_i} \right)^2$ is the Frobenius norm of the Jacobian of the encoder f.

1. Describe the regularizer's effect on f when the regularizer dominates the loss, e.g. $\lambda \to \infty$ (no calculation required)

2 pts ☐

. . . .

. . . .

2. Let us assume that the autoencoder is linear and it uses tied weights (f and g share weights $\mathbf{W}$). Show that in this setting, the regularizer is equivalent to an l2-regularizer on the weights $\mathbf{W}$.

4 pts ☐

. . . .

. . . .

. . . .

. . . .

3.4 Deep Latent Gaussian Models (DLGMs)

1. Which of the following claims are true/false? *(1 point per correct answer, -1 point per incorrect answer, non-negative total points in any case)*

   4 pts [ ]

   The ELBO is a lower bound.

   [ ] True [ ] False

   The ELBO is an approximation to the true, intractable posterior.

   [ ] True [ ] False

   Optimizing the ELBO implies minimizing the entropy of the variational distribution.

   [ ] True [ ] False

   Optimizing the ELBO implies maximizing the KL divergence between the variational distribution and the prior over the latent variables.

   [ ] True [ ] False

2. Let's look at a simple DLGM as presented in the lecture consisting of the following ingredients (from input to output)

   - A neural recognition network $(\mu(\mathbf{x}), \mathbf{\Sigma}(\mathbf{x})) = f_\theta(\mathbf{x})$ mapping inputs to a Gaussian parametrization. The network itself is parametrized by θ.
   - A sampling step $\mathbf{z} \sim \mathcal{N}(\mu(\mathbf{x}), \mathbf{\Sigma}(\mathbf{x}))$
   - A neural reconstruction network $\hat{\mathbf{x}} = g_\xi(\mathbf{z})$ mapping back to the input space. The network itself is parametrized by ξ.

   The stochastic nature of DLGMs makes learning the network more challenging. Explain which parameters, θ or ξ, are affected and why.

   3 pts [ ]

   ....

   ....

   ....

   ....

3. In the lecture you have seen continuous latent variables coming from a Gaussian distribution. For discrete latent variables, we could use a multinomial distribution. However, when using stochastic back-propagation for such a model there is a conceptional problem. Which? (just point to the issue, no detailed explanation necessary)

2 pts ☐

. . . .

. . . .

4 Undirected Deep Models & Generative models (30 pts)

4.1 Generative Adversarial Networks

1. For scalar $\mathbf{x}$ and $\mathbf{y}$, consider the value function $V(\mathbf{x}, \mathbf{y}) = \mathbf{xy}$. Does this game have an equilibrium? If so, where is it?

2 pts ☐

. . . .

. . . .

2. The training criterion for the discriminator D, given any generator G, is to maximize the quantity $V(G, D)$

$$\begin{aligned} V(G, D) &= \int_{\boldsymbol{x}} p_{\text{data}}(\boldsymbol{x}) \log(D(\boldsymbol{x}))dx + \int_{z} p_{\boldsymbol{z}}(\boldsymbol{z}) \log(1 - D(g(\boldsymbol{z})))dz \\ &= \int_{\boldsymbol{x}} p_{\text{data}}(\boldsymbol{x}) \log(D(\boldsymbol{x})) + p_g(\boldsymbol{x}) \log(1 - D(\boldsymbol{x}))dx \end{aligned} \tag{2}$$

Show that for G fixed, the optimal discriminator $D^*_G(\boldsymbol{x})$ is

$$D^*_G(\boldsymbol{x}) = \frac{p_{\text{data}}(\boldsymbol{x})}{p_{\text{data}}(\boldsymbol{x}) + p_g(\boldsymbol{x})} \tag{3}$$

3 pts ☐

. . . .

. . . .

. . . .

3. Express this optimal discriminator $D^*_G(\boldsymbol{x})$ using a sigmoid σ, a logarithm and p_{data}, p_g.

1 pts ☐

. . . .

. . . .

4. Using that $\sigma'(\cdot) = \sigma(\cdot)(1-\sigma(\cdot))$, show that if $D_G^*(\boldsymbol{x}) \neq 1$, then

$$\left\| \frac{\nabla_{\boldsymbol{x}} \log(D_G^*(\boldsymbol{x}))}{1 - D_G^*(\boldsymbol{x})} \right\|^2 = \|\nabla_{\boldsymbol{x}} \log(p_{\text{data}}(\boldsymbol{x})) - \nabla_{\boldsymbol{x}} \log(p_g(\boldsymbol{x}))\|^2. \tag{4}$$

3 pts ☐

. . . .

. . . .

. . . .

. . . .

Remark. Note that as the generator depends on some parameters θ, so does the quantity $J(\theta) = \int_{\boldsymbol{x}} \|\nabla_{\boldsymbol{x}} \log(p_{\text{data}}(\boldsymbol{x})) - \nabla_{\boldsymbol{x}} \log(p_g(\boldsymbol{x}))\|^2 p_{\text{data}}(\boldsymbol{x}) d\boldsymbol{x}$. Now we assume that, although we don't have access to p_{data} in practice, we can compute $\nabla_\theta J(\theta)$ using some trick. (Optimizing this quantity is called *score-matching*.)

5. Consider the two quantities

$$q_1(\theta) = \mathbb{E}_{x \sim p_{\text{data}}}(D_G^*(\boldsymbol{x})) \text{ and } q_2(\theta) = \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}}\left(\|\frac{\nabla_{\boldsymbol{x}} \log(D_G^*(\boldsymbol{x}))}{1 - D_G^*(\boldsymbol{x})}\|^2\right). \tag{5}$$

For each of them, explain if and why, in practice, we can or cannot compute their gradients with respect to θ.

2 pts ☐

. . . .

. . . .

. . . .

4.2 Restricted Boltzmann Machines

Suppose that we are given a restricted Bolztmann machine (RBM) with energy function

$$E(\boldsymbol{x}, \boldsymbol{z}) = -\boldsymbol{z}^T \boldsymbol{W} \boldsymbol{x} - \boldsymbol{c}^T \boldsymbol{x} - \boldsymbol{b}^T \boldsymbol{z},$$

where $\boldsymbol{x}$ is the input and $\boldsymbol{z}$ the hidden variable. Assume that both $\mathbf{x}$ and $\mathbf{z}$ are binary vectors, i.e. $\mathbf{x}, \mathbf{z} \in \{0, 1\}$. The joint probability density is given by $p(\boldsymbol{x}, \boldsymbol{z}) = \frac{e^{-E(\boldsymbol{x}, \boldsymbol{z})}}{Z}$.

1. Express $p(\boldsymbol{x})$ in the form $Z^{-1} e^{-F(\boldsymbol{x})}$ (i.e. marginalize over $\mathbf{z}$), and express F using the non-linearity defined by $s(x) = \log(1 + \exp(x))$. We call F the *free-energy*.

   4 pts ☐

   ....

   ....

   ....

   ....

   ....

   ....

2. The non-linear function $s(x) = \log(1 + \exp(x))$ can be seen as a smooth version of another non-linear function commonly used in neural networks. Which function is this?

   1 pts ☐

   ....

   ....

3. Show that $\nabla_{\boldsymbol{W}} E(\boldsymbol{x}, \boldsymbol{z}) = -\boldsymbol{z}\boldsymbol{x}^T$.

   2 pts ☐

   ....

   ....

   ....

4. Show that $\mathbb{E}_{\boldsymbol{z}}(\nabla_{\boldsymbol{W}} E(\boldsymbol{x}, \boldsymbol{z})|\boldsymbol{x}) = -\boldsymbol{y}\boldsymbol{x}^T$ where $\boldsymbol{y}$ is a vector that you will define, depending on $\boldsymbol{x}$, and using a sigmoid.

4 pts ☐

. . . .

. . . .

. . . .

. . . .

. . . .

4.3 Deep Belief Networks

Suppose that we are given a deep belief network (DBN), consisting of a restricted Bolztmann machine (RBM) with energy function

$$E(\boldsymbol{z}^{(2)}, \boldsymbol{z}^{(3)}) = -\boldsymbol{z}^{(2)T}\boldsymbol{W}^{(3)}\boldsymbol{z}^{(3)} - \boldsymbol{b}^{(2)T}\boldsymbol{z}^{(2)} - \boldsymbol{b}^{(3)T}\boldsymbol{z}^{(3)},$$

with joint probability density given by $p(\boldsymbol{z}^{(2)}, \boldsymbol{z}^{(3)}) = Z^{-1}e^{-E(\boldsymbol{z}^{(2)}, \boldsymbol{z}^{(3)})}$, then stacked successively with two sigmoid belief networks (as in the lectures), $\boldsymbol{z}^{(1)}$ on top of $\boldsymbol{z}^{(2)}$, and then $\boldsymbol{x}$ on top of $\boldsymbol{z}^{(1)}$, meaning that

$$p(z_j^{(1)} = 1|\boldsymbol{z}^{(2)}) = \sigma(b_j^{(1)} + \boldsymbol{W}_{j\cdot}^{(2)}\boldsymbol{z}^{(2)})$$

and

$$p(x_i = 1|\boldsymbol{z}^{(1)}) = \sigma(b_i^{(0)} + \boldsymbol{W}_{i\cdot}^{(1)}\boldsymbol{z}^{(1)}).$$

Assume we only consider binary vectors for $\boldsymbol{x}$ and the $\boldsymbol{z}^{(i)}$'s, i.e. with coordinates in $\{0, 1\}$.

1. (Factorisation of a DBN) In the following expression, replace the $\square$'s by either of $\boldsymbol{x}$, $\boldsymbol{z}^{(1)}$, $\boldsymbol{z}^{(2)}$ or $\boldsymbol{z}^{(3)}$.

$$p(\boldsymbol{x}, \boldsymbol{z}^{(1)}, \boldsymbol{z}^{(2)}, \boldsymbol{z}^{(3)}) = p(\square, \square)p(\square|\square)p(\square|\square). \tag{6}$$

1 pts ☐

. . . .

2. Express $p(\boldsymbol{x}, \boldsymbol{z}^{(1)}, \boldsymbol{z}^{(2)}, \boldsymbol{z}^{(3)})$ as a function of the parameters of the DBN.

3 pts ☐

. . . .

. . . .

. . . .

3. (Variational bound) Now assume that we are trying to approximate $p(\boldsymbol{z}^{(1)}|\boldsymbol{x})$ by $q(\boldsymbol{z}^{(1)}|\boldsymbol{x})$. Show that

$$\log(p(x)) \geqslant \sum_{\boldsymbol{z}^{(1)}} q(\boldsymbol{z}^{(1)}|\boldsymbol{x}) \log\left(p(\boldsymbol{z}^{(1)}, \boldsymbol{x})\right) - \sum_{\boldsymbol{z}^{(1)}} q(\boldsymbol{z}^{(1)}|\boldsymbol{x}) \log\left(q(\boldsymbol{z}^{(1)}|\boldsymbol{x})\right). \qquad (7)$$

3 pts ☐

. . . .

. . . .

. . . .

. . . .

. . . .

4. Recall that the KL-divergence between two probability distributions P and Q is defined by $D_{KL}(Q||P) = \sum_i Q(i) \log(\frac{Q(i)}{P(i)})$. Reformulate the previous inequality using a KL-divergence.

1 pts ☐

. . . .

. . . .

Supplementary Sheet

Supplementary Sheet