ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Deep Learning
Fall 2017

Prof. T. Hofmann

Final Exam

February 5, 2018

First and Last name: _____

Student ID (Legi) Nr: _____

Signature: _____

General Remarks

- Please check that you have all 26 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	Compositional Models and Approximation Theory	24		
2	Feedforward Networks and Backpropagation	24		
3	Learning, Optimization and Regularization	24		
4	CNNs, RNNs, Memory and Attention	24		
5	Factor Models, Autoencoders and Generative Models	24		
Total		120		

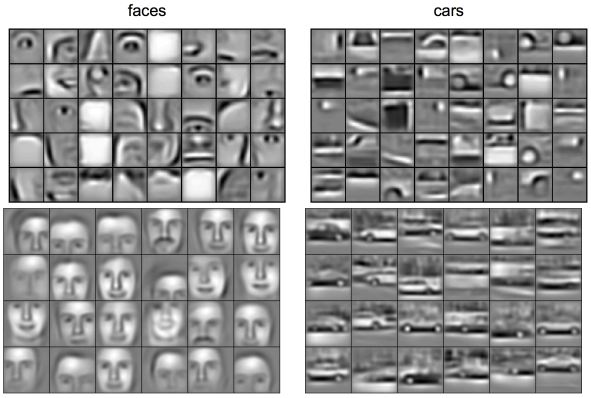
Grade: .....

1 Compositional Models and Approximation Theory

(24 pts)

1.1 Representation Learning

The figure below shows the activation patterns of intermediate representations learned in a deep belief network for two different datasets consisting of faces and cars.



(i) For both datasets, which set of activation patterns (top images or bottom images) corresponds to which level of network layer? Justify your answer and distinguish between low layers, i.e. closer to the input layer and high layers, i.e. closer to the output layer.

2 pts

. . . . . .

. . . . . .

. . . . . .

1.2 Linearity

Consider the following map from $\mathbf{R}^n$ to $\mathbf{R}$:

$$f(x) = w^\top x + b \quad x \in \mathbf{R}^n, 0 \neq w \in \mathbf{R}^n, 0 \neq b \in \mathbf{R} \tag{1}$$

and consider its linearity according to the definition provided in class (additivity + homogeneity).

(i) Is the map $f(x)$ linear? Give a formal derivation.

3 pts

. . . . . .

. . . . . .

. . . . . .

1.3 Level Sets

Consider the following function from $\mathbf{R}^n$ to $\mathbf{R}$:

$$f(x) = \sin(w^\top x + b) \quad x \in \mathbf{R}^n, 0 \neq w \in \mathbf{R}^n, 0 \neq b \in \mathbf{R}. \tag{2}$$

(i) Is $f(x)$ a ridge function? Explain why.

1 pts ☐

. . . . . .

. . . . . .

. . . . . .

(ii) Calculate the derivative $\nabla_x f$.

2 pts ☐

. . . . . .

. . . . . .

. . . . . .

(iii) Give a formal description of the level set $L_f(1)$.

2 pts ☐

. . . . . .

. . . . . .

. . . . . .

1.4 Rectified Linear Units

Consider a rectified linear unit from $\mathbf{R}^2$ to $\mathbf{R}$:

$$f(x) = \max(0, w_1 x_1 + w_2 x_2 + b), \quad w_1 > 0,\ w_2 < 0, \quad b \neq 0. \tag{3}$$

(i) Find the subdifferential of f at each point $x \in \mathbb{R}^2$, i.e.

$$\partial f(x) = \{v \in \mathbb{R}^2 \mid f(y) - f(x) \geq v^\top (y - x) \ \forall y \in \mathbb{R}^2\}.$$

Hint: use the property $\partial f(x) = \text{conv}\left(\cup_{i:\, f_i(x)=f(x)} \partial f_i(x)\right)$ for subdifferential of maximum $f(x) = \max\{f_i(x) \,|\, i = 1, \ldots, n\}$, i.e., the subdifferential of the maximum of functions is the convex hull of the union of subdifferentials of the active functions at x.

2 pts ☐

. . . . . .

. . . . . .

. . . . . .

. . . . . .

(ii) Propose a neural network architecture with the universal approximation property. Give a short proof sketch.

3 pts ☐

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

(iii) Prove that any continuous function in $[0, 1]$ can be uniformly approximated to arbitrary precision by a piecewise linear function.

2 pts ☐

. . . . . .

. . . . . .

. . . . . .

1.5 True / False

Are the following statements true or false? (+1 point for correct answers, -1 point for incorrect answers, no negative total points)

7 pts ☐

(i) Every continuous piecewise linear function from $\mathbf{R}^n \to \mathbf{R}$ can be written as a signed sum of k-hinges with $k \leq n+1$.

[ ] True [ ] False

(ii) Every k-hinge can be expressed as a nested combination of functions of the form $f(x) = \sigma(w^\top x + b)$ where σ denotes the absolute value function.

[ ] True [ ] False

(iii) Every k-hinge can be expressed as a nested combination of functions of the form $f(x) = \sigma(w^\top x + b)$ where σ denotes the $\max(x, 0)$ function.

[ ] True [ ] False

(iv) Every function whose epigraph is a polyhedral set is convex.

[ ] True [ ] False

(v) Every continuous piecewise linear function can be represented exactly as the output of a linear network with two maxout units and a linear output unit.

[ ] True [ ] False

(vi) Each continuous function on a convex set always reaches its infimum on it.

[ ] True [ ] False

(vii) Each continuous function on a bounded set always reaches its supremum on it.

[ ] True [ ] False

2 Feedforward Networks and Backpropagation (24 pts)

2.1 Activation Functions

(i) Consider the activation function

$$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0. \end{cases} \tag{4}$$

Why is (4) generally not used for neural network training?

1 pts ☐

. . . . . .

. . . . . .

(ii) Now, consider a two-layer feedforward network in which the non-linear activations are given by the sigmoid function $\sigma(z) = \frac{1}{1+\exp(-z)}$. Show that there exists an equivalent network, which computes exactly the same function, but with activations given by $\tanh(z) = \frac{\exp(z)-\exp(-z)}{\exp(z)+\exp(-z)}$. First derive the relation between $\sigma(z)$ and $\tanh(z)$ and then show that the parameters of the two networks differ by linear transformations.

4 pts ☐

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

(iii) Consider the following two-layer network

$$F(\mathbf{x}) = h_2(\mathbf{W}_2\, h_1(\mathbf{W}_1\, \mathbf{x} + \mathbf{b}_1) + \mathbf{b}_2) \tag{5}$$

where $h_{1,2}(\cdot)$ denotes some activation function and where $\mathbf{x} \in \mathbb{R}^d, \mathbf{W}_i \in \mathbb{R}^{k\times d}, \mathbf{b}_i \in \mathbb{R}^k$. Prove that the expressiveness of linear networks (i.e. $h(\cdot)$ being the identity function) does not increase with depth.

2 pts ☐

. . . . . .

2.2 Elementary Logic Functions

Feedforward neural networks with linear activations (i.e. identity activation functions) have many limitations. Most famously, they cannot learn the XOR($[x_1, x_2]$) function: XOR($[0, 1]$) = XOR($[1, 0]$) = 1 and XOR($[0, 0]$) = XOR($[1, 1]$) = 0.

(i) Give a short proof or illustrate in the x_1, x_2-plane why this is not possible.

2 pts ☐

(ii) Show that a non-linear two-layer neural network can in fact solve the XOR problem. Consider the network given by $f(\mathbf{x}; \mathbf{W_2}, b_2, \mathbf{W_1}, \mathbf{b}_1) = \mathbf{W}_2\, h(\mathbf{W}_1\mathbf{x} + \mathbf{b}_1) + b_2$ with $h(z) = \max(0, z)$ and

$$\mathbf{W}_1 = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}, \quad \mathbf{b}_1 = \begin{bmatrix} 0 \\ -1 \end{bmatrix}, \quad \mathbf{W}_2 = ?, \quad b_2 = ?$$

Find the corresponding weights $\mathbf{W}_2$ and bias b_2 and show that this network gives the correct output on all inputs $\mathbf{x} \in \{0, 1\}^2$.

2 pts ☐

2.3 Backpropagation and Computational Graphs

Consider the schematic representation in Figure 1 of a two-layer neural network, where x denotes the input, W_1 denotes the weight (we ignore biases in this exercise), h_1 denotes the hidden activation function and ℓ denotes the unregularized loss function.

(i) Write down the regularized loss function denoted by L for an L_2 weight-decay regularizer.

1 pts ☐

. . . . . .

. . . . . .

(ii) Using the figure below, first add the computational nodes corresponding to the L_2 weight-decay regularizer. Then extend the computational graph for backpropagation, i.e. add the nodes for each gradient that contributes to $\frac{\partial L}{\partial w_1}$, including those resulting from weight-decay regularization.

3 pts ☐

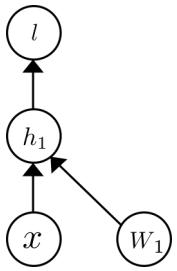


Figure 1: Two-layer neural network

2.4 Reparametrization trick

A recurring task for posterior computations in variational inference and policy learning is computing the gradient of an expectation of a smooth function f

$$\nabla_\theta \mathbb{E}_{q_\theta(z|x)} f(z) = \nabla_\theta \int q_\theta(z|x) f(z) dz, \tag{6}$$

where $z \sim q_\theta(z|x)$ is a parametric model of the true posterior $p_\theta(z|x)$, parametrized by a neural network. Since backpropagation cannot flow through a random node, a common technique is to reparametrize $z = g_\theta(x, \epsilon)$ with an auxiliary noise variable $\epsilon \sim p(\epsilon)$ in order to decouple the generative parameters θ from the randomness in z.

(i) How can you reparametrize a univariate gaussian variable $\bar{z} \sim \mathbb{N}(\mu, \sigma^2)$ in terms of $\epsilon \sim \mathbb{N}(0, 1)$?

1 pts

(ii) Assume that there exists a transformation $z = g_\theta(x, \epsilon)$ that leaves the differential mass invariant, i.e. $q_\theta(z|x)dz = p(\epsilon)d\epsilon$.
Show that one can compute the gradient of Eq. (6) as follows:

$$\nabla_\theta \mathbb{E}_{q_\theta(z|x)} f(z) = \mathbb{E}_{p(\epsilon)} \nabla_\theta f(g_\theta(x, \epsilon)). \tag{7}$$

3 pts

2.5 True / False

Are the following statements true or false? (+1 point for correct answers, -1 point for incorrect answers, no negative total points)

5 pts

(i) A linear perceptron can learn the function $f_1 : \{0, 1\}^3 \to \{0, 1\}$, $f_1 = x_2 \wedge (x_1 \vee (\neg x_1 \wedge x_3))$
[ ] True [ ] False

(ii) A linear perceptron can learn the function $f_2 : \{0, 1\}^3 \to \{0, 1\}$,
$f_2 = (x_1 \wedge \neg x_2) \vee (\neg x_1 \wedge x_2 \wedge x_3))$
[ ] True [ ] False

(iii) The error surface of a deep linear network is globally convex.
[ ] True [ ] False

(iv) Several parameter configurations (i.e. weights and biases) may represent the same neural network function.
[ ] True [ ] False

(v) The number of partial derivatives computed in one backpropagation pass is $V \cdot E$ for a network of V nodes and E edges.
[ ] True [ ] False

3 Learning, Optimization and Regularization (24 pts)

3.1 Stochastic Gradient Descent

The most common way of training Neural Networks involves some variant of Stochastic Gradient Descent (SGD). In its most basic form, a datapoint i is sampled uniformly at random (thus $\mathbb{E}(\nabla f_i(x) = \nabla f(x))$ in each iteration $k = 1, 2, ...$ and the iterates are updates as

$$x_{k+1} = x_k - \alpha_k \nabla f_i(x_k). \tag{8}$$

Assume the function f is smooth, i.e.

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\| \quad \forall\, x, y. \tag{9}$$

Show that, given x_k and a constant step size $\alpha_k = 1/(2L)$, SGD does not converge to a critical point x^*, i.e. that

$$\mathbb{E}\left(\|x^{k+1} - x^*\|_2^2\right) \geq \frac{1}{(2L)^2}\mathbb{E}\|\nabla f_i(x_k)\|_2^2 \tag{10}$$

4 pts

3.2 Risk Function Analysis

(i) We have seen in class that if $\mathcal{R}(\theta)$ is a risk function parametrized by $\theta \in \mathbb{R}^d$, and θ^* is its global minimum, then the Taylor approximation of $\mathcal{R}$ around the optimal θ^* is

$$\mathcal{R}(\theta) \approx \mathcal{R}(\theta^*) + \frac{1}{2}(\theta - \theta^*)^\top \mathbf{H}\,(\theta - \theta^*), \tag{11}$$

where the Hessian $\mathbf{H} := \nabla^2 \mathcal{R}\big|_{\theta^*}$ can be diagonalized as $\mathbf{H} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^\top$, $\mathbf{\Lambda} = \text{diag}(\lambda_1, \ldots, \lambda_d)$.

To simplify computations, you can assume that the Taylor approximation shown above is exact in an ϵ-ball around θ^*, i.e. that the high-order ($O(\|\theta - \theta^*\|^3)$ terms can be ignored for $\|\theta - \theta^*\| < \epsilon$.

Prove that $\mathbf{H}$ is positive semidefinite.

3 pts

(ii) We have seen in class that applying gradient descent to the risk approximation given in Eq. 11 yields the following update for each iteration t:

$$\tilde{\theta}(t+1) - \tilde{\theta}^* = (\mathbf{I} - \eta\mathbf{\Lambda})(\tilde{\theta}(t) - \tilde{\theta}^*), \quad \tilde{\theta} := \mathbf{Q}^\top\theta$$

where $\eta > 0$ is the learning rate. Assuming $\theta(0) = \mathbf{0}$, we explicitly derived in the lecture that

$$\tilde{\theta}(t) = \left[\mathbf{I} - (\mathbf{I} - \eta\mathbf{\Lambda})^t\right]\tilde{\theta}^* \tag{12}$$

Prove that if $\eta < \frac{2}{\max_i \lambda_i}$, then gradient descent converges to the optimal value, i.e. prove that

$$\lim_{t\to\infty} \theta(t) = \theta^*$$

3 pts

3.3 L1 Regularization

In this exercise we would like to understand the well known fact that L1 regularization induces sparsity of the model's parameter vector.

(i) Recall the L1 regularizer of a loss function parametrized by $\theta \in \mathbb{R}^d$ is defined as

$$\Omega(\theta) := \|\theta\|_1 = \sum_{i=1}^{d} |\theta_i|$$

Compute the gradient $\nabla_\theta\Omega(\theta)$ for all θ with non-zero components.

1 pts

(ii) We use the Taylor approximation of the risk function $\mathcal{R}$ around its optimal θ^* value shown in Eq. 11. We assume that the Hessian is diagonal, i.e. $\mathbf{H} = \mathbf{\Lambda} = \text{diag}(\lambda_1, \ldots, \lambda_d)$ where $\lambda_i > 0, \forall i$. This assumption holds if the data was preprocessed to remove all correlations between the input features, which may be accomplished using PCA.

Justify why the L1 regularized risk function $\hat{\mathcal{R}}(\theta) := \mathcal{R}(\theta) + \alpha\|\theta\|_1$, with $\alpha > 0$ can be written as

$$\hat{\mathcal{R}}(\theta) \approx \mathcal{R}(\theta^*) + \sum_{i=1}^{d} \left[\frac{1}{2}\lambda_i(\theta_i - \theta_i^*)^2 + \alpha|\theta_i| \right],$$

2 pts

(iii) Prove that minimizing the approximate L1-regularized cost function $\hat{\mathcal{R}}(\theta)$ has the following closed-form solution

$$\hat{\theta}_i = \text{sign}(\theta_i^*) \max\left(|\theta_i^*| - \frac{\alpha}{\lambda_i}, 0 \right),$$

You may assume that the optimal θ doesn't change its component-wise sign when regularizing, i.e. that $\text{sign}(\hat{\theta}) = \text{sign}(\theta^*)$, where $sign(\cdot)$ is the component-wise sign function assigning a vector of ± 1 values.

2 pts

(iv) We define the 'sparsity factor' of the vector θ as the number of zero components of this vector. Using the previous exercise, prove that a higher regularization factor α results in an increasing 'sparsity factor'.

1 pts

3.4 Challenges of non-convex optimization

When training a deep neural network, it is commonly observed that the gradient norm increases while the training error decreases. This behavior is illustrate in Figure 2.

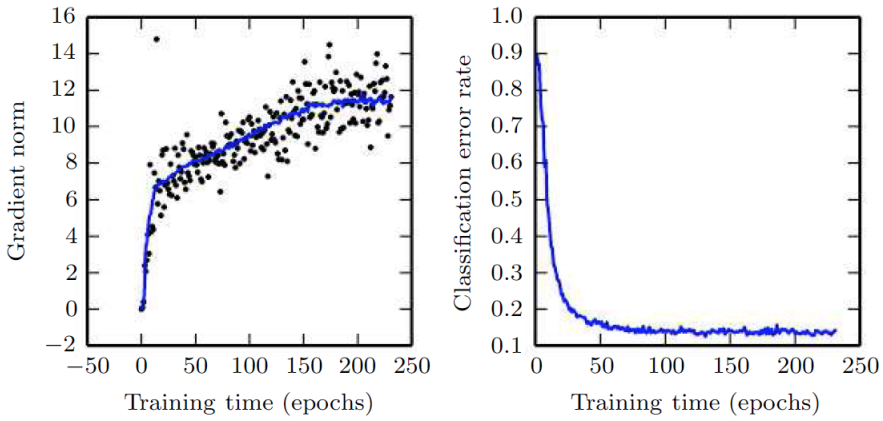
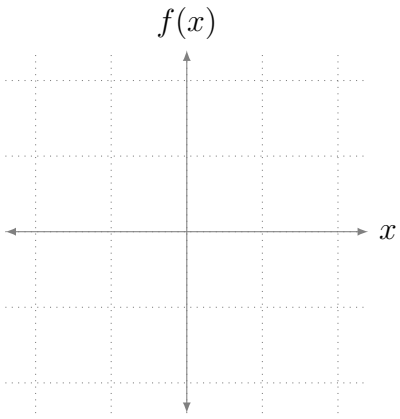


Figure 2:

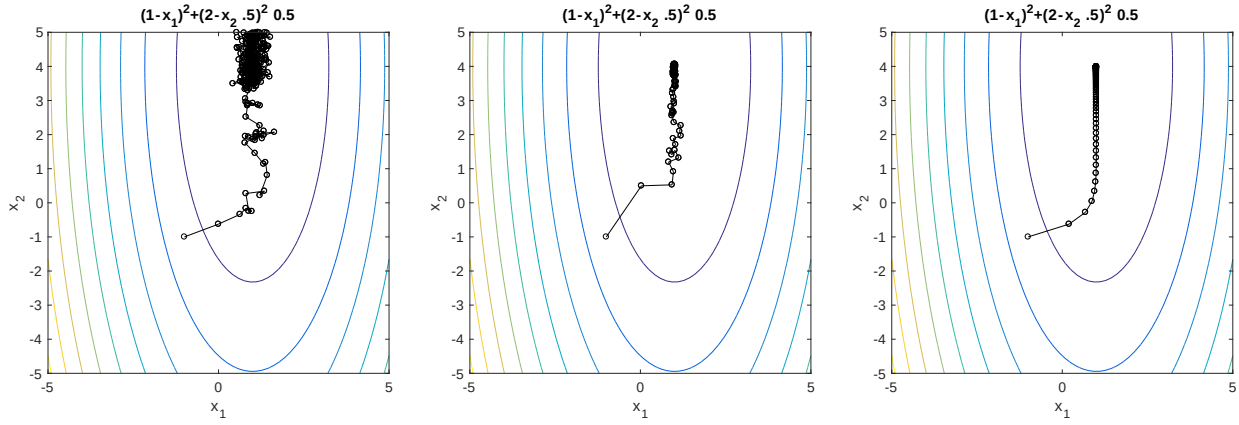
Draw a 1-D function $f(x) > 0$ with a unique global minimum for which the gradient norm increases as the function decreases.

2 pts



3.5 Stochastic Gradient Descent (SGD)

We run 3 different methods to optimize a 2-D function: (i) gradient descent, (ii) stochastic gradient descent with a constant step size, and (iii) stochastic gradient descent with decreasing step sizes. The following figure traces the steps of these methods during optimization.



(a) . . . . . (b) . . . . . (c) . . . . .

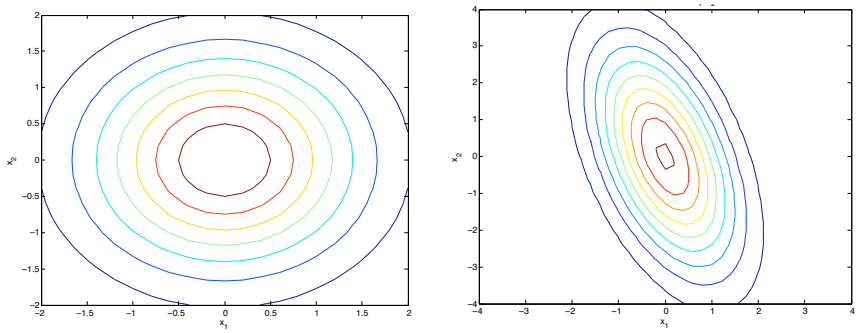
Figure 3: Convergence behavior of different optimization methods: we plot the level set of each function as well as circles to show the steps taken by each optimization method.

Map each optimization method (i-iii) to its corresponding convergence plot (a-c) in the above figure.

2 pts

3.6 Curvature challenge

Consider the two **quadratic** functions with different curvature shown in the figure below.



(a) . . . . . (b) . . . . .

Figure 4: Level sets of two quadratic functions with different curvatures.

For each function we can use one of two optimizers: (i) gradient descent and (ii) Newton's method. Considering the curvature of each function, assign each method (i-ii) to its best optimizer (a-b).

2 pts

3.7 Sharp non-linearities

Consider the objective function shown in Figure 5 with a sharp non-linearity.

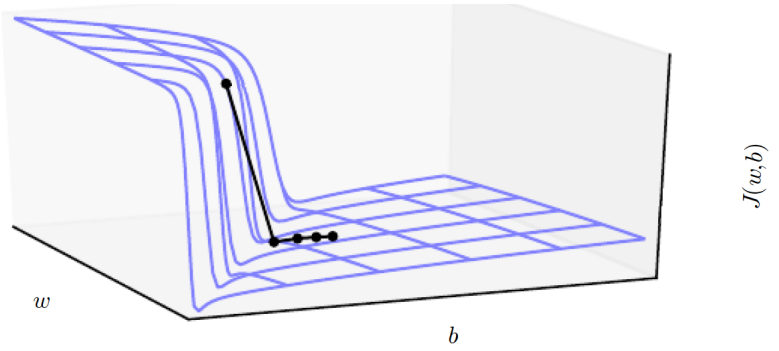


Figure 5: Objective function with a sharp non-linearity.

1. What problem can arise when using an optimization procedure that follows the gradient of this objective function?

   1 pts

   .....

   .....

2. Propose a solution to solve this problem.

   1 pts

   .....

   .....

4 CNNs, RNNs, Memory and Attention (24 pts)

4.1 CNNs

(i) Ignoring potential boundary effects, are the following statements true or false? (+1 point for correct answers, -1 point for incorrect answers, no negative total points)

1. The convolution operator commutes with the translation operator.

   [ ] True [ ] False

2. Applying the max-pooling operator before a translation is the same as applying the max-pooling operator after a translation (assume stride = 1).

   [ ] True [ ] False

3. The convolution operator is nonlinear.

   [ ] True [ ] False

3 pts

(ii) The $k \times k$ correlation operator $\times$ parameterized by $w \in \mathbb{R}^k \times \mathbb{R}^k$ is defined as follows,

$$(x \times w)_{uv} \equiv \sum_{i,j}^{k} w_{ij} x_{u+i,v+j}$$

Let $y = (x \times w)$. Suppose there is a correlation operator in a neural network whose loss function is L. Compute $\frac{dL}{dw}$. Hint: assume that you have back-propagated gradients δ.

3 pts

.....

.....

(iii) Consider a convolutional layer with a kernel of dimension $k \times k$ with f filters and an input tensor of dimension $n \times n \times d$.

(i) What is the dimension of the output layer if padding is zero and stride is one?

.....

(ii) What is the dimension of the output layer with padding p and stride s?

.....

(iii) How many parameters are in this layer (including biases)?

. . . . . .

3 pts ☐

(iv) Show that the function $f(x) = \mathrm{ReLU}(\mathrm{Conv}(x, w) + b)$ is convex.

2 pts ☐

. . . . . .

. . . . . .

. . . . . .

4.2 Recurrent Neural Networks (RNN)

(i) Consider the problem of sequence modeling. What can a RNN do that a fully-convolutional 3-layer CNN (parameters: filter size 3x3, stride 1x1, no padding) can not do?

2 pts ☐

. . . . . .

. . . . . .

. . . . . .

(ii) We want to look at back-propagation through time. We consider an RNN with teacher forcing during training, where the ground truth of the previous step $t-1$ is fed as an additional input to compute the next hidden state h_t. We define the RNN with inputs $x_{1:T}$, computed outputs $\hat{y}_{1:T}$ and ground-truth outputs $y_{1:T}$.

$$
\begin{aligned}
a_t &= F(x_t, h^{(t-1)}, y^{(t-1)}; \theta) \\
h_t &= \sigma(a_t) \\
\hat{y}_t &= G(h_t; \phi) \\
L_t &= H(\hat{y}_t, y_t) \\
L &= \sum_{t=1}^{T} L_t
\end{aligned}
$$

where F, G, and H are smooth vector valued functions, L is a smooth function into the reals, σ is an elementwise nonlinearity, θ and ϕ are parameter vectors and the initial state h_0 is given.

Calculate $\frac{\partial L}{\partial \theta}$, 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.

5 pts ☐

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

. . . . . .

4.3 Memory and Attention

(i) State *exactly* one problem that LSTMs are meant to solve, compared to naive RNNs. Briefly explain how it solves this problem.

2 pts ☐

. . . . . .

. . . . . .

. . . . . .

(ii) How many gates does a GRU cell have? Name them. Same for an LSTM cell.

1 pts ☐

. . . . . .

. . . . . .

(iii) Name *exactly* one advantage that GRUs have over LSTMs.

1 pts ☐

. . . . . .

. . . . . .

(iv) A Neural Turing Machine (NTM) reads and writes from memory, and is said to possess a differentiable memory. But with respect to what is it differentiable?

2 pts ☐

. . . . . .

. . . . . .

5 Factor Models, Autoencoders & Generative Models

(24 pts)

5.1 Factor Analysis (Maximum Likelihood Estimation)

(i) Given a set of random variables $x_{i=\{1,2,\ldots,n\}}$, each sampled from a normal distribution $f(x_i; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_i-\mu)^2}{2\sigma^2}}$ with mean μ and variance σ^2, derive the maximum likelihood estimators of the mean $\hat{\mu}$ and the variance $\hat{\sigma}^2$.

4 pts

(ii) The expectation of a vector $x \in \mathbb{R}^n$ under a Gaussian distribution is given by: $\mathbb{E}(x) = \mu$, where μ is the mean of the Gaussian distribution. The second-order moments of the Gaussian distribution in matrix form are given by $\mathbb{E}(xx^T)$. Show that $\mathbb{E}(xx^T) = \mu\mu^T + \Sigma$, where $\Sigma = cov(x)$ is the covariance matrix and is defined as:

$$cov(x) = \mathbb{E}\left(\big(x - \mathbb{E}(x)\big)\big(x - \mathbb{E}(x)\big)^T\right) \tag{13}$$

3 pts

5.2 Latent Variables

Given two variables x and y with joint distribution $p(x, y)$, the expectation $\mathbb{E}(x)$ and variance $var(x)$ under the conditional distribution $p(x, y)$ are given respectively by: $\mathbb{E}(x) = \mathbb{E}_y\left(\mathbb{E}_x(x|y)\right)$ and $var(x) = \mathbb{E}_y(var_x(x|y)) + var_y(\mathbb{E}_x(x|y))$.
Under a Probabilistic PCA model, the conditional distribution of the observed variable $x \in \mathbb{R}^d$ over the latent variable z (corresponding to the principal-component subspace) is defined as: $p(x|z) = \mathcal{N}(x|Wz + \mu, \sigma^2 I)$, where the mean of x is a linear function of z defined with the matrix $W \in \mathbb{R}^{d\times m}$ and the vector $\mu \in \mathbb{R}^d$. The prior distribution over z is given by the zero-mean and unit-covariance gaussian distribution: $p(z) = \mathcal{N}(z|O, I)$. Show that for the probabilistic PCA model:

- $\mathbb{E}(x) = \mathbb{E}_z\big(\mathbb{E}_x(x|z)\big) = \mu$
- $cov(x) = \mathbb{E}_z(cov_x(x|z)) + cov_z(\mathbb{E}_x(x|z)) = WW^T + \sigma^2 I$

3 pts ☐

. . . . . .

. . . . . .

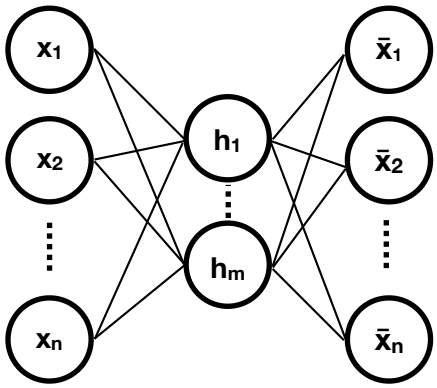
. . . . . .

. . . . . .

. . . . . .

5.3 Linear and Contractive Auto-Encoders

Consider an auto-encoder with input $x \in \mathbb{R}^n$, hidden values $h = f\big(z(x)\big)$ where $z(x) = Wx$, and outputs $\bar{x} = g(Vh)$. The activation functions are $f()$ and $g()$, while the weight matrices are given by $W \in \mathbb{R}^{n\times m}$ and $V \in \mathbb{R}^{m\times n}$.The following figure represents such auto-encoder.



(i) Show that a linear auto-encoder has the same objective function as PCA.

2 pts

. . . . . .

. . . . . .

. . . . . .

(ii) Given a training set D_n, the objective function of a contractive auto-encoder (CAE) is:

$$\sum_{x \in D_n} \left(L\Big(x, g\big(V \cdot f(Wx)\big)\Big) + \lambda \left\| J_h(x) \right\|_2^F \right) \tag{14}$$

where the regularization term is the sum of squares of all partial derivatives of the hidden extracted features with respect to the input dimensions:

$$\left\| J_h(x) \right\|_2^F = \sum_{ij} \left(\frac{\partial h_j(x)}{\partial x_i} \right)^2 \tag{15}$$

Using the chain-rule and assuming that the activation-function $f(x) = \frac{1}{1+e^{-x}}$ is a sigmoid, compute the Jacobian of the hidden layer w.r.t to the input x: $\left(\frac{\partial h_j(x)}{\partial x_i}\right)^2$

3 pts

. . . . . .

. . . . . .

. . . . . .

. . . . . .

5.4 Variational Auto-Encoders

We propose to use the following generative process to model some data $x \in \mathbb{R}^d$:

- A latent code $z \in \mathbb{R}^k$ is sampled from a standard normal $p_0(z) \sim \mathcal{N}(0, I)$.
- Our deterministic generative model $p(x|z)$ maps the latent code z to a distribution over the data x (p will be realized by a neural network).

Throughout the following steps we develop a likelihood-based training procedure for our model.

(i) Give exactly one benefit of modeling z as a stochastic variable.

1 pts

. . . . . .

(ii) We do not know z for a given x in our training data. In order to perform likelihood-based training, we will marginalize over z and introduce an *inference mechanism* $q(z|x)$. For now we will only assume that q defines a valid conditional distribution over z.

Derive a lower bound on the data likelihood (ELBO) by showing that

$$p(x) \geq \mathbb{E}_{q(z|x)}[\log p(x|z)] - D^{\mathsf{KL}}(q(z|x)||p_0(z)) \tag{16}$$

Give hints on what you are doing in every step.

4 pts

. . . . . .

. . . . . .

. . . . . .

. . . . . .

(iii) Why can't we just apply Bayes rule and use $p(z|x)$ as implied by our model above?

1 pts

. . . . . .

(iv) Let's say we choose a more complex, non-Gaussian, distribution for q. Putting the question of parametrization aside, what mathematical benefit might we loose?

1 pts

. . . . . .

. . . . . .

(v) Let p be parametrized by parameters θ and q be parametrized by parameters ζ. By maximizing the ELBO, we maximize the first term in (16) and minimize the KL-term. During training you observe that you managed to minimize the KL-term to zero.

What does this imply for the gradients ∇_θ and ∇_ζ ?

2 pts

. . . . . .

. . . . . .