Answer booklet for students

February 8, 2026

Answer booklet

This document accompanies the book Understanding Deep Learning. It contains answers to a selected subset of the problems at the end of each chapter of the main book. The remaining answers are available only to instructors via the MIT Press.

This booklet has not yet been checked very carefully. I really need your help in this regard and I'd be very grateful if you would mail me at udlbookmail@gmail.com if you cannot understand the text or if you think that you find a mistake. Suggestions for extra problems will also be gratefully received!

Simon Prince
February 8, 2026

Chapter 2

Supervised learning

Problem 2.1 To walk "downhill" on the loss function (equation 2.5), we measure its gradient with respect to the parameters ϕ_0 and ϕ_1. Calculate expressions for the slopes $\partial L/\partial\phi_0$ and $\partial L/\partial\phi_1$.

Problem 2.2 Show that we can find the minimum of the loss function in closed-form by setting the expression for the derivatives from problem 2.1 to zero and solving for ϕ_0 and ϕ_1.

Problem 2.3 Consider reformulating linear regression as a generative model so we have $x = \mathrm{g}[y, \boldsymbol{\phi}] = \phi'_0 + \phi'_1 y$. What is the new loss function? Find an expression for the inverse function $y = \mathrm{g}^{-1}[x, \boldsymbol{\phi}']$ that we would use to perform inference. Will this model make the same predictions as the discriminative version for a given training dataset $\{x_i, y_i\}$? One way to establish this is to write code that fits a line to three data points using both methods and see if the result is the same.

Answer

We can trivially solve for x by rearranging the equation:

$$x = \frac{y - \phi'_0}{\phi'_1}.$$

The answers are not the same; you can show this by finding equations for the slope and intercept of the function of line relating x to y and showing they are not the same. Or you can just try fitting both models to some data. An example is given in figure 2.1 and the fitted lines are slightly different.

For an intuitive argument to why this is the case, consider fitting a line through two points. Then we'll add a third point. The squared distance to this new point vertically (discriminative case) will not be the same as the squared distance to this point horizontally (generative case), and so the new point will change the original line to greater or lesser degree in the two cases.

In the discriminative case, we assume that the errors are in the output y. In the generative case, we assume that the errors are in the input, x.

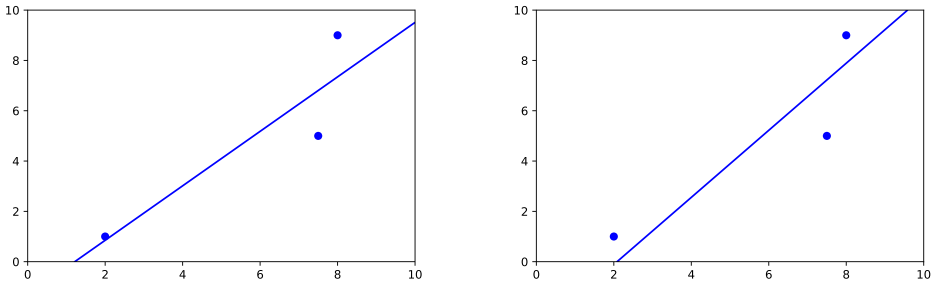


Figure 2.1 Answer to problem 2.3. The original discriminative formulation is on the left and the generative formulation is on the right. The curves are not the same.

Chapter 3

Shallow neural networks

Problem 3.1 What kind of mapping from input to output would be created if the activation function in equation 3.1 was linear so that $a[z] = \psi_0 + \psi_1 z$? What kind of mapping would be created if the activation function was removed, so $a[z] = z$?

Problem 3.2 For each of the four linear regions in figure 3.3j, indicate which hidden units are inactive and which are active (i.e., which do and do not clip their inputs).

Problem 3.3 Derive expressions for the positions of the "joints" in function in figure 3.3j in terms of the ten parameters $\boldsymbol{\phi}$ and the input x. Derive expressions for the slopes of the four linear regions.

Answer

The joints correspond to where the three hidden units cross the x axis, which is where $\theta_{\bullet 0} + \theta_{\bullet 1}x = 0$. Hence, the joints are at $-\theta_{10}/\theta_{11}, -\theta_{20}/\theta_{21}, -\theta_{30}/\theta_{31}$.

The final slopes depend on which hidden units are active and which are not. The slopes for the four regions are:

- $\theta_{31}\phi_3$
- $\theta_{11}\phi_1 + \theta_{31}\phi_3$
- $\theta_{11}\phi_1 + \theta_{21}\phi_2 + \theta_{31}\phi_3$
- $\theta_{11}\phi_1 + \theta_{21}\phi_2$

Problem 3.4 Draw a version of figure 3.3 (from book) where the y-intercept and slope of the third hidden unit have changed as in figure 3.14 (from book). Assume that the remaining parameters remain the same.

Problem 3.5 Prove that the following property holds for $\alpha \in \mathbb{R}^+$:

$$\text{ReLU}[\alpha \cdot z] = \alpha \cdot \text{ReLU}[z].$$

This is known as the *non-negative homogeneity* property of the ReLU function.

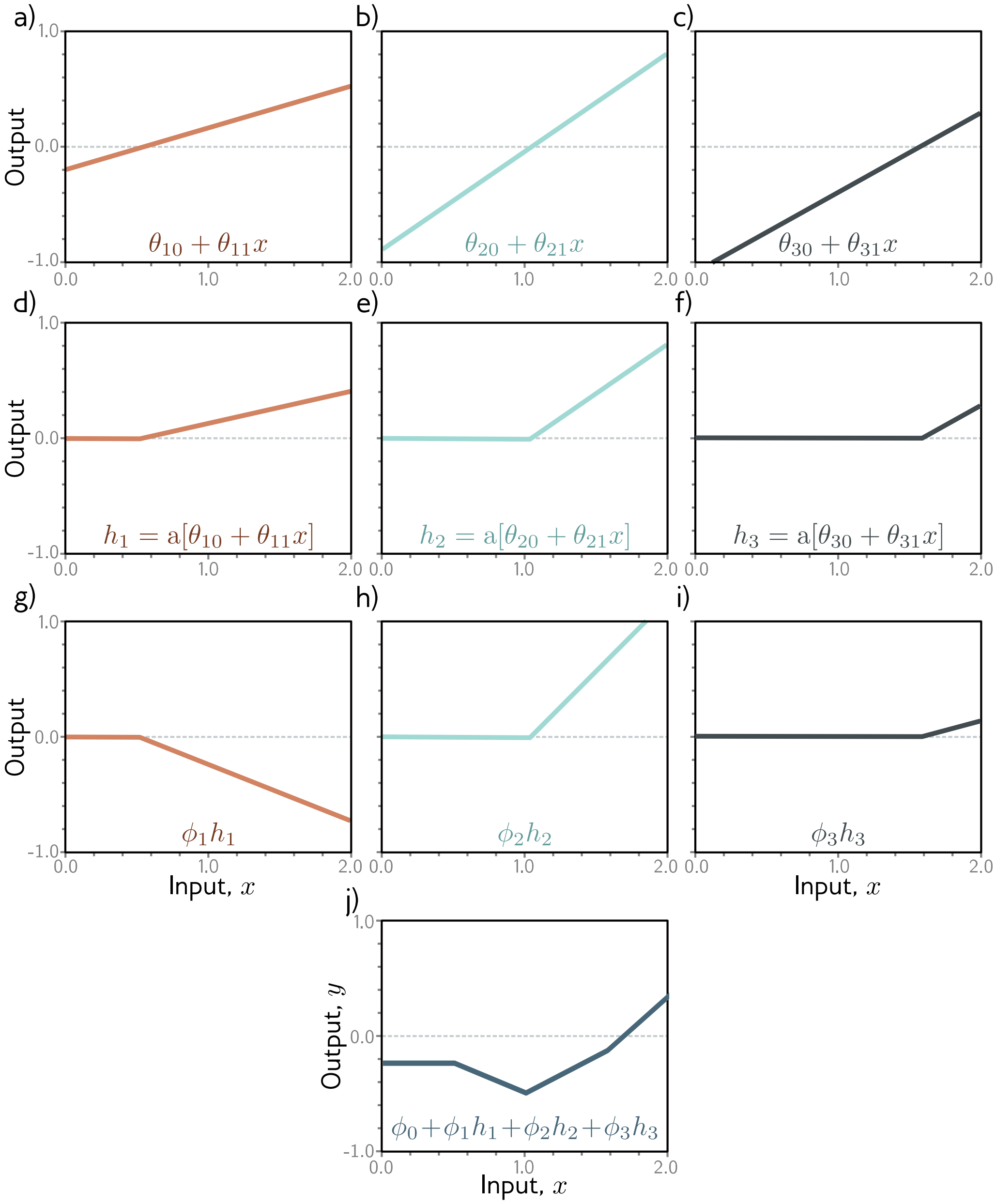


Figure 3.1 Answer to problem 3.4

Problem 3.6 Following on from problem 3.5, what happens to the shallow network defined in equations 3.3 and 3.4 when we multiply the parameters θ_{10} and θ_{11} by a positive constant α and divide the slope ϕ_1 by the same parameter α? What happens if α is negative?

Problem 3.7 Consider fitting the model in equation 3.1 using a least squares loss function. Does this loss function have a unique minimum? i.e., is there is a single "best" set of parameters?

Problem 3.8 Consider replacing the ReLU activation function with (i) the Heaviside step function heaviside[z], (ii) the hyperbolic tangent function tanh[z], and (iii) the rectangular function rect[z], where:

$$\text{heaviside}[z] = \begin{cases} 0 & z < 0 \\ 1 & z \geq 0 \end{cases},$$

$$\text{rect}[z] = \begin{cases} 0 & z < 0 \\ 1 & 0 \leq z \leq 1 \\ 0 & z > 1 \end{cases}.$$

Redraw a version of figure 3.3 for each of these functions. The original parameters were: $\boldsymbol{\phi} = \{\phi_0, \phi_1, \phi_2, \phi_3, \theta_{10}, \theta_{11}, \theta_{20}, \theta_{21}, \theta_{30}, \theta_{31}\} = \{-0.23, -1.3, 1.3, 0.66, -0.2, 0.4, -0.9, 0.9, 1.1, -0.7\}$. Provide an informal description of the family of functions that can be created by neural networks with one input, three hidden units, and one output for each activation function.

Problem 3.9 Show that the third linear region in figure 3.3 has a slope that is the sum of the slopes of the first and fourth linear regions.

Answer

You can see this from the answer to problem 3.3.

Problem 3.10 Consider a neural network with one input, one output, and three hidden units. The construction in figure 3.3 shows how this creates four linear regions. Under what circumstances could this network produce a function with fewer than four linear regions?

Problem 3.11 How many parameters does the model in figure 3.6 have?

Answer

It has $1 \times 4 + 4 \times 2 = 12$ slopes and $4 + 2 = 6$ intercepts giving a total of 18 parameters

Problem 3.12 How many parameters does the model in figure 3.7 have?

Problem 3.13 What is the activation pattern for each of the seven regions in figure 3.8j? In other words, which hidden units are active (pass the input) and which are inactive (clip the input) for each region?

Problem 3.14 Write out the equations that define the network in figure 3.11. There should be three equations to compute the three hidden units from the inputs and two equations to compute the outputs from the hidden units.

Problem 3.15 What is the maximum possible number of 3D linear regions that can be created by the network in figure 3.11?

Answer

This is the number of regions created by the intersections of three planes in 3D space. The maximum number is eight as in figure 3.10c.

Problem 3.16 Write out the equations for a network with two inputs, four hidden units, and three outputs. Draw this model in the style of figure 3.11.

Problem 3.17 Equations 3.11 and 3.12 define a general neural network with D_i inputs, one hidden layer containing D hidden units, and D_o outputs. Find an expression for the number of parameters in the model in terms of D_i, D, and D_o.

Answer

There are $D_i \times D$ slopes connecting the input to the hidden units and $D \times D_o$ slopes connecting the hidden units to the outputs. There are D intercepts feeding into the hidden units and D_o intercepts feeding into the outputs. The total expression is hence:

$$(D_i + 1) \times D + (D + 1) \times D_o$$

Problem 3.18 Show that the maximum number of regions created by a shallow network with $D_i = 2$ dimensional input, $D_o = 1$ dimensional output, and $D = 3$ hidden units is seven as in figure 3.8j. Use the result of Zaslavsky(1975) that the maximum number of regions created by partitioning a D_i-dimensional space with D hyperplanes is $\sum_{j=0}^{D_i} \binom{D}{j}$. What is the maximum number of regions if we add two more hidden units to this model so $D = 5$?

Answer

With three hidden units and two inputs, the computation is:

$$\begin{aligned} N = \binom{3}{0} + \binom{3}{1} + \binom{3}{2} \\ &= 1 + 3 + 3 \\ &= 7 \end{aligned}$$

With five hidden units and two inputs, the computation is:

$$
\begin{aligned}
N = \binom{5}{0} + \binom{5}{1} + \binom{5}{2} \\
&= 1 + 5 + 10 \\
&= 16
\end{aligned}
$$

Chapter 4

Deep neural networks

Problem 4.1 Consider composing the two neural networks in figure 4.1. Draw a plot of the relationship between the input x and output y' for $x \in [-1, 1]$.

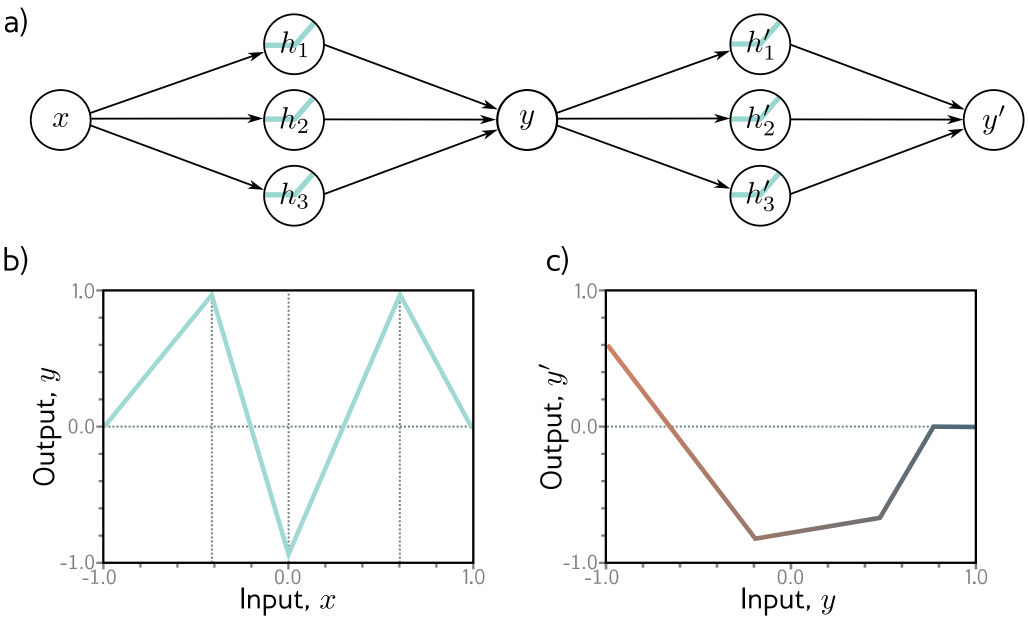


Figure 4.1 Composition of two networks for problem 4.1. a) The output y of the first network becomes the input to the second. b) The first network computes this function with output values $y \in [-1, 1]$. c) The second network computes this function on the input range $y \in [-1, 1]$.

Answer

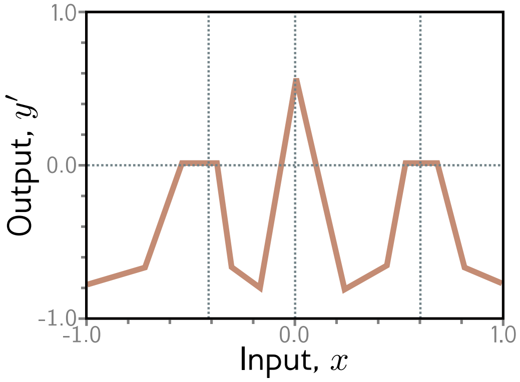


Figure 4.2 Answer to question 4.1.

The answer is depicted in figure 4.2.

Problem 4.2 Identify the hyperparameters in figure 4.6 (Hint: there are four!)

Problem 4.3 Using the non-negative homogeneity property of the ReLU function (see problem 3.5), show that:

$$\mathrm{ReLU}\left[\boldsymbol{\beta}_1+\lambda_1\cdot\boldsymbol{\Omega}_1\mathrm{ReLU}\left[\boldsymbol{\beta}_0+\lambda_0\cdot\boldsymbol{\Omega}_0\mathbf{x}\right]\right]=\lambda_0\lambda_1\cdot\mathrm{ReLU}\left[\frac{1}{\lambda_0\lambda_1}\boldsymbol{\beta}_1+\boldsymbol{\Omega}_1\mathrm{ReLU}\left[\frac{1}{\lambda_0}\boldsymbol{\beta}_0+\boldsymbol{\Omega}_0\mathbf{x}\right]\right],$$

where λ_0 and λ_1 are non-negative scalars. From this, we see that the weight matrices can be re-scaled by any magnitude as long as the biases are also adjusted, and the scale factors can be re-applied at the end of the network.

Problem 4.4 Write out the equations for a deep neural network that takes $D_i = 5$ inputs, $D_o = 4$ outputs and has three hidden layers of sizes $D_1 = 20$, $D_2 = 10$, and $D_3 = 7$, respectively in both the forms of equations 4.15 and 4.16. What are the sizes of each weight matrix $\boldsymbol{\Omega}_{\bullet}$ and bias vector $\boldsymbol{\beta}_{\bullet}$?

Problem 4.5 Consider a deep neural network with $D_i = 5$ inputs, $D_o = 1$ output, and $K = 20$ hidden layers containing $D = 30$ hidden units each. What is the depth of this network? What is the width?

Problem 4.6 Consider a network with $D_i = 1$ input, $D_o = 1$ output, $K = 10$ layers, with $D = 10$ hidden units in each. Would the number of weights increase more if we increased the depth by one or the width by one? Provide your reasoning.

Problem 4.7 Choose values for the parameters $\boldsymbol{\phi} = \{\phi_0, \phi_1, \phi_2, \phi_3, \theta_{10}, \theta_{11}, \theta_{20}, \theta_{21}, \theta_{30}, \theta_{31}\}$ for the shallow neural network in equation 3.1 that will define an identity function over a finite range $x \in [a, b]$.

Problem 4.8 Figure 4.9 shows the activations in the three hidden units of a shallow network (as in figure 3.3). The slopes in the hidden units are 1.0, 1.0, and -1.0, respectively, and the "joints" in the hidden units are at positions 1/6, 2/6, and 4/6. Find values of ϕ_0, ϕ_1, ϕ_2, and ϕ_3 that will combine the hidden unit activations as $\phi_0 + \phi_1 h_1 + \phi_2 h_2 + \phi_3 h_3$ to create a function with four linear regions that oscillate between output values of zero and one. The slope of the leftmost region should be positive, the next one negative, and so on. How many linear regions will we create if we compose this network with itself? How many will we create if we compose it with itself K times?

Answer

In the first region, only hidden unit 3 is active. We need the output to be zero when $x = 0$ and one when $x = 1/6$. This is achieved by setting $\phi_0 = 4$ and $\phi_3 = -6$. The second region has slope $\phi_1 - \phi_3$. We need it to have slope -6 to bring the function back down to zero at $x = 2/6$. Hence $\phi_1 = -12$. In the third region, the slope is $\phi_1 + \phi_2 - \phi_3$ and we need this to take the value 3, to bring the function back to one at $x = 4/6$. This implies that $\phi_2 = 9$. In the fourth region, the slope is $\phi_1 + \phi_2$, and which is -3, which appropriately brings the function back down to zero at $x = 1$.

If we compose it with itself, then we will get $4 \times 4 = 16$ regions as each of the four regions in the second network will be replicated four times. If we compose it with itself K times, we get 4^{K+1} regions.

Problem 4.9 Following problem 4.8, is it possible to create a function with three linear regions that oscillates back and forth between output values of zero and one using a shallow network with two hidden units? Is it possible to create a function with five linear regions that oscillates in the same way using a shallow network with four hidden units?

Answer

It is not possible to create a function with three linear regions that oscillates back and forth between output values of zero and one using a shallow network with two hidden units.

However, it is possible to create a function with five linear regions that oscillates in the same way using a shallow network with four hidden units, and generally, for $N \geq 3$ hidden units it's possible to make a function that oscillates back and forth $N + 1$ times.

Problem 4.10 Consider a deep neural network with a single input, a single output, and K hidden layers, each of which contains D hidden units. Show that this network will have a total of $3D + 1 + (K - 1)D(D + 1)$ parameters.

Problem 4.11 Consider two neural networks that map a scalar input x to a scalar output y. The first network is shallow and has $D = 95$ hidden units. The second is deep and has $K = 10$ layers, each containing $D = 5$ hidden units. How many parameters does each network have? How many linear regions can each network make? Which would run faster?

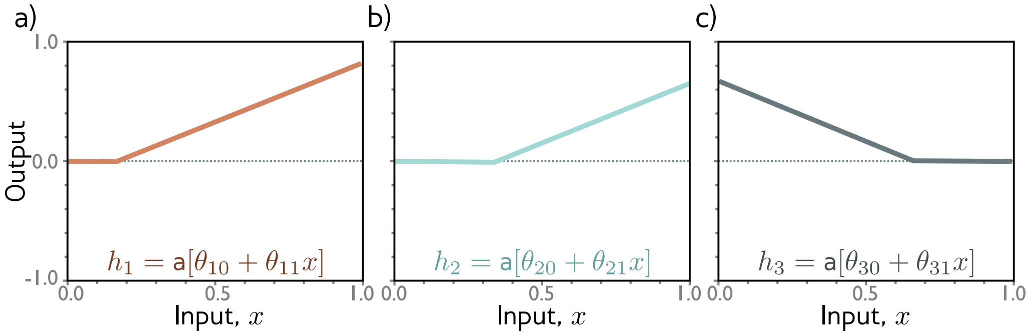


Figure 4.3 Hidden unit activations for problem 4.8. a) First hidden unit has a joint at position $x = 1/6$ and a slope of one in the active region. b) Second hidden unit has a joint at position $x = 2/6$ and a slope of one in the active region. c) Third hidden unit has a joint at position $x = 4/6$ and a slope of minus one in the active region.

Answer

Using the formula from problem 4.10, the shallow network has $3\times95+1 = 286$ parameters, and the second network has $3 \times 5 + 1 + (9 \times 5 \times 6) = 286$ parameters. They are both the same.

The shallow network can create 96 regions; since there is just one input, each hidden unit creates one joint, for a total of 95 joints separating 96 linear regions. The number of linear regions for the deep network is given by equation 4.17 and is 60,466,176.

In principle, the shallow network will be faster to run on modern hardware as the computation is more parallel.

Chapter 5

Loss functions

Problem 5.1 Show that the logistic sigmoid function $\text{sig}[z]$ maps $z = -\infty$ to 0, $z = 0$ to 0.5 and $z = \infty$ to 1 where:

$$\text{sig}[z] = \frac{1}{1 + \exp[-z]}.$$

Problem 5.2 The loss L for binary classification for a single training pair $\{\mathbf{x}, y\}$ is:

$$L = (1 - y)\log\left[1 - \text{sig}[\text{f}[\mathbf{x}, \boldsymbol{\phi}]]\right] + y \log\left[\text{sig}[\text{f}[\mathbf{x}, \boldsymbol{\phi}]]\right],$$

where $\text{sig}[\bullet]$ is defined in equation 5.1. Plot this loss as a function of the transformed network output $\text{sig}[\text{f}[\mathbf{x}, \boldsymbol{\phi}]] \in [0, 1]$ (i) when the training label $y = 0$ and (ii) when $y = 1$.

Problem 5.3 Suppose we want to build a network that predicts the direction y in radians of the prevailing wind based on local measurements of barometric pressure $\mathbf{x}$. A suitable distribution over circular domains is the von Mises distribution (figure 5.13 from book):

$$Pr(y|\mu, \kappa) = \frac{\exp\big[\kappa \cos[y - \mu]\big]}{2\pi \cdot \text{Bessel}_0[\kappa]},$$

where μ is a measure of the mean direction and κ is a measure of the concentration (i.e., the inverse of the variance). The term $\text{Bessel}_0[\kappa]$ is a modified Bessel function of order 0.

Use the recipe from section 5.2 to develop a loss function for learning the parameter μ of a model $\text{f}[\mathbf{x}, \boldsymbol{\phi}]$ to predict the most likely wind direction. Your solution should treat the concentration κ as constant. How would you perform inference?

Answer

$$\begin{aligned} L &= \sum_{i=1}^{I} -\log\left[\frac{\exp[\kappa\cos[y_i - \text{f}[\mathbf{x}_i, \boldsymbol{\phi}]]]}{2\pi \cdot \text{Bessel}_0[\kappa]}\right] \\ &= \sum_{i=1}^{I}\Big(-\kappa\cos[y_i - \text{f}[\mathbf{x}_i, \boldsymbol{\phi}]] + \log\left[\text{Bessel}_0[\kappa]\right]\Big). \end{aligned}$$

which can be simplified to:

$$L = \sum_{i=1}^{I} -\cos[y_i - \mathrm{f}[\mathbf{x}_i, \boldsymbol{\phi}]],$$

by removing constant additive and multiplicative factors.

To perform inference you would take the maximum of the distribution which is just the predicted parameter μ. This may be out of the range $[-\pi, \pi]$, in which case we would add / remove multiples of 2π until it is in the original range.

Problem 5.4 Sometimes, the predictions y for a given input $\mathbf{x}$ are naturally multimodal as in figure 5.14a from book; there is more than one valid prediction for a given input. In this case, we might use a weighted sum of normal distributions as the distribution over the output. This is known as a *mixture of Gaussians* model . For example, a mixture of two Gaussians has distribution parameters $\boldsymbol{\theta} = \{\lambda, \mu_1, \sigma_1^2, \mu_2, \sigma_2^2\}$ is defined by:

$$Pr(y|\lambda, \mu_1, \mu_2, \sigma_1^2, \sigma_2^2) = \frac{\lambda}{\sqrt{2\pi\sigma_1^2}} \exp\left[\frac{-(y-\mu_1)^2}{2\sigma_1^2}\right] + \frac{1-\lambda}{\sqrt{2\pi\sigma_2^2}} \exp\left[\frac{-(y-\mu_2)^2}{2\sigma_2^2}\right],$$

where $\lambda \in [0, 1]$ controls the relative weight of the two normal distributions, which have means μ_1, μ_2 and variances σ_1^2, σ_2^2, respectively. This model can represent a distribution with two peaks (figure 5.14b from book) or a distribution with one peak but a more complex shape (figure 5.14c from book).

Use the recipe from section 5.2 to construct a loss function for training a model $\mathbf{f}[x, \boldsymbol{\phi}]$ that takes input x, has parameters $\boldsymbol{\phi}$, and predicts a mixture of two Gaussians. The loss should be based on I training data pairs $\{x_i, y_i\}$. What possible problems do you foresee when we perform inference in this model?

Answer

The loss function can be written as:

$$L = -\sum_{i=1}^{I} \log\left[\frac{\mathrm{sig}[\mathrm{f}_1[\mathbf{x}_i, \boldsymbol{\phi}]]}{\sqrt{2\pi \mathrm{f}_3[\mathbf{x}_i, \boldsymbol{\phi}]^2}} \exp\left[\frac{-(y_i - \mathrm{f}_2[\mathbf{x}_i, \boldsymbol{\phi}])^2}{2\mathrm{f}_3[\mathbf{x}_i, \boldsymbol{\phi}]^2}\right] + \frac{1-\mathrm{sig}[\mathrm{f}_1[\mathbf{x}_i, \boldsymbol{\phi}]]}{\sqrt{2\pi \mathrm{f}_5[\mathbf{x}_i, \boldsymbol{\phi}]^2}} \exp\left[\frac{-(y_i - \mathrm{f}_4[\mathbf{x}_i, \boldsymbol{\phi}])^2}{2\mathrm{f}_5[\mathbf{x}_i, \boldsymbol{\phi}]^2}\right]\right]$$

Inference is slightly more tricky as there is no simple closed form expression for the mode of this distribution. We might have to find it using optimization. See `http://www.cs.toronto.edu/~miguel/research/GMmodes.html` for further details.

Problem 5.5 Consider extending the model from problem 5.3 to predict the wind direction using a mixture of two von Mises distributions. Write an expression for the likelihood $Pr(y|\boldsymbol{\theta})$ for this model. How many outputs will the network need to produce?

Problem 5.6 Consider building a model to predict the number of pedestrians $y \in \{0, 1, 2, \ldots\}$ that will pass a given point in the city in the next minute, based on data $\mathbf{x}$ that contains information about the time of day, the longitude and latitude, and the type of neighborhood. A suitable distribution for modeling counts is the Poisson distribution (figure 5.15 from book). This has a single parameter $\lambda > 0$ called the *rate* that represents the mean of the distribution. The distribution has probability density function:

$$Pr(y = k) = \frac{\lambda^k e^{-\lambda}}{k!}.$$

Use the recipe in section 5.2 to design a loss function for this model assuming that we have access to I training pairs $\{\mathbf{x}_i, y_i\}$.

Problem 5.7 Consider a multivariate regression problem where we predict 10 outputs so $\mathbf{y} \in \mathbb{R}^{10}$ and model each with an independent normal distribution where the means μ_d are predicted by the network, and variances σ^2 are all the same. Write an expression for the likelihood $Pr(\mathbf{y}|\mathbf{f}[\mathbf{x}, \boldsymbol{\phi}])$ for this model. Show that minimizing the negative log-likelihood of this model is still equivalent to minimizing a sum of squared terms if we don't estimate the variance σ^2.

Problem 5.8 Construct a loss function for making multivariate predictions $\mathbf{y}$ based on independent normal distributions with different variances σ_d^2 for each dimension. Assume that this is a heteroscedastic model so that both the means μ_d and variances σ_d^2 change as a function of the data.

Answer

$$L = -\sum_{i=1}^{I} \log \left[\prod_{d=1}^{D_o} \frac{1}{\sqrt{2\pi \mathrm{f}_{2d}[\mathbf{x}_i, \boldsymbol{\phi}]^2}} \exp\left[-\frac{(y_{id} - \mathrm{f}_{1d}[\mathbf{x}_i, \boldsymbol{\phi}])^2}{2\mathrm{f}_{2d}[\mathbf{x}_i, \boldsymbol{\phi}]^2} \right] \right].$$

where $\mathrm{f}_{1d}[\mathbf{x}_i, \boldsymbol{\phi}]$ predicts the mean of the $d^t h$ dimension and $\mathrm{f}_{2d}[\mathbf{x}_i, \boldsymbol{\phi}]^2$ predicts the variance.

Problem 5.9 Consider a multivariate regression problem in which we predict the height of an individual in meters and their weight in kilos from some data $\mathbf{x}$. Here, the units take quite different values. What problems do you see this causing? Propose two solutions to these problems.

Answer

The height uses different units than the weight and the numbers will be smaller. Consequently, the least squares loss will focus much more on the weight than the height. Two possible solutions are (i) to rescale the outputs so that they have the same standard deviation, build a model that predicts the rescaled outputs, and scale them back after inference or (ii) learn a separate variance for the two dimensions so that the model can automatically take care of this. The second approach can be done in either a homoscedastic or heteroscedastic context.

Problem 5.10 Extend the model from problem 5.3 to predict both the wind direction and the wind speed and define the associated loss function.

Chapter 6

Fitting models

Problem 6.1 Show that the derivatives of the least squares loss function in equation 6.5 are given by the expressions in equation 6.7.

Problem 6.2 A surface is convex if the eigenvalues of the Hessian $\mathbf{H}[\boldsymbol{\phi}]$ are positive everywhere. In this case, the surface has a unique minimum, and optimization is easy. Find an algebraic expression for the Hessian matrix,

$$\mathbf{H}[\boldsymbol{\phi}] = \begin{bmatrix} \frac{\partial^2 L}{\partial \phi_0^2} & \frac{\partial^2 L}{\partial \phi_0 \partial \phi_1} \\ \frac{\partial^2 L}{\partial \phi_1 \partial \phi_0} & \frac{\partial^2 L}{\partial \phi_1^2} \end{bmatrix},$$

for the linear regression model (equation 6.5). Prove that this function is convex by showing that the eigenvalues are always positive. This can be done by showing that both the trace and the determinant of the matrix are positive.

Problem 6.3 Compute the derivatives of the least squares loss $L[\boldsymbol{\phi}]$ with respect to the Gabor model (equation 6.8).

Problem 6.4 The logistic regression model uses a linear function to predict which of two classes $y \in \{0, 1\}$ an input $\mathbf{x}$ belongs to. For a 1D input and a 1D output, it has two parameters, ϕ_0 and ϕ_1, and is defined by:

$$Pr(y = 1|x) = \text{sig}[\phi_0 + \phi_1 x],$$

where sig[•] is the logistic sigmoid function:

$$\text{sig}[z] = \frac{1}{1 + \exp[-z]}.$$

(i) Plot y against x for this model for different values of ϕ_0 and ϕ_1 and explain the qualitative meaning of each parameter. (ii) What is a suitable loss function for this model? (iii) Compute the derivatives of this loss function with respect to the parameters. (iv) Generate ten data points from a normal distribution with mean -1 and standard deviation 1 and assign the label $y = 0$ to them. Generate another ten data points from a normal distribution with mean 1 and standard deviation 1 and assign the label $y = 1$ to these. Plot the loss as a heatmap in terms of the two parameters ϕ_0 and ϕ_1. (v) Is this loss function convex? How could you prove this?

Answer

(i) The result looks like a sigmoid function which shifts to the left as we increase ϕ_0 and gets steeper as we increase ϕ_1.

(ii) The binary cross entropy loss:

$$L[\boldsymbol{\phi}] = \sum_{i=1}^{I} -(1-y_i)\log\Bigl[1-\text{sig}[\phi_0+\phi_1 x]\Bigr] - y_i \log\Bigl[\text{sig}[\phi_0+\phi_1 x]\Bigr]$$

(iii) The derivatives of the sigmoid function is:

$$\frac{\partial \text{sig}[z]}{\partial z} = \frac{\exp[-z]}{(1+\exp[-z])^2}$$

It follows that the derivatives of the loss function are:

$$\begin{aligned}
\frac{\partial L}{\partial \phi_0} &= \sum_{i=1}^{I}\left(\frac{1-y_i}{1-\text{sig}[\phi_0+\phi_1 x_i]} - \frac{y_i}{\text{sig}[\phi_0+\phi_1 x_i]}\right)\frac{\exp[-\phi_0-\phi_1 x_i]}{(1+\exp[-\phi_0-\phi_1 x_i)^2}\\
\frac{\partial L}{\partial \phi_1} &= \sum_{i=1}^{I}\left(\frac{1-y_i}{1-\text{sig}[\phi_0+\phi_1 x_i]} - \frac{y_i}{\text{sig}[\phi_0+\phi_1 x_i]}\right)\frac{x_i\cdot\exp[-\phi_0-\phi_1 x_i]}{(1+\exp[-\phi_0-\phi_1 x_i)^2}
\end{aligned}$$

(iv) The heatmap is shown in figure 6.1 (will differ depending on random numbers drawn).

(v) It is convex as can be seen in figure 6.1. You could prove this by examining the Hessian matrix.

Problem 6.5 Compute the derivatives of the least squares loss with respect to the ten parameters of the simple neural network model introduced in equation 3.1:

$$\text{f}[x,\boldsymbol{\phi}] = \phi_0 + \phi_1\text{a}[\theta_{10}+\theta_{11}x] + \phi_2\text{a}[\theta_{20}+\theta_{21}x] + \phi_3\text{a}[\theta_{30}+\theta_{31}x].$$

Think carefully about what the derivative of the ReLU function $a[\bullet]$ will be.

Answer

The derivative of a least squares loss for function $\text{f}[x,\boldsymbol{\phi}]$ is given by:

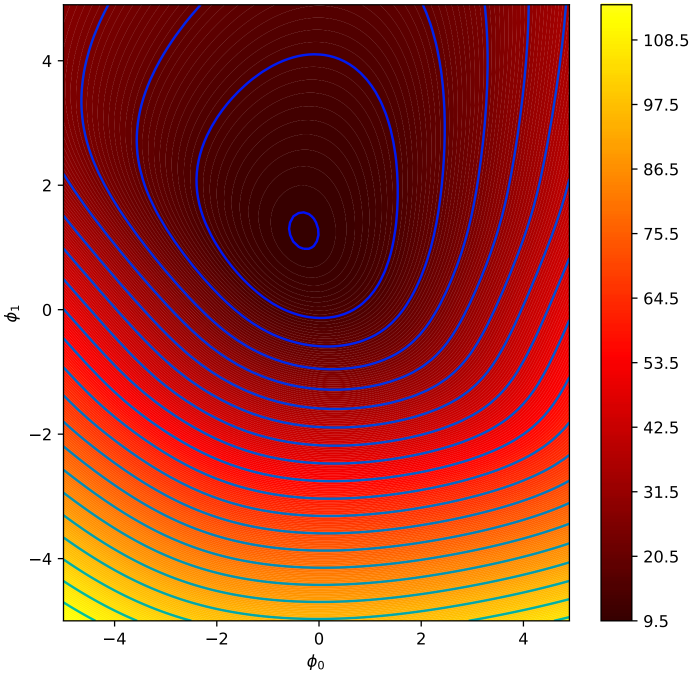


Figure 6.1 Loss function for logistic regression model in 6.4.

$$\frac{\partial L}{\partial \phi_j} = -2\sum_i (y - \text{f}[x_i, \boldsymbol{\phi}])\frac{\partial \text{f}[x_i, \boldsymbol{\phi}]}{\partial \phi_j}.$$

The derivative of the ReLU is zero if the input z is less than zero and one if the input z is greater than zero, which we can write as $\mathbb{I}[z > 0]$. The derivative terms can hence be written as:

$$
\begin{aligned}
\frac{\partial \mathrm{f}[x_i, \boldsymbol{\phi}]}{\partial \phi_0} &= 1 \\
\frac{\partial \mathrm{f}[x_i, \boldsymbol{\phi}]}{\partial \phi_1} &= \mathrm{a}[\theta_{10} + \theta_{11} x_i] \\
\frac{\partial \mathrm{f}[x_i, \boldsymbol{\phi}]}{\partial \phi_2} &= \mathrm{a}[\theta_{20} + \theta_{21} x_i] \\
\frac{\partial \mathrm{f}[x_i, \boldsymbol{\phi}]}{\partial \phi_3} &= \mathrm{a}[\theta_{30} + \theta_{31} x_i]
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial \mathrm{f}[x_i, \boldsymbol{\phi}]}{\partial \theta_{10}} &= \phi_1 \cdot \mathbb{I}[\theta_{10} + \theta_{11} x_i > 0] \\
\frac{\partial \mathrm{f}[x_i, \boldsymbol{\phi}]}{\partial \theta_{11}} &= \phi_1 \cdot x_i \cdot \mathbb{I}[\theta_{10} + \theta_{11} x_i > 0] \\
\frac{\partial \mathrm{f}[x_i, \boldsymbol{\phi}]}{\partial \theta_{20}} &= \phi_2 \cdot \mathbb{I}[\theta_{20} + \theta_{21} x_i > 0] \\
\frac{\partial \mathrm{f}[x_i, \boldsymbol{\phi}]}{\partial \theta_{21}} &= \phi_2 \cdot x_i \cdot \mathbb{I}[\theta_{20} + \theta_{21} x_i > 0] \\
\frac{\partial \mathrm{f}[x_i, \boldsymbol{\phi}]}{\partial \theta_{30}} &= \phi_3 \cdot \mathbb{I}[\theta_{30} + \theta_{31} x_i > 0] \\
\frac{\partial \mathrm{f}[x_i, \boldsymbol{\phi}]}{\partial \theta_{31}} &= \phi_3 \cdot x_i \cdot \mathbb{I}[\theta_{30} + \theta_{31} x_i > 0]
\end{aligned}
$$

Problem 6.6 Which of the functions in figure 6.11 from the book is convex? Justify your answer. Characterize each of the points 1-7 as (i) a local minimum, (ii) the global minimum, or (iii) neither.

Problem 6.7 The gradient descent trajectory for path 1 in figure 6.5a oscillates back and forth inefficiently as it moves down the valley toward the minimum. It's also notable that it turns at right angles to the previous direction at each step. Provide a qualitative explanation for these phenomena. Propose a solution that might help prevent this behavior.

Answer

The direction must turn at right angles – if the slope was still moving downhill at all in the current direction, then we should continue moving forward in the previous stage. The oscillation happens because the trajectory slightly overshoots the center of a descending valley. It has to turn at right angles, and then overshoots again in the other direction and so on. There are many ways to avoid this behaviour including using the Newton method which takes account of the second derivative, or using a momentum term, which averages out these oscillations.

Problem 6.8 Can (non-stochastic) gradient descent with a *fixed* learning rate escape local minima?

Answer

Yes! The distance moved just depends on the gradient at the current point and the learning rate. The movement pays no attention to whether it crosses from valley to valley. Usually, the learning rate is very small though, so this may not actually happen in practice.

Problem 6.9 We run the stochastic gradient descent algorithm for 1,000 iterations on a dataset of size 100 with a batch size of 20. How many epochs are we running the algorithm for?

Problem 6.10 Show that the momentum term $\mathbf{m}_t$ (equation 6.11) is an infinite weighted sum of the gradients at the previous iterations and derive an expression for the coefficients (weights) of that sum.

Problem 6.11 What dimensions will the Hessian have if the model has one million parameters?

Chapter 7

Gradients and initialization

Problem 7.1 A two-layer network with two hidden units in each layer can be defined as:

$$\begin{aligned} y \;\; = \;\; & \phi_0 + \phi_1 \mathrm{a}\left[\psi_{01} + \psi_{11}\mathrm{a}[\theta_{01} + \theta_{11}x] + \psi_{21}\mathrm{a}[\theta_{02} + \theta_{12}x]\right] \\ & \quad +\phi_2 \mathrm{a}[\psi_{02} + \psi_{12}\mathrm{a}[\theta_{01} + \theta_{11}x] + \psi_{22}\mathrm{a}[\theta_{02} + \theta_{12}x]], \end{aligned}$$

where the functions $a[\bullet]$ are ReLU functions. Compute the derivatives of the output y with respect to each of the 13 parameters $\boldsymbol{\phi}_{\bullet}, \theta_{\bullet\bullet}$, and $\psi_{\bullet\bullet}$ directly (i.e., not using the backpropagation algorithm). The derivative of the ReLU function with respect to its input $\partial a[z]/\partial z$ is the indicator function $\mathbb{I}[z > 0]$, which returns one if the argument is greater than zero and zero otherwise (figure 7.6).

Problem 7.2 Find an expression for the final term in each of the five chains of derivatives in equation 7.12.

Problem 7.3 What size are each of the terms in equation 7.19?

Problem 7.4 Calculate the derivative $\partial \ell_i / \partial f[\mathbf{x}_i, \boldsymbol{\phi}]$ for the least squares loss function:

$$\ell_i = (y_i - \mathrm{f}[\mathbf{x}_i, \boldsymbol{\phi}])^2.$$

Problem 7.5 Calculate the derivative $\partial \ell_i / \partial \mathrm{f}[\mathbf{x}_i, \boldsymbol{\phi}]$ for the binary classification loss function:

$$\ell_i = -(1 - y_i)\log\left[1 - \mathrm{sig}[\mathrm{f}[\mathbf{x}_i, \boldsymbol{\phi}]]\right] - y_i \log\left[\mathrm{sig}[\mathrm{f}[\mathbf{x}_i, \boldsymbol{\phi}]]\right],$$

where the function sig[•] is the logistic sigmoid and is defined as:

$$\mathrm{sig}[z] = \frac{1}{1 + \exp[-z]}.$$

Problem 7.6 Show that for $\mathbf{z} = \boldsymbol{\beta} + \boldsymbol{\Omega}\mathbf{h}$:

$$\frac{\partial \mathbf{z}}{\partial \mathbf{h}} = \boldsymbol{\Omega}^T,$$

where $\partial\mathbf{z}/\partial\mathbf{h}$ is a matrix containing the term $\partial z_i/\partial h_j$ in its i^{th} column and j^{th} row. To do this, first find an expression for the constituent elements $\partial z_i/\partial h_j$, and then consider the form that the matrix $\partial\mathbf{z}/\partial\mathbf{h}$ must take.

Answer

We have:

$$z_i = \beta_i + \sum_j \omega_{ij} h_j$$

and so when we take the derivative, we get:

$$\frac{\partial z_i}{\partial h_j} = \omega_{ij}$$

This will be at the i^{th} column and j^{th} row (i.e., at position j,i) of the matrix $\partial\mathbf{z}/\partial\mathbf{h}$, which is hence $\boldsymbol{\Omega}^T$.

Problem 7.7 Consider the case where we use the logistic sigmoid (see equation 7.1) as an activation function, so $h = \text{sig}[f]$. Compute the derivative $\partial h/\partial f$ for this activation function. What happens to the derivative when the input takes (i) a large positive value and (ii) a large negative value?

Problem 7.8 Consider using (i) the Heaviside function and (ii) the rectangular function as activation functions:

$$\text{Heaviside}[z] = \begin{cases} 0 & z < 0 \\ 1 & z \geq 0 \end{cases},$$

and

$$\text{rect}[z] = \begin{cases} 0 & z < 0 \\ 1 & 0 \leq z \leq 1 \\ 0 & z > 1 \end{cases}.$$

Discuss why these functions are problematic for neural network training with gradient-based optimization methods.

Problem 7.9 Consider a loss function $\ell[\mathbf{f}]$, where $\mathbf{f} = \boldsymbol{\beta} + \boldsymbol{\Omega}\mathbf{h}$. We want to find how the loss ℓ changes when we change $\boldsymbol{\Omega}$, which we'll express with a matrix that contains the derivative $\partial\ell/\partial\Omega_{ij}$ at the i^{th} row and j^{th} column. Find an expression for $\partial f_i/\partial\Omega_{ij}$ and using the chain rule, show that:

$$\frac{\partial\ell}{\partial\boldsymbol{\Omega}} = \frac{\partial\ell}{\partial\mathbf{f}}\mathbf{h}^T.$$

Answer

We have

$$f_i = \beta_i + \sum_j \boldsymbol{\Omega}_{ij}h_j,$$

and so:

$$\frac{\partial f_i}{\partial\boldsymbol{\Omega}_{ij}} = h_j.$$

Using the chain rule, we have

$$\frac{\partial\ell}{\partial\boldsymbol{\Omega}_{ij}} = \frac{\partial\ell}{\partial f_i}\frac{\partial f_i}{\partial\boldsymbol{\Omega}_{ij}} = \frac{\partial\ell}{\partial f_i}h_j$$

Converting back to vector form, we have

$$\frac{\partial\ell}{\partial\boldsymbol{\Omega}} = \frac{\partial\ell}{\partial\mathbf{f}}\mathbf{h}^T,$$

as required.

Problem 7.10 Derive the equations for the backward pass of the backpropagation algorithm for a network that uses leaky ReLU activations, which are defined as:

$$\mathrm{a}[z] = \mathrm{ReLU}[z] = \begin{cases} \alpha\cdot z & z < 0 \\ z & z \geq 0 \end{cases},$$

where α is a small positive constant (typically 0.1).

Answer

The Leaky RELU has a gradient of +1 when the input is greater than zero and α when it is less. The backpropagation equations (equation 7.23) will hence be the same, except for the update:

$$\frac{\partial\ell_i}{\partial\mathbf{f}_{k-1}} = \mathbb{I}[\mathbf{f}_{k-1} > 0]\odot\left(\boldsymbol{\Omega}_k^T\frac{\partial\ell_i}{\partial\mathbf{f}_k}\right) + \mathbb{I}[\mathbf{f}_{k-1} < 0]\odot\alpha\left(\boldsymbol{\Omega}_k^T\frac{\partial\ell_i}{\partial\mathbf{f}_k}\right)$$

Problem 7.11 Consider training a network with fifty layers using gradient checkpointing. Assume that we store the pre-activations at every tenth hidden layer during the forward pass. Explain how to compute the derivatives in this situation.

Problem 7.12 This problem explores computing derivatives on general acyclic computational graphs. Consider the function:

$$y = \exp\left[\exp[x] + \exp[x]^2\right] + \sin[\exp[x] + \exp[x]^2].$$

We can break this down into a series of intermediate computations so that:

$$\begin{aligned} f_1 &= \exp[x] \\ f_2 &= f_1^2 \\ f_3 &= f_1 + f_2 \\ f_4 &= \exp[f_3] \\ f_5 &= \sin[f_3] \\ y &= f_4 + f_5. \end{aligned}$$

The associated computational graph is depicted in figure 7.9 in the book. Compute the derivative $\partial y/\partial x$ by *reverse-mode differentiation*. In other words, compute in order:

$$\frac{\partial y}{\partial f_5}, \frac{\partial y}{\partial f_4}, \frac{\partial y}{\partial f_3}, \frac{\partial y}{\partial f_2}, \frac{\partial y}{\partial f_1} \text{ and } \frac{\partial y}{\partial x},$$

using the chain rule in each case to make use of the derivatives already computed.

Answer

The derivatives are:

$$\begin{aligned} \frac{\partial y}{\partial f_5} &= 1 \\ \frac{\partial y}{\partial f_4} &= 1 \\ \frac{\partial y}{\partial f_3} &= \frac{\partial y}{\partial f_4}\frac{\partial f_4}{\partial f_3} + \frac{\partial y}{\partial f_5}\frac{\partial f_5}{\partial f_3} = \exp[f_3] + \cos[f_3] \\ \frac{\partial y}{\partial f_2} &= \frac{\partial y}{\partial f_3}\frac{\partial f_3}{\partial f_2} = \exp[f_3] + \cos[f_3] \\ \frac{\partial y}{\partial f_1} &= \frac{\partial y}{\partial f_3}\frac{\partial f_3}{\partial f_1} + \frac{\partial y}{\partial f_2}\frac{\partial f_2}{\partial f_1} = \exp[f_3] + \cos[f_3] + 2f_1(\exp[f_3] + \cos[f_3]) \\ \frac{\partial y}{\partial x} &= \frac{\partial y}{\partial f_1}\frac{\partial f_1}{\partial x} = \exp[x]\left(\exp[f_3] + \cos[f_3] + 2f_1(\exp[f_3] + \cos[f_3])\right). \end{aligned}$$

Problem 7.13 For the same function in problem 7.12, compute the derivative $\partial y/\partial x$ by *forward-mode differentiation*. In other words, compute in order:

$$\frac{\partial f_1}{\partial x}, \frac{\partial f_2}{\partial x}, \frac{\partial f_3}{\partial x}, \frac{\partial f_4}{\partial x}, \frac{\partial f_5}{\partial x}, \text{ and } \frac{\partial y}{\partial x},$$

using the chain rule in each case to make use of the derivatives already computed. Why do we not use forward-mode differentiation when we calculate the parameter gradients for deep networks?

Answer

The derivatives are:

$$\begin{aligned}
\frac{\partial f_1}{\partial x} &= \exp[x]\\
\frac{\partial f_2}{\partial x} &= \frac{\partial f_2}{\partial f_1}\frac{\partial f_1}{\partial x} = 2f_1\exp[x]\\
\frac{\partial f_3}{\partial x} &= \frac{\partial f_3}{\partial f_1}\frac{\partial f_1}{\partial x} + \frac{\partial f_3}{\partial f_2}\frac{\partial f_2}{\partial x} = \exp[x](1+2f_1)\\
\frac{\partial f_4}{\partial x} &= \frac{\partial f_4}{\partial f_3}\frac{\partial f_3}{\partial x} = \exp[x](1+2f_1)\exp[f_3]\\
\frac{\partial f_5}{\partial x} &= \frac{\partial f_5}{\partial f_3}\frac{\partial f_3}{\partial x} = \exp[x](1+2f_1)\cos[f_3]\\
\frac{\partial y}{\partial x} &= \frac{\partial y}{\partial f_4}\frac{\partial f_4}{\partial x} + \frac{\partial y}{\partial f_5}\frac{\partial f_5}{\partial x} = \exp[x]\left(\exp[f_3]+\cos[f_3]+2f_1(\exp[f_3]+\cos[f_3])\right).
\end{aligned}$$

We don't use forward mode differentiation because we need to compute intermediate terms $\partial y/\partial\bullet$ to update the gradients and these naturally fall out of reverse mode differentiation.

Problem 7.14 Consider a random variable a with variance $\text{Var}[a] = \sigma^2$ and a symmetrical distribution around the mean $\mathbb{E}[a] = 0$. Prove that if we pass this variable through the ReLU function:

$$b = \text{ReLU}[a] = \begin{cases} 0 & a < 0 \\ a & a \geq 0 \end{cases},$$

then the second moment of the transformed variable is $\mathbb{E}[b^2] = \sigma^2/2$.

Problem 7.15 What would you expect to happen if we initialized all of the weights and biases in the network to zero?

Problem 7.16 Implement the code in figure 7.8 in PyTorch and plot the training loss as a function of the number of epochs.

Problem 7.17 Change the code in figure 7.8 to tackle a binary classification problem. You will need to (i) change the targets y so they are binary, (ii) change the network to predict numbers between zero and one (iii) change the loss function appropriately.

Chapter 8

Measuring performance

Problem 8.1 Will the multi-class cross-entropy training loss in figure 8.2 ever reach zero? Explain your reasoning.

Problem 8.2 What values should we choose for the three weights and biases in the first layer of the model in figure 8.4a so that the responses at the hidden units are as depicted in figures 8.4b–d?

Problem 8.3 Given a training dataset consisting of I input/output pairs $\{x_i, y_i\}$, show how the parameters $\{\beta, \omega_1, \omega_2, \omega_3\}$ for the model in figure 8.4a using the least squares loss function can be found in closed form.

Answer

The key is that the first part of the network is now deterministic; we can compute the activations at the three hidden units for any input. Denoting these by h_1, h_2 and h_3, we now have a linear regression problem:

$$y_i = \beta + \omega_1 h_{1i} + \omega_2 h_{2i} + \omega_3 h_{3i},$$

where i indexes the training data.

Problem 8.4 Consider the curve in figure 8.10b at the point where we train a model with a hidden layer of size 200, which would have 50,410 parameters. What do you predict will happen to the training and test performance if we increase the number of training examples from 10,000 to 50,410?

Problem 8.5 Consider the case where the model capacity exceeds the number of training data points, and the model is flexible enough to reduce the training loss to zero. What are the implications of this for fitting a heteroscedastic model? Propose a method to resolve any problems that you identify.

Problem 8.6 Show that the angle between two random samples from a 1000-dimensional standard Gaussian distribution is are orthogonal with high probability.

Problem 8.7 The volume of a hypersphere with radius r in D dimensions is:

$$\text{Vol}[r] = \frac{r^D \pi^{D/2}}{\Gamma[D/2+1]},$$

where $\Gamma[\bullet]$ is the Gamma function. Show using Stirling's formula that the volume of a hypersphere of diameter one (radius $r=0.5$) becomes zero as the dimension increases.

Problem 8.8 Consider a hypersphere of radius $r = 1$. Show the proportion of the total volume in the 1% of the radius closest to the surface of the hypersphere becomes one as the dimension increases.

Answer

Volume of hypersphere of radius $r = 1$ is given by:

$$V_1 = \frac{\pi^{D/2}}{\Gamma[D/2+1]}.$$

Volume of hypersphere of radius $r = 0.99$ is given by:

$$V_{0.99} = \frac{0.99^D \pi^{D/2}}{\Gamma[D/2+1]}.$$

The proportion p in the last one percent is hence:

$$p = \frac{V_1 - V_{0.99}}{V_1} = 1 - 0.99^D.$$

which tends to one as $D \to \infty$.

Problem 8.9 Figure 8.13c in the book shows the distribution of distances of samples of a standard normal distribution as the dimension increases. Empirically verify this finding by sampling from the standard normal distributions in 25, 100, and 500 dimensions and plotting a histogram of the distances from the center. What closed-form probability distribution describes these distances?

Chapter 9

Regularization

Problem 9.1 Consider a model where the prior distribution over the parameters is a normal distribution with mean zero and variance $\sigma^2_{\boldsymbol\phi}$ so that

$$Pr(\boldsymbol\phi) = \prod_{j=1}^{J} \text{Norm}_{\phi_j}[0, \sigma^2_{\boldsymbol\phi}].$$

where j indexes the model parameters. When we apply a prior, we maximize $\prod_{i=1}^{I} Pr(\mathbf{y}_i|\mathbf{x}_i, \boldsymbol\phi)Pr(\boldsymbol\phi)$. Show that the associated loss function of this model is equivalent to L2 regularization.

Problem 9.2 How do the gradients of the loss function change when L2 regularization (equation 9.5) is added?

Problem 9.3 Consider a linear regression model $y = \phi_0 + \phi_1 x$ where x is the input, y is the output, and ϕ_0 and ϕ_1 are the intercept and slope parameters, respectively. Assume we have I training examples $\{x_i, y_i\}$ and use a least squares loss. Consider adding Gaussian noise with mean zero and variance σ^2_x to the inputs x_i at each training iteration. Derive an expression for the expected loss.

Answer

The effective loss function is now:

$$\begin{aligned}
\tilde{L} &= \sum_{i=1}^{I}(\phi_0 + \phi_1(x_i + \epsilon_i) - y_i)^2\\
&= \sum_{i=1}^{I}(\phi_0 + \phi_1 x_i - y_i + \phi_1\epsilon_i)^2\\
&= \sum_{i=1}^{I}(\phi_0 + \phi_1 x_i - y_i)^2 + \phi_1^2\sum_{i=1}^{I}\epsilon_i^2 + 2\phi_1\sum_{i=1}^{I}(\phi_0 + \phi_1 x_i - y_i)\epsilon_i
\end{aligned}$$

where ϵ_i is the noise value that was added to the i^{th} examples. Taking expectations, and noting that $\mathbb{E}[\epsilon_i]=0$ and $\mathbb{E}[\epsilon_i^2]=\sigma_x^2$, we get:

$$\begin{aligned}\mathbb{E}[\tilde{L}] &= \sum_{i=1}^{I}(\phi_0+\phi_1 x_i - y_i)^2 + \phi_1^2\sum_{i=1}^{I}\sigma_x^2\\ &= L+\left(I\sigma_x^2\right)\phi_1^2,\end{aligned}$$

which is equivalent to applying L2 regularization with constant $\lambda = I\sigma_x^2$.

Problem 9.4 Derive the loss function for multi-class classification when we use label smoothing so that the target probability distribution has 0.9 at the correct class and the remaining probability mass of 0.1 is divided between the remaining D_o-1 classes.

Answer

The probability of the data is now:

$$Pr(y_i|\mathbf{x}_i,\boldsymbol{\phi}) = 0.9\cdot \text{softmax}_{y_i}[\text{f}[\mathbf{x}_i,\boldsymbol{\phi}]] + \sum_{z\in\{1\ldots D_o\}\setminus y_i}\frac{0.1}{D_o-1}\cdot\text{softmax}_z[\text{f}[\mathbf{x}_i,\boldsymbol{\phi}]],$$

and the loss function is the negative log probability of this quantity.

Problem 9.5 Show that the weight decay parameter update with decay rate λ:

$$\boldsymbol{\phi}\longleftarrow(1-\lambda)\boldsymbol{\phi}-\alpha\frac{\partial L}{\partial\boldsymbol{\phi}},$$

on the original loss function $\text{L}[\boldsymbol{\phi}]$ is equivalent to a standard gradient update using L2 regularization so that the modified loss function $\tilde{\text{L}}[\boldsymbol{\phi}]$ is:

$$\tilde{\text{L}}[\boldsymbol{\phi}] = \text{L}[\boldsymbol{\phi}] + \frac{\lambda}{2\alpha}\sum_k \phi_k^2,$$

where $\boldsymbol{\phi}$ are the parameters, and α is the learning rate.

Problem 9.6 Consider a model with two parameters $\boldsymbol{\phi}=[\boldsymbol{\phi}_0,\boldsymbol{\phi}_1]^T$. Draw the L0, L$\frac{1}{2}$, L1, and L2 regularization terms in a similar form to that shown in figure 9.1b.

Chapter 10

Convolutional networks

Problem 10.1 Show that the operation in equation 10.3 is equivariant with respect to translation.

Answer

$$\mathrm{f}_i[\mathrm{t}[\mathbf{x}]] = \omega_1 x_{i-1+\tau} + \omega_2 x_{i+\tau} + \omega_3 x_{i+1+\tau} = f_{i+\tau}[\mathbf{x}] = \mathrm{t}[\mathrm{f}_i[\mathbf{x}]]$$

Problem 10.2 Equation 10.3 defines 1D convolution with a kernel size of three, stride of one, and dilation one. Write out the equivalent equation for the 1D convolution with a kernel size of three and a stride of two as pictured in figure 10.3a–b.

Problem 10.3 Write out the equation for the 1D dilated convolution with a kernel size of three and a dilation rate of two, as pictured in figure 10.3d.

Problem 10.4 Write out the equation for a 1D convolution with kernel size seven, dilation rate of three, and stride of three.

Problem 10.5 Draw weight matrices in the style of figure 10.4d for (i) the strided convolution in figure 10.3a–b, (ii) the convolution with kernel size 5 in figure 10.3c, and (iii) the dilated convolution in figure 10.3d.

Problem 10.6 Draw a 12×6 weight matrix in the style of figure 10.4d relating the inputs $x_1, \ldots, x_6$ to the outputs $h_1, \ldots, h_{12}$ in the multi-channel convolution as depicted in figures 10.5a–b.

Answer

The answer is depicted in figure 10.1a.

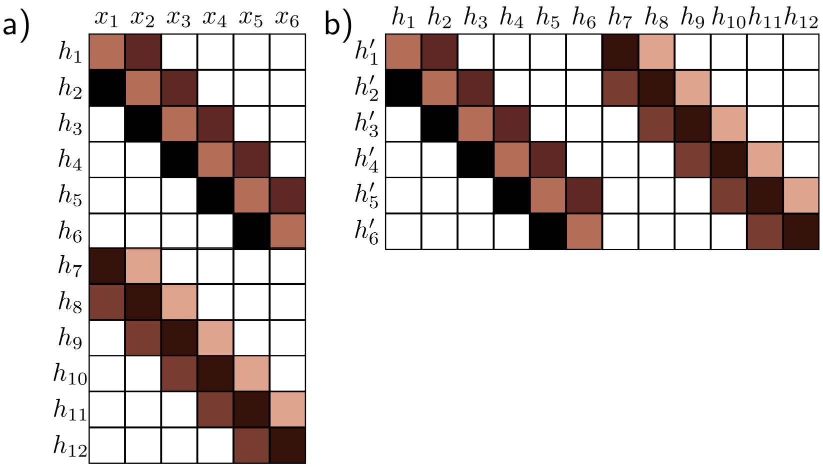

a)
x_1 x_2 x_3 x_4 x_5 x_6
h_1 h_2 h_3 h_4 h_5 h_6 h_7 h_8 h_9 h_{10} h_{11} h_{12}
b)
h_1 h_2 h_3 h_4 h_5 h_6 h_7 h_8 h_9 h_{10} h_{11} h_{12}
h'_1 h'_2 h'_3 h'_4 h'_5 h'_6

Figure 10.1 Answer to a) problem 10.6 and b) problem 10.7.

Problem 10.7 Draw a 6×12 weight matrix in the style of figure 10.4d relating the inputs $h_1, \ldots, h_{12}$ to the outputs $h'_1, \ldots, h'_6$ in the multi-channel convolution in figure 10.5c.

Answer

The answer is depicted in figure 10.1b.

Problem 10.8 Consider a 1D convolutional network where the input has three channels. The first hidden layer is computed using a kernel size of three and has four channels. The second hidden layer is computed using a kernel size of five and has ten channels. How many biases and how many weights are needed for each of these two convolutional layers?

Problem 10.9 A network consists of three 1D convolutional layers. At each layer, a zero-padded convolution with kernel size three, stride one, and dilation one is applied. What size is the receptive field of the hidden units in the third layer?

Problem 10.10 A network consists of three 1D convolutional layers. At each layer, a zero-padded convolution with kernel size seven, stride one, and dilation one is applied. What size is the receptive field of hidden units in the third layer?

Problem 10.11 Consider a convolutional network with 1D input $\mathbf{x}$. The first hidden layer $\mathbf{H}_1$ is computed using a convolution with kernel size five, stride two, and a dilation rate of one. The second hidden layer $\mathbf{H}_2$ is computed using a convolution with kernel size three, stride one, and a dilation rate of one. The third hidden layer $\mathbf{H}_3$ is computed using a convolution with kernel size five, stride one, and a dilation rate of two. What are the receptive field sizes at each hidden layer?

Problem 10.12 The 1D convolutional network in figure 10.7 was trained using stochastic gradient descent with a learning rate of 0.01 and a batch size of 100 on a training dataset of 4,000 examples for 100,000 steps. How many epochs was the network trained for?

Problem 10.13 Draw a weight matrix in the style of figure 10.4d that shows the relationship between the 24 inputs and the 24 outputs in figure 10.9 from the book.

Problem 10.14 Consider a 2D convolutional layer with kernel size 5×5 that takes 3 input channels and returns 10 output channels. How many convolutional weights are there? How many biases?

Problem 10.15 Draw a weight matrix in the style of figure 10.4d that samples every other variable in a 1D input (i.e., the 1D analog of figure 10.11a). Show that the weight matrix for 1D convolution with kernel size three and stride two is equivalent to composing the matrices for 1D convolution with kernel size three and stride one and this sampling matrix.

Problem 10.16 Consider the AlexNet network (figure 10.16 in book). How many parameters are used in each convolutional and fully connected layer? What is the total number of parameters?

Answer

- Between the image and first layer, there are $3 \times 96 \times 11 \times 11 = 34,848$ weights and 96 biases.
- Between the first layer and second layer, there are $96 \times 256 \times 5 \times 5 = 614,400$ weights 256 biases
- Between the second layer and third layer, there are $256 \times 384 \times 3 \times 3 = 884,736$ weights and 384 biases.
- Between the third layer and fourth layer, there are $384 \times 384 \times 3 \times 3 = 1,327,104$ weights and 384 biases
- Between the fourth layer and fifth layer, there are $384 \times 256 \times 3 \times 3 = 884,736$ weights and 256 biases
- At the end of the last convolutional layer, the representation is halved in size by the maxpool operation. Rounding, down, it now has size $6 \times 6 \times 256 = 9,216$. So there are $9,216 \times 4096 = 37,748,736$ weights and 4096 biases
- Between the next two fully connected layers, there are $4096 \times 4096 = 16,777,216$ weights and 4096 biases.
- Between the last two fully connected layers, there are $4096 \times 1000 = 4,096,000$ weights and 1000 biases.

Summing all these together, we get the total number of parameters is $34,848 + 96 + 614,400 + 256 + 884,736 + 384 + 1,327,104 + 384 + 884,736 + 256 + 37,748,736 + 4096 + 16,777,216 + 4096 + 4,096,000 + 1000 = 62,378,344$ parameters.

This is not quite what they claim in the paper. This might possibly be because of the way they trained across multiple GPUs. If anyone sees a mistake in my calculations or can explain this, then please get in touch.

Update: It turns out this is because Krizhevsky et al. (2012) actually had more than one version of their network and these differ a bit in their parameters. (Single GPU version/Dual GPU version). These computations are for the dual GPU model.

Problem 10.17 What is the receptive field size at the first three layers of AlexNet (i.e., the first three orange blocks in figure 10.16 in the book)?

Problem 10.18 How many weights and biases are there at each convolutional layer and fully connected layer in the VGG architecture (figure 10.17 in the book)?

Problem 10.19 Consider two hidden layers of size 224×224 with C_1 and C_2 channels, respectively, connected by a 3×3 convolutional layer. Describe how to initialize the weights using He initialization.

Answer

He initialization is based on the number of weights that contribute to each hidden unit. In this case, there would be $C_1 \times 3 \times 3$ weights contributing to each hidden unit, so we would initialize with mean zero and a variance of $2/9C_1$.

Chapter 11

Residual networks

Problem 11.1 Derive equation 11.5 from the network definition in equation 11.4.

Problem 11.2 Unraveling the four block network in figure 11.4a produces one path of length zero, four paths of length one, six paths of length two, four paths of length three, and one path of length four. How many paths of each length would there be if with (i) three residual blocks and (ii) five residual blocks? Deduce the rule for K residual blocks.

Problem 11.3 Show that the derivative of the network in equation 11.5 with respect to the first layer $\mathbf{f}_1[\mathbf{x}]$ is given by equation 11.6.

Problem 11.4 Explain why the values in the two branches of the residual blocks in figure 11.6a in the book are uncorrelated. Show that the variance of the sum of uncorrelated variables is the sum of their individual variances.

Answer

They are uncorrelated because the input to the main branch of the residual block is multiplied by the weight matrix, which is normally distributed with mean zero. This can equally multiply by a positive or negative number so they are uncorrelated.

The mean of a sum $z = x + y$ of variables x and y is given by:

$$\mathbb{E}[z] = \mathbb{E}[x] + \mathbb{E}[y] = \mu_x + \mu_y$$

The variance of a sum $z = x + y$ of variables x and y is given by:

$$\begin{aligned}\text{Var}[z] &= \mathbb{E}[(z - \mathbb{E}[z])^2] \\ &= \mathbb{E}[(x + y - \mu_x - \mu_y)^2] \\ &= \mathbb{E}[((x - \mu_x) + (y - \mu_y))^2] \\ &= \mathbb{E}[(x - \mu_x)^2] + \mathbb{E}[(y - \mu_y)^2] + 2\mathbb{E}[(x - \mu_x)(y - \mu_y)] \\ &= \text{Var}[x] + \text{Var}[y] + \text{Covar}[x, y]\end{aligned}$$

The covariance is zero if the variables are uncorrelated and so variance is the sum of the individual variances.

Problem 11.5 The forward pass for batch normalization given a batch of scalar values $\{z_i\}_{i=1}^{I}$ consists of the following operations (figure 11.15 from book):

$$\begin{array}{rlrl} f_1 &= \mathbb{E}[z_i] & f_5 &= \sqrt{f_4+\epsilon} \\ f_{2i} &= x_i - f_1 & f_6 &= 1/f_5 \\ f_{3i} &= f_{2i}^2 & f_{7i} &= f_{2i}\times f_6 \\ f_4 &= \mathbb{E}[f_{3i}] & z_i' &= f_{7i}\times\gamma+\delta, \end{array}$$

where $\mathbb{E}[z_i] = \frac{1}{I}\sum_i z_i$. Write Python code to implement the forward pass. Now derive the algorithm for the backward pass. Work backward through the computational graph computing the derivatives to generate a set of operations that computes $\partial z_i'/\partial z_i$ for every element in the batch. Write Python code to implement the backward pass.

Answer

Answer can be found here.

Problem 11.6 Consider a fully connected neural network with one input, one output, and ten hidden layers, each of which contains twenty hidden units. How many parameters does this network have? How many parameters will it have if we place a batch normalization operation between each linear transformation and ReLU?

Problem 11.7 Consider applying an L2 regularization penalty to the weights in the convolutional layers in figure 11.7a, but not to the scaling parameters of the subsequent BatchNorm layers. What do you expect will happen as training proceeds?

Answer

The penalty will prefer small values for the weights, and this will be compensated for by the BatchNorm scaling. Eventually, the weights will become very small, and this may cause numerical problems.

Problem 11.8 Consider a convolutional residual block that contains a batch normalization operation, followed by a ReLU activation function, and then a 3×3 convolutional layer. If the input and output both have 512 channels, how many parameters are needed to define this block? Now consider a bottleneck residual block that contains three batch normalization/ReLU/convolution sequences. The first uses a 1×1 convolution to reduce the number of channels from 512 to 128. The second uses a 3×3 convolution with the same number of input and output channels.

The third uses a 1×1 convolution to increase the number of channels from 128 to 512 (see figure 11.17b). How many parameters are needed to define this block?

Problem 11.9 The U-Net is completely convolutional and can be run with any sized image after training. Why do we not train with a collection of arbitrary sized images?

Chapter 12

Transformers

Problem 12.1 Consider a self-attention mechanism that processes N inputs of length D to produce N outputs of the same size. How many weights and biases are used to compute the queries, keys, and values? Assume that all three quantities are also of length D. How many attention weights a[•, •] will there be? How many weights and biases would there be in a fully connected shallow network relating all DN inputs to all DN outputs?

Problem 12.2 Why might we want to ensure that the input to the self-attention mechanism is the same size as the output?

Problem 12.3 Show that the self-attention mechanism (equation 12.8) is equivariant to a permutation $\mathbf{XP}$ of the data $\mathbf{X}$, where $\mathbf{P}$ is a permutation matrix. In other words, show that:

$$\mathbf{Sa}[\mathbf{XP}] = \mathbf{Sa}[\mathbf{X}]\mathbf{P}.$$

Answer

$$\begin{aligned}
\mathbf{Sa}[\mathbf{XP}] &= (\boldsymbol{\beta}_v\mathbf{1}^{\mathbf{T}} + \boldsymbol{\Omega}_v\mathbf{XP})\mathbf{Softmax}[(\boldsymbol{\beta}_q\mathbf{1}^{\mathbf{T}} + \boldsymbol{\Omega}_q\mathbf{XP})^{\mathbf{T}}(\boldsymbol{\beta}_k\mathbf{1}^{\mathbf{T}} + \boldsymbol{\Omega}_k\mathbf{XP})] \\
&= (\boldsymbol{\beta}_v\mathbf{1}^{\mathbf{T}}\mathbf{P} + \boldsymbol{\Omega}_v\mathbf{XP})\mathbf{Softmax}[(\boldsymbol{\beta}_q\mathbf{1}^{\mathbf{T}}\mathbf{P} + \boldsymbol{\Omega}_q\mathbf{XP})^{\mathbf{T}}(\boldsymbol{\beta}_k\mathbf{1}^{\mathbf{T}}\mathbf{P} + \boldsymbol{\Omega}_k\mathbf{XP})] \\
&= (\boldsymbol{\beta}_v\mathbf{1}^{\mathbf{T}} + \boldsymbol{\Omega}_v\mathbf{X})\mathbf{P}\mathbf{Softmax}[\mathbf{P}^{\mathbf{T}}(\boldsymbol{\beta}_q\mathbf{1}^{\mathbf{T}} + \boldsymbol{\Omega}_q\mathbf{X})^{\mathbf{T}}(\boldsymbol{\beta}_k\mathbf{1}^{\mathbf{T}} + \boldsymbol{\Omega}_k\mathbf{X})\mathbf{P}] \\
&= (\boldsymbol{\beta}_v\mathbf{1}^{\mathbf{T}} + \boldsymbol{\Omega}_v\mathbf{X})\mathbf{P}\mathbf{P}^{\mathbf{T}}\mathbf{Softmax}[(\boldsymbol{\beta}_q\mathbf{1}^{\mathbf{T}} + \boldsymbol{\Omega}_q\mathbf{X})^{\mathbf{T}}(\boldsymbol{\beta}_k\mathbf{1}^{\mathbf{T}} + \boldsymbol{\Omega}_k\mathbf{X})]\mathbf{P} \\
&= (\boldsymbol{\beta}_v\mathbf{1}^{\mathbf{T}} + \boldsymbol{\Omega}_v\mathbf{X})\mathbf{Softmax}[(\boldsymbol{\beta}_q\mathbf{1}^{\mathbf{T}} + \boldsymbol{\Omega}_q\mathbf{X})^{\mathbf{T}}(\boldsymbol{\beta}_k\mathbf{1}^{\mathbf{T}} + \boldsymbol{\Omega}_k\mathbf{X})]\mathbf{P} \\
&= \mathbf{Sa}[\mathbf{X}]\mathbf{P}
\end{aligned}$$

Problem 12.4 Consider the softmax operation:

$$y_i = \text{softmax}_i[\mathbf{z}] = \frac{\exp[z_i]}{\sum_{j=1}^{5}\exp[z_j]},$$

in the case where there are five inputs with values: $z_1 = -3$, $z_2 = 1$, $z_3 = 100$, $z_4 = 5$, $z_5 = -1$. Compute the 25 derivatives, $\partial y_i/\partial z_j$ for all $i, j \in \{1, 2, 3, 4, 5\}$. What do you conclude?

Problem 12.5 Why is implementation more efficient if the values, queries, and keys in each of the H heads each have dimension D/H where D is the original dimension of the data?

Problem 12.6 BERT was pre-trained using two tasks. The first task requires the system to predict missing (masked) words. The second task requires the system to classify pairs of sentences as being adjacent or not in the original text. Identify whether each of these tasks is generative or contrastive (see section 9.3.6 in the book). Why do you think they used two tasks? Propose two novel contrastive tasks that could be used to pre-train a language model.

Problem 12.7 Consider adding a new token to a precomputed masked self-attention mechanism with N tokens. Describe the *extra* computation that must be done to incorporate this new token.

Problem 12.8 Computation in vision transformers expands quadratically with the number of patches. Devise two methods to reduce the computation using the principles from figure 12.15.

Problem 12.9 Consider representing an image with a grid of 16×16 patches, each represented by a patch embedding of length 512. Compare the amount of computation required in the DaViT transformer to perform attention (i) between the patches, using all of the channels, and (ii) between the channels, using all of the patches.

Problem 12.10 Attention weights are usually computed as:

$$a[\mathbf{x}_m, \mathbf{x}_n] = \text{softmax}_m\left[\mathbf{k}_m^T\mathbf{q}_n\right] = \frac{\exp\left[\mathbf{k}_{\bullet}^T\mathbf{q}_n\right]}{\sum_{m'=1}^{N}\exp\left[\mathbf{k}_{m'}^T\mathbf{q}_n\right]}.$$

Consider replacing $\exp\left[\mathbf{k}_m^T\mathbf{q}_n\right]$ with the dot product $\mathbf{g}[\mathbf{k}_m]^T\mathbf{g}[\mathbf{q}_n]$ where $\mathbf{g}[\bullet]$ is a nonlinear transformation. Show how this makes the computation of the attention weights more efficient.

Answer

$$a[\mathbf{x}_m, \mathbf{x}_n] = \frac{\mathrm{g}\left[\mathbf{k}_m\right]^T\mathrm{g}\left[\mathbf{q}_n\right]}{\sum_{m'=1}^{N}\mathrm{g}\left[\mathbf{k}_{m'}\right]^T\mathrm{g}\left[\mathbf{q}_n\right]} == \frac{\mathrm{g}\left[\mathbf{k}_m\right]^T\mathrm{g}\left[\mathbf{q}_n\right]}{\left(\sum_{m'=1}^{N}\mathrm{g}\left[\mathbf{k}_{m'}\right]^T\right)\mathrm{g}\left[\mathbf{q}_n\right]}.$$

Using this formulation, we can compute the summation before the dot product. When we increase N (i.e. we add an extra token during decoding), the original computation of an attention value is $\mathcal{O}[ND]$ (we sum N terms, each of order D). However, in the modified version, we need only to add an extra dot product in the denominator, which costs $\mathcal{O}[D']$ where D' is the dimension of the transformed vector $\mathrm{g}[\bullet]$

See equations 12-14 of this blog for more details.

Chapter 13

Graph neural networks

Problem 13.1 Write out the adjacency matrices for the two graphs in figure 13.2.

Problem 13.2 Draw graphs that correspond to the following adjacency matrices:

$$\mathbf{A}_1 = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 \end{bmatrix} \quad \text{and} \quad \mathbf{A}_2 = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}.$$

Answer

See figure 13.1.

Problem 13.3 Consider the two graphs in figure 13.2. How many ways are there to walk from node one to node two in (i) three steps and (ii) seven steps?

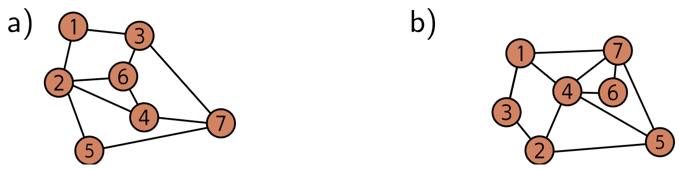


Figure 13.1 Solutions to problem 13.2.

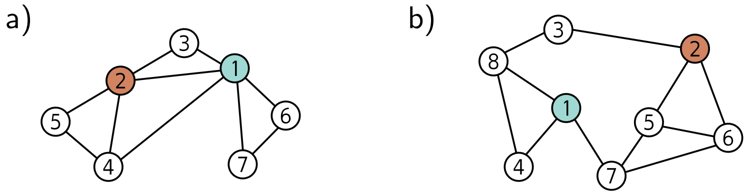


Figure 13.2 Graphs for problems 13.1, 13.3, and 13.8.

Answer

To calculate, multiply adjacency matrix by itself K times. For the first graph (i) 9 and (ii) 879. For the second graph (i) 3 and (ii) 203.

Problem 13.4 The diagonal of $\mathbf{A}^2$ in figure 13.4c in the book contains the number of edges that connect to each corresponding node. Explain this phenomenon.

Problem 13.5 What permutation matrix is responsible for the transformation between the graphs in figures 13.5a–c and figure 13.5d–f?

Problem 13.6 Prove that:

$$\text{sig}\left[\beta_K + \boldsymbol{\omega}_K \mathbf{H}_K \mathbf{1}\right] = \text{sig}\left[\beta_K + \boldsymbol{\omega}_K \mathbf{H}_K \mathbf{P}\mathbf{1}\right],$$

where $\mathbf{P}$ is an $N \times N$ permutation matrix (a matrix that is all zeros except for exactly one entry in each row and each column which is one) and $\mathbf{1}$ is an $N \times 1$ vector of ones.

Problem 13.7 Consider the simple GNN layer:

$$\begin{aligned}\mathbf{H}_{k+1} &= \text{GraphLayer}[\mathbf{H}_k, \mathbf{A}] \\ &= \mathbf{a}\left[\boldsymbol{\beta}_k \mathbf{1}^T + \boldsymbol{\Omega}_k \begin{bmatrix}\mathbf{H}_k \\ \mathbf{H}_k\mathbf{A}\end{bmatrix}\right],\end{aligned}$$

where $\mathbf{H}$ is a $D \times N$ matrix containing the N node embeddings in its columns, $\mathbf{A}$ is the $N \times N$ adjacency matrix, $\boldsymbol{\beta}$ is the bias vector, and $\boldsymbol{\Omega}$ is the weight matrix. Show that this layer is equivariant to permutations of the node order so that:

$$\text{GraphLayer}[\mathbf{H}_k, \mathbf{A}]\mathbf{P} = \text{GraphLayer}[\mathbf{H}_k\mathbf{P}, \mathbf{P}^T\mathbf{A}\mathbf{P}],$$

where $\mathbf{P}$ is an $N \times N$ permutation matrix.

Answer

$$\begin{aligned}
\text{GraphLayer}[\mathbf{H}_k\mathbf{P}, \mathbf{P}^T\mathbf{A}\mathbf{P}] &= \mathbf{a}\left[\boldsymbol{\beta}_k\mathbf{1}^T + \boldsymbol{\Omega}_k\begin{bmatrix}\mathbf{H}_k\mathbf{P}\\ \mathbf{H}_k\mathbf{P}\mathbf{P}^T\mathbf{A}\mathbf{P}\end{bmatrix}\right]\\
&= \mathbf{a}\left[\boldsymbol{\beta}_k\mathbf{1}^T + \boldsymbol{\Omega}_k\begin{bmatrix}\mathbf{H}_k\mathbf{P}\\ \mathbf{H}_k\mathbf{A}\mathbf{P}\end{bmatrix}\right]\\
&= \mathbf{a}\left[\boldsymbol{\beta}_k\mathbf{1}^T\mathbf{P} + \boldsymbol{\Omega}_k\begin{bmatrix}\mathbf{H}_k\mathbf{P}\\ \mathbf{H}_k\mathbf{A}\mathbf{P}\end{bmatrix}\right]\\
&= \mathbf{a}\left[\boldsymbol{\beta}_k\mathbf{1}^T + \boldsymbol{\Omega}_k\begin{bmatrix}\mathbf{H}_k\\ \mathbf{H}_k\mathbf{A}\end{bmatrix}\right]\mathbf{P}\\
&= \text{GraphLayer}[\mathbf{H}_k, \mathbf{A}]\mathbf{P}
\end{aligned}$$

Problem 13.8 What is the degree matrix $\mathbf{D}$ for each graph in figure 13.14 in the book?

Problem 13.9 The authors of GraphSAGE propose a pooling method in which the node embedding is averaged together with its neighbors so that:

$$\mathbf{agg}[n] = \frac{1}{1 + |\text{ne}[n]|}\left(\mathbf{h}_n + \sum_{m\in\text{ne}[n]}\mathbf{h}_m\right).$$

Show how this operation can be computed simultaneously for all node embeddings in the $D \times N$ embedding matrix $\mathbf{H}$ using linear algebra. You will need to use both the adjacency matrix $\mathbf{A}$ and the degree matrix $\mathbf{D}$.

Problem 13.10 Devise a graph attention mechanism based on dot-product self-attention and draw its mechanism in the style of figure 13.12 in the book.

Answer

This is pictured in figure 13.3.

Problem 13.11 Draw the edge graph associated with the graph in figure 13.15a in the book.

Answer

See figure 13.4a.

Problem 13.12 Draw the node graph corresponding to the edge graph in figure 13.15b in the book.

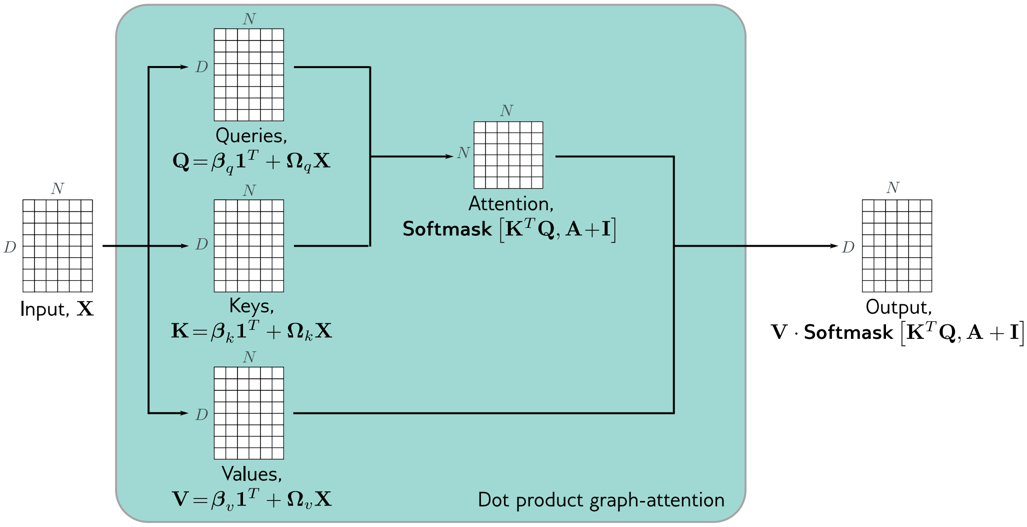

N
D
Queries,
$\mathbf{Q}=\boldsymbol{\beta}_q\mathbf{1}^T+\boldsymbol{\Omega}_q\mathbf{X}$
N
D
Input, $\mathbf{X}$
N
D
Keys,
$\mathbf{K}=\boldsymbol{\beta}_k\mathbf{1}^T+\boldsymbol{\Omega}_k\mathbf{X}$
N
D
Values,
$\mathbf{V}=\boldsymbol{\beta}_v\mathbf{1}^T+\boldsymbol{\Omega}_v\mathbf{X}$
N
N
Attention,
Softmask $\left[\mathbf{K}^T\mathbf{Q},\mathbf{A}+\mathbf{I}\right]$
Dot product graph-attention
N
D
Output,
$\mathbf{V}\cdot$ Softmask $\left[\mathbf{K}^T\mathbf{Q},\mathbf{A}+\mathbf{I}\right]$

Figure 13.3 Answer to problem 13.10.

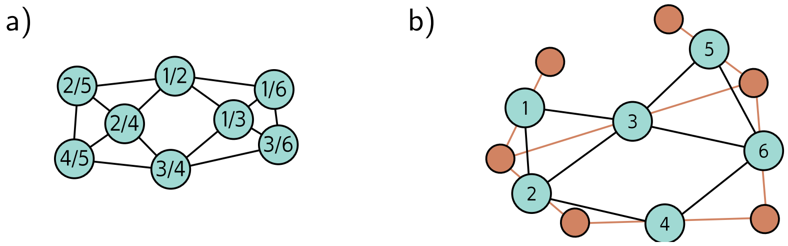


Figure 13.4 a) Answer to problem 13.12.b) Answer to problem 13.13.

Answer

See figure 13.4b.

Problem 13.13 For a general undirected graph, describe how the adjacency matrix of the node graph relates to the adjacency matrix of the corresponding edge graph.

Problem 13.14 Design a layer that updates a node embedding $\mathbf{h}_n$ based on its neighboring node embeddings $\{\mathbf{h}_m\}_{m\in\text{ne}[n]}$ and neighboring edge embeddings $\{\mathbf{e}_m\}_{m\in\text{nee}[n]}$. You should consider the possibility that the edge embeddings are not the same size as the node embeddings.

Answer

$$\mathbf{H}_{k+1}=\mathbf{a}\Big[\boldsymbol{\beta}_k\mathbf{1}^T+\boldsymbol{\Omega}_k\mathbf{H}_k(\mathbf{A}+\mathbf{I})+\boldsymbol{\Phi}_k\mathbf{E}_k\mathbf{N}\Big],$$

where the matrix $\mathbf{\Phi} \in \mathbb{R}^{|\mathcal{N}|\times|E|}$ contains parameters for the edges $\mathbf{E}$ contains the edge embeddings and $\mathbf{N}$ is a matrix that contains the indices of the edges that connect to node k in it's k^{th} column.

Chapter 14

Unsupervised learning

Chapter 15

Generative Adversarial Networks

Problem 15.1 What will the loss be in equation 15.9 when $Pr(\mathbf{x}^*) = Pr(\mathbf{x})$?

Problem 15.2 Write an equation relating the loss L in equation 15.8 to the Jensen-Shannon distance $D_{JS}\left[Pr(\mathbf{x}^*)\,||\;Pr(\mathbf{x})\right]$ in equation 15.9.

Answer

$$D_{JS}\Bigl[Pr(\mathbf{x}^*)\,||\;Pr(\mathbf{x})\Bigr] = -\frac{1}{2}L + \log[2].$$

Problem 15.3 Consider computing the earth mover's distance using linear programming in the primal form. The discrete distributions $Pr(x = i)$ and $q(x = j)$ are defined on $x = 1, 2, 3, 4$ and

$$\mathbf{b} = [Pr(x{=}1), Pr(x{=}2), Pr(x{=}3), Pr(x{=}4), q(x{=}1), q(x{=}2), q(x{=}3), q(x{=}4)]^T.$$

Write out the contents of the 8×16 matrix $\mathbf{A}$. You may assume that the contents of $\mathbf{P}$ has been vectorized into $\mathbf{p}$ column-first.

Problem 15.4 Calculate (i) the KL divergence, (ii) the reverse KL divergence,(iii) the Jensen-Shannon divergence, and (iv) the Wasserstein distance between the distributions:

$$Pr(z) = \begin{cases} 0 & z < 0 \\ 1 & 0 \leq z \leq 1 \\ 0 & z > 1 \end{cases},$$

and

$$Pr(z) = \begin{cases} 0 & z < a \\ 1 & a \leq z \leq a+1 \\ 0 & z > a \end{cases}.$$

for the range $a \in [-3, 3]$. To get a formula for the Wasserstein distance for this special case, consider the total "earth" (i.e., probability mass) that must be moved and multiply this by the distance it must move.

Answer

- The KL divergence is zero when $a = 0$ and infinite otherwise.
- The reverse KL divergence is zero when $a = 0$ and infinite otherwise.
- The Jenson-Shannon divergence is $\log 2$ when $a \leq -1$ or $a \geq 1$ and takes the value $|a| \log[2]$ otherwise
- The Wasserstein distance is $|a|$

Problem 15.5 The KL distance and Wasserstein distances between univariate Gaussian distributions are given by:

$$d_{kl} = \log\left[\frac{\sigma_2}{\sigma_1}\right] + \frac{\sigma_1^2 + (\mu_1 - \mu_2)^2}{2\sigma_2^2} - \frac{1}{2},$$

and

$$d_w = (\mu_1 - \mu_2)^2 + \sigma_1 + \sigma_2 - 2\sqrt{\sigma_1\sigma_2},$$

respectively. Plot these distances as a function of $\mu_1 - \mu_2$ for the case when $\sigma_1 = \sigma_2 = 1$.

Problem 15.6 Consider a latent variable $\mathbf{z}$ with dimension 100. Consider truncating the values of this variable to (i) $\tau = 2.0$, (ii) $\tau = 1.0$, (iii) $\tau = 0.5$, (iv) $\tau = 0.04$ standard deviations. What proportion of the original probability distribution is truncated in each case?

Chapter 16

Normalizing flows

Problem 16.1 Consider transforming a uniform base density defined on $z \in [0,1]$ using the function $x = \mathrm{f}[z] = z^2$. Find an expression for the transformed distribution $Pr(x)$.

Problem 16.2 Consider transforming a standard normal distribution:

$$Pr(z) = \frac{1}{\sqrt{2\pi}} \exp\left[\frac{-z^2}{2}\right],$$

with the function:

$$x = \mathrm{f}[z] = \frac{1}{1+\exp[-z]}.$$

Find an expression for the transformed distribution $Pr(x)$.

Answer

Inverting the transformation gives:

$$z = \log\left[\frac{x}{1-x}\right].$$

The derivative of the transformation is:

$$\frac{\partial z}{\partial x} = \frac{1}{x(1-x)}$$

So, we have:

$$Pr(x) = \frac{1}{\sqrt{2\pi}} \exp\left[\frac{-\log\left[\frac{x}{1-x}\right]^2}{2}\right] \left|\frac{1}{x(1-x)}\right|.$$

Problem 16.3 Write expressions for the Jacobian of the inverse mapping $\mathbf{z} = \mathbf{f}^{-1}[\mathbf{x}, \phi]$ and the absolute determinant of that Jacobian in forms similar to equations 16.6 and 16.7.

Answer

The Jacobian of the inverse mapping can be expressed as:

$$\frac{\partial \mathbf{f}^{-1}[\mathbf{x},\boldsymbol{\phi}]}{\partial \mathbf{x}} = \frac{\partial \mathbf{f}_1^{-1}[\mathbf{f}_2,\boldsymbol{\phi}_1]}{\partial \mathbf{f}_2} \cdot \frac{\partial \mathbf{f}_2^{-1}[\mathbf{f}_3,\boldsymbol{\phi}_2]}{\partial \mathbf{f}_3} \cdot \ldots \frac{\partial \mathbf{f}_{K-1}^{-1}[\mathbf{f}_K,\boldsymbol{\phi}_{K-1}]}{\partial \mathbf{f}_K} \frac{\partial \mathbf{f}_K^{-1}[\mathbf{x},\boldsymbol{\phi}_K]}{\partial \mathbf{x}} \cdot,$$

and the determinant can either be computed as the product of the determinants in this expression, or just the inverse of the original determinant:

$$\left|\frac{\partial \mathbf{f}^{-1}[\mathbf{x},\boldsymbol{\phi}]}{\partial \mathbf{x}}\right| = \left|\frac{\partial \mathbf{f}_K[\mathbf{f}_{K-1},\boldsymbol{\phi}_K]}{\partial \mathbf{f}_{K-1}}\right|^{-1} \cdot \left|\frac{\partial \mathbf{f}_{K-1}[\mathbf{f}_{K-2},\boldsymbol{\phi}_{K-1}]}{\partial \mathbf{f}_{K-2}}\right|^{-1} \ldots \left|\frac{\partial \mathbf{f}_2[\mathbf{f}_1,\boldsymbol{\phi}_2]}{\partial \mathbf{f}_1}\right|^{-1} \cdot \left|\frac{\partial \mathbf{f}_1[\mathbf{z},\boldsymbol{\phi}_1]}{\partial \mathbf{z}}\right|^{-1}.$$

Problem 16.4 Compute the inverse and the determinant of the following matrices by hand:

$$\boldsymbol{\Omega}_1 = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & -5 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 2 \end{bmatrix} \qquad \boldsymbol{\Omega}_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 2 & 4 & 0 & 0 \\ 1 & -1 & 2 & 0 \\ 4 & -2 & -2 & 1 \end{bmatrix}.$$

Problem 16.5 Consider a random variable $\mathbf{z}$ with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$ that is transformed as $\mathbf{x} = \mathbf{A}\mathbf{z} + \mathbf{b}$. Show that the expected value of $\mathbf{x}$ is $\mathbf{A}\boldsymbol{\mu} + \mathbf{b}$ and that the covariance of $\mathbf{x}$ is $\mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^T$.

Problem 16.6 Prove that if $\mathbf{x} = \mathbf{f}[\mathbf{z}] = \mathbf{A}\mathbf{z} + \mathbf{b}$ and $Pr(\mathbf{z}) = \text{Norm}_{\mathbf{z}}[\boldsymbol{\mu}, \boldsymbol{\Sigma}]$, then $Pr(\mathbf{x}) = \text{Norm}_{\mathbf{x}}[\mathbf{A}\boldsymbol{\mu} + \mathbf{b}, \mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^T]$ using the relation:

$$Pr(\mathbf{x}) = \text{Pr}(\mathbf{z}) \cdot \left|\frac{\partial \mathbf{f}[\mathbf{z}]}{\partial \mathbf{z}}\right|^{-1}.$$

Answer

We have $\mathbf{z} = \mathbf{A}^{-1}(\mathbf{x} - \mathbf{b})$.

$$\begin{aligned}
Pr(\mathbf{x}) &= \text{Pr}(\mathbf{z})\cdot\left|\frac{\partial \mathbf{f}[\mathbf{z}]}{\partial \mathbf{z}}\right|^{-1}\\
&= \frac{1}{|\boldsymbol{\Sigma}|^{1/2}\pi^{d/2}}\exp\left[-0.5(\mathbf{z}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{z}-\boldsymbol{\mu})\right]\cdot|\mathbf{A}|^{-1}\\
&= \frac{1}{|\mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^T|^{1/2}\pi^{d/2}}\exp\left[-0.5(\mathbf{z}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{z}-\boldsymbol{\mu})\right]\\
&= \frac{1}{|\mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^T|^{1/2}\pi^{d/2}}\exp\left[-0.5(\mathbf{A}^{-1}(\mathbf{x}-\mathbf{b})-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{A}^{-1}(\mathbf{x}-\mathbf{b})-\boldsymbol{\mu})\right]\\
&= \frac{1}{|\mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^T|^{1/2}\pi^{d/2}}\exp\left[-0.5((\mathbf{x}-\mathbf{A}\boldsymbol{\mu}-\mathbf{b}))^T\mathbf{A}^{-T}\boldsymbol{\Sigma}^{-1}\mathbf{A}^{-1}(\mathbf{x}-\mathbf{A}\boldsymbol{\mu}-\mathbf{b})\right],
\end{aligned}$$

as required.

Problem 16.7 The Leaky ReLU is defined as:

$$\text{LReLU}[z] = \begin{cases} 0.1z & z<0\\ z & z\geq 0\end{cases}.$$

Write an expression for the inverse of the leaky ReLU. Write an expression for the inverse absolute determinant of the Jacobian $|\partial\mathbf{f}[\mathbf{z}]/\partial\mathbf{z}|^{-1}$ for an elementwise transformation $\mathbf{x}=\mathbf{f}[z]$ of the multivariate variable $\mathbf{z}$ where:

$$\mathbf{f}[\mathbf{z}] = \Big[\text{LReLU}[z_1], \text{LReLU}[z_2], \ldots, \text{LReLU}[z_D]\Big]^T.$$

Problem 16.8 Consider applying the piecewise linear function $\text{f}[h,\boldsymbol{\phi}]$ defined in equation 16.12 in the book for the domain $h'\in[0,1]$ elementwise to an input $\mathbf{h}=[h_1,h_2,\ldots,h_D]^T$ so that $\mathbf{f}[\mathbf{h}] = [\text{f}[h_1,\boldsymbol{\phi}],\text{f}[h_2,\boldsymbol{\phi}],\ldots,\text{f}[h_D,\boldsymbol{\phi}]]$. What is the Jacobian $\partial\text{f}[\mathbf{h}]/\partial\mathbf{h}$? What is the determinant of the Jacobian?

Problem 16.9 Consider constructing an element-wise flow based on a conical combination of square root functions in equally spaced bins:

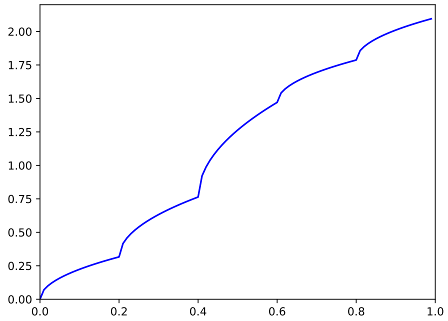
$$\mathbf{h}' = \text{f}[h,\boldsymbol{\phi}] = \sqrt{[Kh-b+1]\,\phi_b}+\sum_{k=1}^{b-1}\sqrt{\phi_k},$$

where $b=\lfloor Kh\rfloor+1$ is the bin that h falls into, and the parameters $\boldsymbol{\phi}_k$ are positive and sum to one. Consider the case where $K=5$ and $\phi_1=0.1,\phi_2=0.2,\phi_3=0.5,\phi_4=0.1,\phi_5=0.1$. Draw the function $\text{f}[h,\boldsymbol{\phi}]$. Draw the inverse function $\text{f}^{-1}[h',\boldsymbol{\phi}]$.

Answer

The function is illustrated in figure 16.1. Inverse is just transposed figure.

Figure 16.1 Answer to question 16.9.



Problem 16.10 Draw the structure of the Jacobian (i.e., indicating which elements are zero) for the forward mapping of the residual flow in figure 16.8 for the cases where $\mathbf{f}_1[\bullet, \boldsymbol{\phi}_1]$ and $\mathbf{f}_2[\bullet, \boldsymbol{\phi}_2]$ are (i) a fully connected neural network, (ii) an elementwise flow.

Problem 16.11 Write out the expression for the KL divergence in equation 16.25. Why does it not matter if we can only evaluate the probability $q(\mathbf{x})$ up to a scaling factor κ? Does the network have to be invertible to minimize this loss function? Explain your reasoning.

Answer

$$L = -\sum_i \log[q(\mathrm{f}[\mathbf{z}_i, \boldsymbol{\phi}])]$$

It doesn't matter, since this will only add another additive log term of the form $\sum_i \log[C]$ where C is the rescaling constant and this does not depend on $\boldsymbol{\phi}$.

No, it doesn't have to be invertible – we generate the samples ourselves and so we know the latent variables.

Chapter 17

Variational autoencoders

Problem 17.1 How many parameters are needed to create a 1D mixture of Gaussians with $n = 5$ components (equation 17.4 in the book)? State the possible range of values that each parameter could take.

Problem 17.2 A function is concave if its second derivative is less than or equal to zero everywhere. Show that this is true for the function $\mathrm{g}[x] = \log[x]$.

Problem 17.3 For convex functions, Jensen's inequality works the other way around.

$$g\big[\mathbb{E}[y]\big] \leq \mathbb{E}\big[g[y]\big].$$

A function is convex if its second derivative is non-negative everywhere. Show that the function $g[x] = x^{2n}$ is convex for arbitrary $n \in [1, 2, 3, \ldots]$. Use this result with Jensen's inequality to show that the square of the mean $\mathbb{E}[x]$ of a distribution $Pr(x)$ must be less than or equal to its second moment $\mathbb{E}[x^2]$.

Problem 17.4 Show that the ELBO as expressed in equation 17.18 can alternatively be derived from the KL divergence between the variational distribution $q(\mathbf{z}|\mathbf{x})$ and the true posterior distribution $Pr(\mathbf{z}|\mathbf{x}, \boldsymbol{\phi})$:

$$\mathrm{D}_{KL}\Big[q(\mathbf{z}|\mathbf{x})\Big|\Big|Pr(\mathbf{z}|\mathbf{x}, \boldsymbol{\phi})\Big] = \int q(\mathbf{z}|\mathbf{x}) \log\left[\frac{q(\mathbf{z}|\mathbf{x})}{Pr(\mathbf{z}|\mathbf{x}, \boldsymbol{\phi})}\right] d\mathbf{z}.$$

Start by using Bayes's rule (equation 17.19).

Answer

$$\begin{aligned} D_{KL}\Big[q(\mathbf{z}|\mathbf{x})||Pr(\mathbf{z}|\mathbf{x}, \boldsymbol{\phi})\Big] &= \int q(\mathbf{z}|\mathbf{x}) \log\left[\frac{Pr(\mathbf{x}|\boldsymbol{\phi})q(\mathbf{z}|\mathbf{x})}{Pr(\mathbf{x}|\mathbf{z}, \boldsymbol{\phi})Pr(\mathbf{z})}\right] d\mathbf{z} \\ &= \int q(\mathbf{z}|\mathbf{x}) \left(\log\left[Pr(\mathbf{x}|\boldsymbol{\phi})\right] + \log\left[\frac{q(\mathbf{z}|\mathbf{x})}{Pr(\mathbf{z})}\right] - \log[Pr(\mathbf{x}|\mathbf{z}, \boldsymbol{\phi}]\right) d\mathbf{z} \\ &= \log\big[Pr(\mathbf{x}|\boldsymbol{\phi})\big] + D_{KL}\Big[q(\mathbf{z}|\mathbf{x})||Pr(\mathbf{z})\Big] - \mathbb{E}_q\left[\log[Pr(\mathbf{x}|\mathbf{z}, \boldsymbol{\phi}]\right]. \end{aligned}$$

The left-hand side is a distance and so it must be non-negative and hence we have:

$$\begin{aligned}\mathbb{E}_q\left[\log[Pr(\mathbf{x}|\mathbf{z},\boldsymbol{\phi})]\right]-D_{KL}\Bigl[q(\mathbf{z}|\mathbf{x})||Pr(\mathbf{z})\Bigr] &= \log[Pr(\mathbf{x}|\boldsymbol{\phi})-D_{KL}\Bigl[q(\mathbf{z}|\mathbf{x})||Pr(\mathbf{z}|\mathbf{x},\boldsymbol{\phi})\Bigr]\\ &\leq \log[Pr(\mathbf{x}|\boldsymbol{\phi}).\end{aligned}$$

Problem 17.5 The reparameterization trick computes the derivative of an expectation of a function f$[x]$:

$$\frac{\partial}{\partial\boldsymbol{\phi}}\mathbb{E}_{Pr(x|\boldsymbol{\phi})}\bigl[\mathrm{f}[x]\bigr],$$

with respect to the parameters $\boldsymbol{\phi}$ of the distribution $Pr(x|\boldsymbol{\phi})$ that the expectation is over. Show that this derivative can also be computed as:

$$\begin{aligned}\frac{\partial}{\partial\boldsymbol{\phi}}\mathbb{E}_{Pr(x|\boldsymbol{\phi})}\bigl[\mathrm{f}[x]\bigr] &= \mathbb{E}_{Pr(x|\boldsymbol{\phi})}\left[\mathrm{f}[x]\frac{\partial}{\partial\boldsymbol{\phi}}\log\bigl[Pr(\mathbf{x}|\boldsymbol{\phi})\bigr]\right]\\ &\approx \frac{1}{I}\sum_{i=1}^{I}\mathrm{f}[x_i]\frac{\partial}{\partial\boldsymbol{\phi}}\log\bigl[Pr(x_i|\boldsymbol{\phi})\bigr].\end{aligned}$$

This method is known as the *REINFORCE algorithm* or *score function estimator*.

Problem 17.6 Why is it better to use spherical linear interpolation rather than regular linear interpolation when moving between points in the latent space? Hint: consider figure 8.13.

Problem 17.7 Derive the EM algorithm for the 1D mixture of Gaussians algorithm with N components. To do this, you need to (i) find an expression for the posterior distribution $Pr(z|x)$ over the latent variable $z \in \{1, 2, \ldots, N\}$ for a data point x and (ii) find an expression that updates the evidence lower bound given the posterior distributions for all of the data points. You will need to use Lagrange multipliers to ensure that the weights $\lambda_1, \ldots, \lambda_N$ of the Gaussians sum to one.

Answer

See section 7.4 of my previous book.

Chapter 18

Diffusion models

Problem 18.1 Show that if $\mathrm{Cov}[\mathbf{x}_{t-1}] = \mathbf{I}$ and we use the update:

$$\mathbf{x}_t = \sqrt{1-\beta_t}\cdot\mathbf{x}_{t-1} + \sqrt{\beta_t}\cdot\boldsymbol{\epsilon}_t,$$

then $\mathrm{Cov}[\mathbf{x}_t] = \mathbf{I}$, so the variance stays the same.

Problem 18.2 Consider the variable:

$$z = a\cdot\epsilon_1 + b\cdot\epsilon_2,$$

where both ϵ_1 and ϵ_2 are drawn from independent standard normal distributions with mean zero and unit variance. Show that:

$$\begin{aligned}\mathbb{E}[z] &= 0\\ \mathrm{Var}[z] &= a^2+b^2,\end{aligned}$$

so we could equivalently compute $z = \sqrt{a^2+b^2}\cdot\epsilon$ where ϵ is also drawn from a standard normal distribution.

Problem 18.3 Continue the process in equation 18.5 to show that:

$$\mathbf{z}_3 = \sqrt{(1-\beta_3)(1-\beta_2)(1-\beta_1)}\cdot\mathbf{x} + \sqrt{1-(1-\beta_3)(1-\beta_2)(1-\beta_1)}\cdot\boldsymbol{\epsilon}',$$

where $\boldsymbol{\epsilon}'$ is a draw from a standard normal distribution.

Problem 18.4 Prove the relation:

$$\mathrm{Norm}_{\mathbf{v}}\left[\mathbf{A}\mathbf{w},\mathbf{B}\right] \propto \mathrm{Norm}_{\mathbf{w}}\Big[(\mathbf{A}^T\mathbf{B}^{-1}\mathbf{A})^{-1}\mathbf{A}^T\mathbf{B}^{-1}\mathbf{v}, (\mathbf{A}^T\mathbf{B}^{-1}\mathbf{A})^{-1}\Big].$$

Answer

$$
\begin{aligned}
\mathrm{Norm}_{\mathbf{v}}\left[\mathbf{Aw},\mathbf{B}\right] &= \kappa_1\cdot\exp\left[-0.5(\mathbf{v}-\mathbf{Aw})^T\mathbf{B}^{-1}(\mathbf{v}-\mathbf{Aw})\right]\\
&= \kappa_1\cdot\exp\left[-0.5\left(\mathbf{v}^T\mathbf{B}^{-1}\mathbf{v}-2\mathbf{v}^T\mathbf{B}^{-1}\mathbf{Aw}+\mathbf{w}^T\mathbf{A}^T\mathbf{B}^{-1}\mathbf{Aw}\right)\right]\\
&= \kappa_2\cdot\exp\left[-0.5\left(-2\mathbf{v}^T\mathbf{B}^{-1}\mathbf{Aw}+\mathbf{w}^T\mathbf{A}^T\mathbf{B}^{-1}\mathbf{Aw}\right)\right]
\end{aligned}
$$

where we have absorbed the first term into the constant κ_2 in the last line. Now we complete the square by adding a new term:

$$
\begin{aligned}
&\mathrm{Norm}_{\mathbf{v}}\left[\mathbf{Aw},\mathbf{B}\right]\\
&=\kappa_2\cdot\exp\left[-0.5\left(-2\mathbf{v}^T\mathbf{B}^{-1}\mathbf{Aw}+\mathbf{w}^T\mathbf{A}^T\mathbf{B}^{-1}\mathbf{Aw}\right)\right]\\
&=\kappa_3\cdot\exp\left[-0.5\left(\mathbf{v}^T\mathbf{B}^{-1}\mathbf{A}(\mathbf{A}^T\mathbf{B}^{-1}\mathbf{A})^{-1}\mathbf{A}^T\mathbf{B}^{-1}\mathbf{v}-2\mathbf{v}^T\mathbf{B}^{-1}\mathbf{Aw}+\mathbf{w}^T\mathbf{A}^T\mathbf{B}^{-1}\mathbf{Aw}\right)\right]\\
&=\kappa_3\cdot\exp\left[-0.5\left(\mathbf{w}-(\mathbf{A}^T\mathbf{B}^{-1}\mathbf{A})^{-1}\mathbf{A}^T\mathbf{B}^{-1}\mathbf{v}\right)^T\left(\mathbf{A}^T\mathbf{B}^{-1}\mathbf{A}\right)\left(\mathbf{w}-(\mathbf{A}^T\mathbf{B}^{-1}\mathbf{A})^{-1}\mathbf{A}^T\mathbf{B}^{-1}\mathbf{v}\right)\right]\\
&=\kappa_4\cdot\mathrm{Norm}_{\mathbf{w}}\left[(\mathbf{A}^T\mathbf{B}^{-1}\mathbf{A})^{-1}\mathbf{A}^T\mathbf{B}^{-1}\mathbf{v},(\mathbf{A}^T\mathbf{B}^{-1}\mathbf{A})^{-1}\right].
\end{aligned}
$$

Problem 18.5 Prove the relation:

$$
\mathrm{Norm}_{\mathbf{x}}[\mathbf{a},\mathbf{A}]\mathrm{Norm}_{\mathbf{x}}[\mathbf{b},\mathbf{B}] \quad\propto\quad \mathrm{Norm}_{\mathbf{x}}\left[(\mathbf{A}^{-1}+\mathbf{B}^{-1})^{-1}(\mathbf{A}^{-1}\mathbf{a}+\mathbf{B}^{-1}\mathbf{b}),(\mathbf{A}^{-1}+\mathbf{B}^{-1})^{-1}\right].
$$

Answer

$$
\begin{aligned}
&\mathrm{Norm}_{\mathbf{x}}[\mathbf{a},\mathbf{A}]\mathrm{Norm}_{\mathbf{x}}[\mathbf{b},\mathbf{B}]\\
&\qquad=\kappa_1\cdot\exp\left[-0.5(\mathbf{x}-\mathbf{a})^T\mathbf{A}^{-1}(\mathbf{x}-\mathbf{a})\right]\exp\left[-0.5(\mathbf{x}-\mathbf{b})^T\mathbf{B}^{-1}(\mathbf{x}-\mathbf{b})\right]\\
&\qquad=\kappa_1\cdot\exp\left[-0.5\left((\mathbf{x}-\mathbf{a})^T\mathbf{A}^{-1}(\mathbf{x}-\mathbf{a})-(\mathbf{x}-\mathbf{b})^T\mathbf{B}^{-1}(\mathbf{x}-\mathbf{b})\right)\right]\\
&\qquad=\kappa_1\cdot\exp\left[-0.5\left(\mathbf{x}^T(\mathbf{A}^{-1}+\mathbf{B}^{-1})\mathbf{x}-2\mathbf{x}(\mathbf{A}^{-1}\mathbf{a}+\mathbf{B}^{-1}\mathbf{b})+\mathbf{a}^T\mathbf{A}^{-1}\mathbf{a}+\mathbf{b}^T\mathbf{B}^{-1}\mathbf{b}\right)\right]\\
&\qquad=\kappa_2\cdot\exp\left[-0.5\left(\mathbf{x}^T(\mathbf{A}^{-1}+\mathbf{B}^{-1})\mathbf{x}-2\mathbf{x}(\mathbf{A}^{-1}\mathbf{a}+\mathbf{B}^{-1}\mathbf{b})\right)\right]
\end{aligned}
$$

where we have absorbed the last two tewrms into the constant κ_2 in the last line. Now we complete the square by adding a new term:

$$\begin{aligned}
\text{Norm}_{\mathbf{x}}[\mathbf{a},\mathbf{A}]\text{Norm}_{\mathbf{x}}[\mathbf{b},\mathbf{B}] \quad =& \\
&= \kappa_2 \cdot \exp\left[-0.5\left(\mathbf{x}^T(\mathbf{A}^{-1}+\mathbf{B}^{-1})\mathbf{x} - 2\mathbf{x}(\mathbf{A}^{-1}\mathbf{a}+\mathbf{B}^{-1}\mathbf{b})\right)\right] \\
&= \kappa_3 \cdot \exp\Bigg[-0.5\Big(\mathbf{x}^T(\mathbf{A}^{-1}+\mathbf{B}^{-1})\mathbf{x} - 2\mathbf{x}(\mathbf{A}^{-1}\mathbf{a}+\mathbf{B}^{-1}\mathbf{b}) \\
&\qquad\qquad + (\mathbf{A}^{-1}\mathbf{a}+\mathbf{B}^{-1}\mathbf{b})^T(\mathbf{A}^{-1}+\mathbf{B}^{-1})^{-1}(\mathbf{A}^{-1}\mathbf{a}+\mathbf{B}^{-1}\mathbf{b})\Big)\Bigg] \\
&= \kappa_3 \cdot \exp\Bigg[-0.5\left(\mathbf{x}-(\mathbf{A}^{-1}+\mathbf{B}^{-1})^{-1}(\mathbf{A}^{-1}\mathbf{a}+\mathbf{B}^{-1}\mathbf{b})\right)^T(\mathbf{A}^{-1}+\mathbf{B}^{-1}) \\
&\qquad\qquad \left(\mathbf{x}-(\mathbf{A}^{-1}+\mathbf{B}^{-1})^{-1}(\mathbf{A}^{-1}\mathbf{a}+\mathbf{B}^{-1}\mathbf{b})\right)\Bigg] \\
&= \kappa_4 \cdot \text{Norm}_{\mathbf{x}}\Big[(\mathbf{A}^{-1}+\mathbf{B}^{-1})^{-1}(\mathbf{A}^{-1}\mathbf{a}+\mathbf{B}^{-1}\mathbf{b}), (\mathbf{A}^{-1}+\mathbf{B}^{-1})^{-1}\Big]
\end{aligned}$$

Problem 18.6 Derive equation 18.15.

Answer

$$q(\mathbf{z}_{t-1}|\mathbf{z}_t,\mathbf{x}) \quad \propto \quad \text{Norm}_{\mathbf{z}_{t-1}}\left[\frac{1}{\sqrt{1-\beta_t}}\mathbf{z}_t, \frac{\beta_t}{1-\beta_t}\mathbf{I}\right]\text{Norm}_{\mathbf{z}_{t-1}}\left[\sqrt{\alpha_{t-1}}\cdot\mathbf{x}, (1-\alpha_{t-1})\mathbf{I}\right]$$

Using the result from the previous question, the variance of the product distribution is:

$$
\begin{aligned}
\boldsymbol{\Sigma}' &= \left(\frac{1-\beta_t}{\beta_t}+\frac{1}{1-\alpha_{t-1}}\right)^{-1}\mathbf{I}\\
&= \left(\frac{(1-\beta_t)(1-\alpha_{t-1})+\beta_t}{\beta_t(1-\alpha_{t-1})}\right)^{-1}\mathbf{I}\\
&= \left(\frac{1-\beta_t-\alpha_{t-1}+\beta_t\alpha_{t-1}+\beta_t}{\beta_t(1-\alpha_{t-1})}\right)^{-1}\mathbf{I}\\
&= \left(\frac{1-\alpha_{t-1}+\beta_t\alpha_{t-1}}{\beta_t(1-\alpha_{t-1})}\right)^{-1}\mathbf{I}\\
&= \left(\frac{1-\alpha_{t-1}+\beta_t\alpha_{t-1}}{\beta_t(1-\alpha_{t-1})}\right)^{-1}\mathbf{I}\\
&= \left(\frac{1-(1-\beta_t)\alpha_{t-1}}{\beta_t(1-\alpha_{t-1})}\right)^{-1}\mathbf{I}\\
&= \left(\frac{1-\alpha_t}{\beta_t(1-\alpha_{t-1})}\right)^{-1}\mathbf{I}\\
&= \frac{\beta_t(1-\alpha_{t-1})}{1-\alpha_t}\mathbf{I}.
\end{aligned}
$$

The mean is given by:

$$
\begin{aligned}
\boldsymbol{\mu}' &= \frac{\beta_t(1-\alpha_{t-1})}{1-\alpha_t}\left(\frac{1-\beta_t}{\beta_t\sqrt{1-\beta_t}}\mathbf{z}_t+\frac{\sqrt{\alpha_{t-1}}}{1-\alpha_{t-1}}\mathbf{x}\right)\\
&= \frac{(1-\alpha_{t-1})}{1-\alpha_t}\sqrt{(1-\beta_t}\mathbf{z}_t+\frac{\sqrt{\alpha_{t-1}}\beta_t}{1-\alpha_t}\mathbf{x},
\end{aligned}
$$

which gives:

$$
q(\mathbf{z}_{t-1}|\mathbf{z}_t,\mathbf{x}) = \text{Norm}_{\mathbf{z}_{t-1}}\left[\frac{(1-\alpha_{t-1})}{1-\alpha_t}\sqrt{1-\beta_t}\mathbf{z}_t+\frac{\sqrt{\alpha_{t-1}}\beta_t}{1-\alpha_t}\mathbf{x},\ \frac{\beta_t(1-\alpha_{t-1})}{1-\alpha_t}\mathbf{I}\right],
$$

as required.

Problem 18.7 Derive the third line of equation 18.25 from the second line.

Answer

$$
\begin{aligned}
\text{ELBO}\big[\boldsymbol{\phi}_{1\ldots T}\big] &\approx \int q(\mathbf{z}_{1\ldots T}|\mathbf{x})\left(\log\left[Pr(\mathbf{x}|\mathbf{z}_1,\boldsymbol{\phi}_1)\right]+\sum_{t=2}^{T}\log\left[\frac{Pr(\mathbf{z}_{t-1}|\mathbf{z}_t,\boldsymbol{\phi}_t)}{q(\mathbf{z}_{t-1}|\mathbf{z}_t,\mathbf{x})}\right]\right)d\mathbf{z}_{1\ldots T}\\
&\approx \int q(\mathbf{z}_{1\ldots T}|\mathbf{x})\log\left[Pr(\mathbf{x}|\mathbf{z}_1,\boldsymbol{\phi}_1)\right]d\mathbf{z}_{1\ldots T}+\int q(\mathbf{z}_{1\ldots T}|\mathbf{x})\sum_{t=2}^{T}\log\left[\frac{Pr(\mathbf{z}_{t-1}|\mathbf{z}_t,\boldsymbol{\phi}_t)}{q(\mathbf{z}_{t-1}|\mathbf{z}_t,\mathbf{x})}\right]d\mathbf{z}_{1\ldots T}
\end{aligned}
$$

Expanding the first term, we have:

$$\begin{aligned}\int q(\mathbf{z}_{1\ldots T}|\mathbf{x})\log\left[Pr(\mathbf{x}|\mathbf{z}_1,\boldsymbol{\phi}_1)\right]d\mathbf{z}_{1\ldots T} &= \int q(\mathbf{z}_1|\mathbf{x})\log\left[Pr(\mathbf{x}|\mathbf{z}_1,\boldsymbol{\phi}_1)\right]d\mathbf{z}_1\\ &= \mathbb{E}_{\mathbf{q}(z_1|\mathbf{x})}\Bigl[\log\left[Pr(\mathbf{x}|\mathbf{z}_1,\boldsymbol{\phi}_1)\right]\Bigr].\end{aligned}$$

where we have marginalized over all the variables except $\mathbf{z}_1$ in the first line and then written the result as an expectation.

Expanding the second term, we have:

$$\begin{aligned}&\int q(\mathbf{z}_{1\ldots T}|\mathbf{x})\sum_{t=2}^{T}\log\left[\frac{Pr(\mathbf{z}_{t-1}|\mathbf{z}_t,\boldsymbol{\phi}_t)}{q(\mathbf{z}_{t-1}|\mathbf{z}_t,\mathbf{x})}\right]d\mathbf{z}_{1\ldots T}\\
&\qquad= \sum_{t=2}^{T}\int q(\mathbf{z}_{1\ldots T}|\mathbf{x})\log\left[\frac{Pr(\mathbf{z}_{t-1}|\mathbf{z}_t,\boldsymbol{\phi}_t)}{q(\mathbf{z}_{t-1}|\mathbf{z}_t,\mathbf{x})}\right]d\mathbf{z}_{1\ldots T}\\
&\qquad= \sum_{t=2}^{T}\int\!\!\int q(\mathbf{z}_{t-1},\mathbf{z}_t|\mathbf{x})\log\left[\frac{Pr(\mathbf{z}_{t-1}|\mathbf{z}_t,\boldsymbol{\phi}_t)}{q(\mathbf{z}_{t-1}|\mathbf{z}_t,\mathbf{x})}\right]d\mathbf{z}_{t-1}d\mathbf{z}_t\\
&\qquad= \sum_{t=2}^{T}\int\!\!\int q(\mathbf{z}_{t-1}|\mathbf{z}_t|\mathbf{x})q(\mathbf{z}_t|\mathbf{x})\log\left[\frac{Pr(\mathbf{z}_{t-1}|\mathbf{z}_t,\boldsymbol{\phi}_t)}{q(\mathbf{z}_{t-1}|\mathbf{z}_t,\mathbf{x})}\right]d\mathbf{z}_{t-1}d\mathbf{z}_t\\
&\qquad= \sum_{t=2}^{T}\int q(\mathbf{z}_t|\mathbf{x})\int q(\mathbf{z}_{t-1}|\mathbf{z}_t|\mathbf{x})\log\left[\frac{Pr(\mathbf{z}_{t-1}|\mathbf{z}_t,\boldsymbol{\phi}_t)}{q(\mathbf{z}_{t-1}|\mathbf{z}_t,\mathbf{x})}\right]d\mathbf{z}_{t-1}d\mathbf{z}_t\\
&\qquad= -\sum_{t=2}^{T}\int q(\mathbf{z}_t|\mathbf{x})D_{KL}\Bigl[q(\mathbf{z}_{t-1}|\mathbf{z}_t,\mathbf{x})\Bigl|\Bigr|Pr(\mathbf{z}_{t-1}|\mathbf{z}_t,\boldsymbol{\phi}_t)\Bigr]d\mathbf{z}_t\\
&\qquad= -\sum_{t=2}^{T}\mathbb{E}_{q(\mathbf{z}_t|\mathbf{x})}\Bigl[D_{KL}\Bigl[q(\mathbf{z}_{t-1}|\mathbf{z}_t,\mathbf{x})\Bigl|\Bigr|Pr(\mathbf{z}_{t-1}|\mathbf{z}_t,\boldsymbol{\phi}_t)\Bigr]\Bigr]\end{aligned}$$

where we have marginalized over all irrelevant terms $\mathbf{z}_\bullet$ between the second and third times, and factored the remaining joint distribution using the conditional probability rule between the third and fourth lines.

Problem 18.8 The KL-divergence between two normal distributions in D dimensions with means $\mathbf{a}$ and $\mathbf{b}$ and covariance matrices $\mathbf{A}$ and $\mathbf{B}$ is given by:

$$D_{KL}\Bigl[\text{Norm}_{\mathbf{w}}[\mathbf{a},\mathbf{A}]\Bigl|\Bigr|\text{Norm}_{\mathbf{w}}[\mathbf{b},\mathbf{B}]\Bigr] = \frac{1}{2}\left(\text{tr}\left[\mathbf{B}^{-1}\mathbf{A}\right]-d+(\mathbf{a}-\mathbf{b})^T\mathbf{B}^{-1}(\mathbf{a}-\mathbf{b})+\log\left[\frac{|\mathbf{B}|}{|\mathbf{A}|}\right]\right).$$

Substitute the definitions from equation 18.27 into this expression and show that the only term that depends on the parameters $\boldsymbol{\phi}$ is the first term from equation 18.28.

Answer

We start with:

$$\begin{aligned} Pr(\mathbf{z}_{t-1}|\mathbf{z}_t,\boldsymbol{\phi}_t) &= \text{Norm}_{\mathbf{z}_{t-1}}\Big[\mathbf{f}_t[\mathbf{z}_t,\boldsymbol{\phi}_t],\sigma_t^2\mathbf{I}\Big] \\ q(\mathbf{z}_{t-1}|\mathbf{z}_t,\mathbf{x}) &= \text{Norm}_{\mathbf{z}_{t-1}}\left[\frac{(1-\alpha_{t-1})}{1-\alpha_t}\sqrt{1-\beta_t}\mathbf{z}_t+\frac{\sqrt{\alpha_{t-1}}\beta_t}{1-\alpha_t}\mathbf{x},\frac{\beta_t(1-\alpha_{t-1})}{1-\alpha_t}\mathbf{I}\right]. \end{aligned}$$

We denote:

$$\begin{aligned} \mathbf{a} &= \frac{(1-\alpha_{t-1})}{1-\alpha_t}\sqrt{1-\beta_t}\mathbf{z}_t+\frac{\sqrt{\alpha_{t-1}}\beta_t}{1-\alpha_t}\mathbf{x} \\ \mathbf{A} &= \frac{\beta_t(1-\alpha_{t-1})}{1-\alpha_t}\mathbf{I} \\ \mathbf{b} &= \mathbf{f}_t[\mathbf{z}_t,\boldsymbol{\phi}_t] \\ \mathbf{B} &= \sigma_t^2\mathbf{I}. \end{aligned}$$

It's clear that only the quadratic term will remain so:

$$\begin{aligned} D_{KL}\Big[q(\mathbf{z}_{t-1}|\mathbf{z}_t,\mathbf{x})\Big|\Big|Pr(\mathbf{z}_{t-1}|\mathbf{z}_t,\boldsymbol{\phi}_t)\Big] &= \\ \frac{1}{2\sigma_t^2}\left\|\frac{(1-\alpha_{t-1})}{1-\alpha_t}\sqrt{1-\beta_t}\mathbf{z}_t+\frac{\sqrt{\alpha_{t-1}}\beta_t}{1-\alpha_t}\mathbf{x}-\mathbf{f}_t[\mathbf{z}_t,\boldsymbol{\phi}_t]\right\|^2 &+ C, \end{aligned}$$

where C is a constant, as required.

Problem 18.9 If $\alpha_t=\prod_{s=1}^t 1-\beta_s$, then show that:

$$\sqrt{\frac{\alpha_t}{\alpha_{t-1}}}=\sqrt{1-\beta_t}.$$

Answer

$$\begin{aligned} \sqrt{\frac{\alpha_t}{\alpha_t}} &= \sqrt{\frac{\prod_{s=1}^t 1-\beta_s}{\prod_{s=1}^{t-1} 1-\beta_s}} \\ &= \sqrt{(1-\beta_t)} \end{aligned}$$

Problem 18.10 If $\alpha_t=\prod_{s=1}^t 1-\beta_s$, then show that:

$$\frac{(1-\alpha_{t-1})(1-\beta_t)+\beta_t}{(1-\alpha_t)\sqrt{1-\beta_t}}=\frac{1}{\sqrt{1-\beta_t}}.$$

Answer

$$
\begin{aligned}
\frac{(1-\alpha_{t-1})(1-\beta_t)+\beta_t}{(1-\alpha_t)\sqrt{1-\beta_t}} &= \frac{1-\beta_t-\alpha_{t-1}(1-\beta_t)+\beta_t}{(1-\alpha_t)\sqrt{1-\beta_t}}\\
&= \frac{1-\alpha_{t-1}(1-\beta_t)}{(1-\alpha_t)\sqrt{1-\beta_t}}\\
&= \frac{1-\alpha_t}{(1-\alpha_t)\sqrt{1-\beta_t}}\\
&= \frac{1}{\sqrt{1-\beta_t}}.
\end{aligned}
$$

Problem 18.11 Prove equation 18.37.

Answer

We start with:

$$
\begin{aligned}
\mathbf{x} &= \frac{1}{\sqrt{\alpha_1}}\cdot\mathbf{z}_t - \frac{\sqrt{1-\alpha_1}}{\sqrt{\alpha_1}}\cdot\boldsymbol{\epsilon}_{i1}\\
\mathbf{f}_1[\mathbf{z}_1,\boldsymbol{\phi}_1] &= \frac{1}{\sqrt{1-\beta_1}}\mathbf{z}_1 - \frac{\beta_1}{\sqrt{1-\alpha_1}\sqrt{1-\beta_1}}\mathbf{g}_1[\mathbf{z}_1,\boldsymbol{\phi}_1].
\end{aligned}
$$

$$
\begin{aligned}
\frac{1}{2\sigma_1^2}\Bigl\|\mathbf{x}_i-\mathbf{f}_1[\mathbf{z}_{i1},\boldsymbol{\phi}_1]\Bigr\|^2 &= \frac{1}{2\sigma_1^2}\Bigl\|\frac{1}{\sqrt{\alpha_1}}\cdot\mathbf{z}_1-\frac{\sqrt{1-\alpha_1}}{\sqrt{\alpha_1}}\cdot\boldsymbol{\epsilon}_{i1}-\frac{1}{\sqrt{1-\beta_1}}\mathbf{z}_1+\frac{\beta_1}{\sqrt{1-\alpha_1}\sqrt{1-\beta_1}}\mathbf{g}_1[\mathbf{z}_1,\boldsymbol{\phi}_1]\Bigr\|^2\\
&= \frac{1}{2\sigma_1^2}\Bigl\|\frac{1}{\sqrt{1-\beta_1}}\cdot\mathbf{z}_1-\frac{\sqrt{1-\alpha_1}}{\sqrt{1-\beta_1}}\cdot\boldsymbol{\epsilon}_{i1}-\frac{1}{\sqrt{1-\beta_1}}\mathbf{z}_1+\frac{\beta_1}{\sqrt{1-\alpha_1}\sqrt{1-\beta_1}}\mathbf{g}_1[\mathbf{z}_1,\boldsymbol{\phi}_1]\Bigr\|^2\\
&= \frac{1}{2\sigma_1^2}\Bigl\|-\frac{\sqrt{1-\alpha_1}}{\sqrt{1-\beta_1}}\cdot\boldsymbol{\epsilon}_{i1}+\frac{\beta_1}{\sqrt{1-\alpha_1}\sqrt{1-\beta_1}}\mathbf{g}_1[\mathbf{z}_1,\boldsymbol{\phi}_1]\Bigr\|^2\\
&= \frac{1}{2\sigma_1^2}\Bigl\|-\frac{1-\alpha_1}{\sqrt{1-\alpha_1}\sqrt{1-\beta_1}}\cdot\boldsymbol{\epsilon}_{i1}+\frac{\beta_1}{\sqrt{1-\alpha_1}\sqrt{1-\beta_1}}\mathbf{g}_1[\mathbf{z}_1,\boldsymbol{\phi}_1]\Bigr\|^2\\
&= \frac{1}{2\sigma_1^2}\Bigl\|-\frac{\beta_1}{\sqrt{1-\alpha_1}\sqrt{1-\beta_1}}\cdot\boldsymbol{\epsilon}_{i1}+\frac{\beta_1}{\sqrt{1-\alpha_1}\sqrt{1-\beta_1}}\mathbf{g}_1[\mathbf{z}_1,\boldsymbol{\phi}_1]\Bigr\|^2\\
&= \frac{1}{2\sigma_1^2}\Bigl\|\frac{\beta_1}{\sqrt{1-\alpha_1}\sqrt{1-\beta_1}}\mathbf{g}_1[\mathbf{z}_1,\boldsymbol{\phi}_1]-\frac{\beta_1}{\sqrt{1-\alpha_1}\sqrt{1-\beta_1}}\cdot\boldsymbol{\epsilon}_{i1}\Bigr\|^2
\end{aligned}
$$

as required.

Problem 18.12 Classifier-free guidance allows us to create more stereotyped “canonical” images of a given class. When we described transformer decoders, generative adversarial networks, and the GLOW algorithm, we also discussed methods to reduce the amount of variation and produce more stereotyped outputs. What were these? Do you think it’s inevitable that we should limit the output of generative models in this way?

Chapter 19

Reinforcement learning

Problem 19.1 Figure 19.18 in the book shows a single trajectory through the example MDP. Calculate the return for each step in the trajectory given that the discount factor γ is 0.9.

Problem 19.2 Prove the policy improvement theorem. Consider changing from policy π to policy π', where for state s_t the new policy π' chooses the action that maximizes the expected return:

$$\pi'[a_t|s_t] \leftarrow \underset{a_t}{\operatorname{argmax}}\biggl[r[s_t,a_t] + \gamma\cdot\sum_{s_{t+1}} Pr(s_{t+1}|s_t,a_t)v[s_{t+1}|\pi]\biggr].$$

and for all other states, the policies are the same. Show that the value $v[s_t|\pi]$ for the original policy must be less than or equal to $v[s_t|\pi']$ for the new policy (notation indicates using π' for state s_t and π thereafter):

$$\begin{aligned} v[s_t|\pi] &\leq q\Bigl[s_t, \pi'[a_t|s_t]\Bigm|\pi\Bigr] \\ &= \mathbb{E}_{\pi'}\Bigl[r_{t+1} + \gamma\cdot v[s_{t+1}|\pi]\Bigr]. \end{aligned}$$

Hint: Start by writing the term $v[s_{t+1}|\pi]$ in terms of the new policy.

Answer

The general version of the policy improvement theorem considers a pair π and π' of deterministic policies such that, for all states, $q\Bigl[s, \pi'[a|s]\Bigm|\pi\Bigr] \geq\geq v[s|\pi]$. If this is the case, then the policy π' must be as good, or better than π. That is, it must obtain greater or equal expected return from all states: $Vv[s|\pi'] \geq v[s|\pi]$. The problem statement above is just a special case where the policy only differs in a single state.

The policy improvement theorem is easily proved:

$$\begin{aligned}
v[s_t|\pi] &\leq q\Big[s_t, \pi'[a_t|s_t]\Big|\pi\Big] \\
&= \mathbb{E}_{\pi'}\Big[r_{t+1} + \gamma \cdot v[s_{t+1}|\pi]\Big] \\
&\leq \mathbb{E}_{\pi'}\Big[r_{t+1} + \gamma \cdot q\Big[s_{t+1}, \pi'[a_{t+1}|s_{t+1}]\Big|\pi\Big]\Big] \\
&= \mathbb{E}_{\pi'}\Big[r_{t+1} + \gamma \cdot \mathbb{E}_{\pi'}\Big[r_{t+2} + \gamma \cdot v[s_{t+2}|\pi]\Big]\Big] \\
&= \mathbb{E}_{\pi'}\Big[r_{t+1} + \gamma \cdot r_{t+2} + \gamma^2 \cdot v[s_{t+2}|\pi]\Big] \\
&\leq \mathbb{E}_{\pi'}\Big[r_{t+1} + \gamma \cdot r_{t+2} + \gamma^2 \cdot q\Big[s_{t+2}, \pi'[a_{t+2}|s_{t+2}]\Big|\pi\Big]\Big] \\
&\vdots \\
&\leq \mathbb{E}_{\pi'}\Big[r_{t+1} + \gamma \cdot r_{t+2} + \gamma^2 r_{t+3} + \gamma^3 r_{t+4} + \ldots\Big] \\
&= v[s_t|\pi']
\end{aligned}$$

In this derivation, the term $\mathbb{E}_{\pi'}\Big[r_{t+2} + \gamma \cdot v[s_{t+2}|\pi]\Big]\Big]$ indicates that we take the expected state transitions up to state s_{t+1} with respect to π' and the remaining transitions with respect to π.

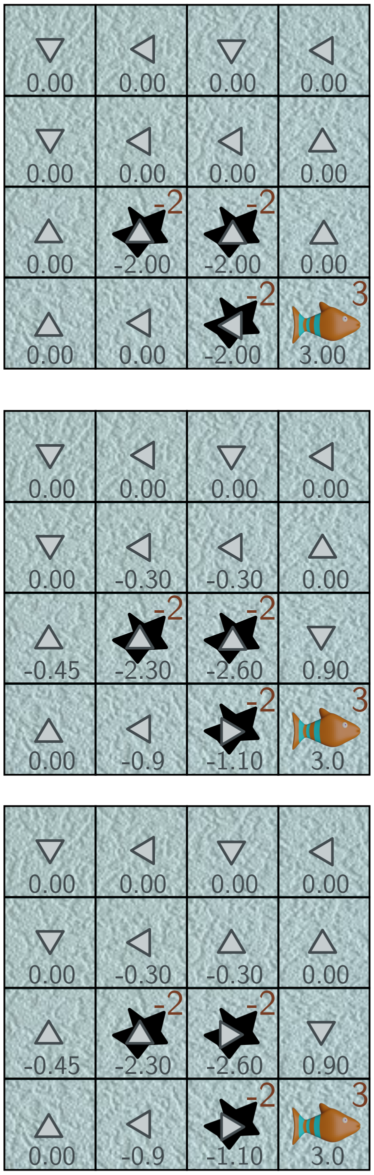
See these notes or this webpage for a proof in different notation.

Problem 19.3 Show that when the values of state and policy are initialized as in figure 19.10a, they become those in figure 19.10b after two iterations of (i) policy evaluation (in which all states are updated based on their current values and then replace the previous ones) and (ii) policy improvement. The state transition allots half the probability to the direction the policy indicates and divides the remaining probability equally between the other valid actions. The reward function returns -2 regardless of the action when the penguin leaves a hole. The reward function returns +3 regardless of the action when the penguin leaves the fish tile, and the episode ends, so the fish tile has a value of +3.

Problem 19.4 The Boltzmann policy strikes a balance between exploration and exploitation by basing the action probabilities $\pi[a|s]$ on the current state-action reward function $q[s,a]$:

$$\pi[a|s] = \frac{\exp[q[s,a]/\tau]}{\sum_{a'} \exp[q[s,a']/\tau]}.$$

Explain how the temperature parameter τ can be varied to prioritize exploration or exploitation.



$$
\begin{aligned}
v[s_1] &= 0.0 + 0.9(0.00 \times 0.0 + 0.50 \times 0.0 + 0.50 \times 0.0 + 0.00 \times 0.0) = 0.0\\
v[s_2] &= 0.0 + 0.9(0.00 \times 0.0 + 0.25 \times 0.0 + 0.25 \times 0.0 + 0.50 \times 0.0) = 0.0\\
v[s_3] &= 0.0 + 0.9(0.00 \times 0.0 + 0.25 \times 0.0 + 0.50 \times 0.0 + 0.25 \times 0.0) = 0.0\\
v[s_4] &= 0.0 + 0.9(0.00 \times 0.0 + 0.00 \times 0.0 + 0.50 \times 0.0 + 0.50 \times 0.0) = 0.0\\
v[s_5] &= 0.0 + 0.9(0.25 \times 0.0 + 0.25 \times 0.0 + 0.50 \times 0.0 + 0.00 \times 0.0) = 0.0\\
v[s_6] &= 0.0 + 0.9(0.17 \times 0.0 + 0.17 \times 0.0 + 0.50 \times 0.0 + 0.17 \times 0.0) = 0.0\\
v[s_7] &= 0.0 + 0.9(0.17 \times 0.0 + 0.17 \times 0.0 + 0.17 \times 0.0 + 0.50 \times 0.0) = 0.0\\
v[s_8] &= 0.0 + 0.9(0.50 \times 0.0 + 0.00 \times 0.0 + 0.25 \times 0.0 + 0.25 \times 0.0) = 0.0\\
v[s_9] &= 0.0 + 0.9(0.50 \times 0.0 + 0.25 \times 0.0 + 0.25 \times 0.0 + 0.00 \times 0.0) = 0.0\\
v[s_{10}] &= -2.0 + 0.9(0.50 \times 0.0 + 0.17 \times 0.0 + 0.17 \times 0.0 + 0.17 \times 0.0) = -2.0\\
v[s_{11}] &= -2.0 + 0.9(0.50 \times 0.0 + 0.17 \times 0.0 + 0.17 \times 0.0 + 0.17 \times 0.0) = -2.0\\
v[s_{12}] &= 0.0 + 0.9(0.50 \times 0.0 + 0.00 \times 0.0 + 0.25 \times 0.0 + 0.25 \times 0.0) = 0.0\\
v[s_{13}] &= 0.0 + 0.9(0.50 \times 0.0 + 0.50 \times 0.0 + 0.00 \times 0.0 + 0.00 \times 0.0) = 0.0\\
v[s_{14}] &= 0.0 + 0.9(0.25 \times 0.0 + 0.25 \times 0.0 + 0.00 \times 0.0 + 0.50 \times 0.0) = 0.0\\
v[s_{15}] &= -2.0 + 0.9(0.25 \times 0.0 + 0.25 \times 0.0 + 0.00 \times 0.0 + 0.50 \times 0.0) = -2.0\\
v[s_{16}] &= 3.0
\end{aligned}
$$

$$
\begin{aligned}
v[s_1] &= 0.0 + 0.9(0.00 \times 0.0 + 0.50 \times 0.0 + 0.50 \times 0.0 + 0.00 \times 0.0) = 0.00\\
v[s_2] &= 0.0 + 0.9(0.00 \times 0.0 + 0.25 \times 0.0 + 0.25 \times 0.0 + 0.50 \times 0.0) = 0.00\\
v[s_3] &= 0.0 + 0.9(0.00 \times 0.0 + 0.25 \times 0.0 + 0.50 \times 0.0 + 0.25 \times 0.0) = 0.00\\
v[s_4] &= 0.0 + 0.9(0.00 \times 0.0 + 0.00 \times 0.0 + 0.50 \times 0.0 + 0.50 \times 0.0) = 0.00\\
v[s_5] &= 0.0 + 0.9(0.25 \times 0.0 + 0.25 \times 0.0 + 0.50 \times 0.0 + 0.00 \times 0.0) = 0.00\\
v[s_6] &= 0.0 + 0.9((1/6) \times 0.0 + (1/6) \times 0.0 + (1/6) \times -2.0 + 0.50 \times 0.0) = -0.3\\
v[s_7] &= 0.0 + 0.9((1/6) \times 0.0 + (1/6) \times 0.0 + (1/6) \times -2.0 + 0.50 \times 0.0) = -0.3\\
v[s_8] &= 0.0 + 0.9(0.50 \times 0.0 + 0.00 \times 0.0 + 0.25 \times 0.0 + 0.25 \times 0.0) = 0.00\\
v[s_9] &= 0.0 + 0.9(0.50 \times 0.0 + 0.25 \times -2.0 + 0.25 \times 0.0 + 0.00 \times 0.0) = 0.45\\
v[s_{10}] &= -2.0 + 0.9(0.50 \times 0.0 + (1/6) \times -2.0 + (1/6) \times 0.0 + (1/6) \times 0.0) = -2.3\\
v[s_{11}] &= -2.0 + 0.9(0.50 \times 0.0 + (1/6) \times 0.0 + (1/6) \times -2.0 + (1/6) \times -2.0) = -2.6\\
v[s_{12}] &= 0.0 + 0.9(0.25 \times 0.0 + 0.00 \times 0.0 + 0.50 \times 3.0 + 0.25 \times -2.0) = 0.90\\
v[s_{13}] &= 0.0 + 0.9(0.50 \times 0.0 + 0.50 \times 0.0 + 0.00 \times 0.0 + 0.00 \times 0.0) = 0.00\\
v[s_{14}] &= 0.0 + 0.9(0.25 \times -2.0 + 0.25 \times -2.0 + 0.00 \times 0.0 + 0.50 \times 0.0) = 0.90\\
v[s_{15}] &= -2.0 + 0.9(0.25 \times -2.0 + 0.50 \times 3.0 + 0.00 \times 0.0 + 0.25 \times 0.0) = -1.10\\
v[s_{16}] &= 3.0
\end{aligned}
$$

Figure 19.1 Answer to problem 19.3.

Problem 19.5 When the learning rate α is one, the Q-Learning update is given by:

$$
\mathrm{f}[\mathrm{q}[s,a]] = r[s,a] + \gamma \cdot \max_a \bigl[\mathrm{q}[s',a]\bigr].
$$

where s is a state and s' is the subsequent state. Show that this is a contraction mapping (equation 16.20), so that:

$$\Big|\Big|\mathrm{f}[\mathrm{q}_1[s,a]] - \mathrm{f}[\mathrm{q}_2[s,a]]\Big|\Big|_\infty < \Big|\Big|\mathrm{q}_1[s,a] - \mathrm{q}_2[s,a]\Big|\Big|_\infty \qquad \forall\, q_1, q_2.$$

where $||\bullet||_\infty$ represents the ℓ_∞ norm. It follows that a fixed point will exist by Banach's theorem and that the updates will eventually converge.

Answer

$$\begin{aligned}
\Big|\Big|\mathrm{f}[\mathrm{q}_1[s,a]] - \mathrm{f}[\mathrm{q}_2[s,a]]\Big|\Big|_\infty &= \Big|\Big|r[a,s] + \gamma\cdot\max_a\big[\mathrm{q}_1[s'_1,a]\big] \\
&\qquad -r[a,s] - \gamma\cdot\max_a\big[\mathrm{q}_2[s'_2,a]\big]\Big|\Big|_\infty \\
&= \Big|\Big|\gamma\cdot\max_a\big[\mathrm{q}_1[s'_1,a]\big] - \gamma\cdot\max_a\big[\mathrm{q}_2[s'_2,a]\big]\Big|\Big|_\infty \\
&= \gamma\Big|\Big|\max_a\big[\mathrm{q}_1[s'_1,a]\big] - \max_a\big[\mathrm{q}_2[s'_2,a_{]}\big]\Big|\Big|_\infty \\
&\leq \gamma\Big|\Big|\max_a\big[\mathrm{q}_1[s'_1,a] - \mathrm{q}_2[s'_2,a]\big]\Big|\Big|_\infty \\
&\leq \gamma\Big|\Big|\max_{a,s}\big[\mathrm{q}_1[s,a] - \mathrm{q}_2[s,a]\big]\Big|\Big|_\infty \\
&= \gamma\Big|\Big|\mathrm{q}_1[s,a] - \mathrm{q}_2[s,a]\Big|\Big|_\infty \\
&\leq \Big|\Big|\mathrm{q}_1[s,a] - \mathrm{q}_2[s,a]\Big|\Big|_\infty
\end{aligned}$$

We move from line 3 to line four by noting that line 4 is less constrained than line 3 and similarly between lines 4 and 5. The max operator disappears since the ℓ^∞ norm takes the maximum anyway.

Problem 19.6 Show that:

$$\mathbb{E}_{\boldsymbol{\tau}}\left[\frac{\partial}{\partial\boldsymbol{\theta}}\log\big[Pr(\boldsymbol{\tau}|\boldsymbol{\theta})\big]b\right] = 0,$$

where b does not depend on τ, and so adding a baseline update doesn't change the expected policy gradient update.

Problem 19.7 Suppose that we want to estimate a quantity $\mathbb{E}[a]$ from samples $a_1, a_2 \ldots a_I$. Consider that we also have paired samples $b_1, b_2 \ldots b_I$ that are samples that co-vary with a where $\mathbb{E}[b] = \mu_b$. We define a new variable:

$$a' = a - c(b - \mu_b)$$

Show that $\text{Var}[a'] \leq \text{Var}[a]$ when the constant c is chosen judiciously. Find an expression for the optimal value of c.

Answer

Start by computing an expession for the mean of a':

$$\mathbb{E}[a'] = \mathbb{E}[a] - \mathbb{E}[c(b - \mu_b)] = \mathbb{E}[a].$$

Then compute an expression for the variance of a':

$$\begin{aligned}
\text{Var}[a'] &= \mathbb{E}[a'^2] - \mathbb{E}[a]^2 \\
&= \mathbb{E}[(a - c(b - \mu_b))^2] - \mathbb{E}[a]^2 \\
&= \mathbb{E}[a^2] + \mathbb{E}[c^2(b - \mu_b)^2] - 2\mathbb{E}[a \cdot c(b - \mu_b)] - \mathbb{E}[a]^2 \\
&= \text{Var}[a] + c^2\mathbb{E}[(b - \mu_b)^2] - 2c\mathbb{E}[a \cdot (b - \mu_b)] \\
&= \text{Var}[a] + c^2\text{Var}[b] - 2c\text{Covar}[a, b].
\end{aligned}$$

We want this to be minimized so we take the derivative with respect to c and set to zero to get:

$$c^* = \frac{\text{Covar}[a, b]}{\text{Var}[b]}.$$

If we substitute this into the last line of the expression for $\text{Var}[a']$ and simplify, we get:

$$\text{Var}[a'] = \text{Var}[a] - \frac{\text{Covar}[a, b]^2}{\text{Var}[b]} = (1 - \rho^2_{a,b})\text{Var}[a],$$

where $0 \leq \rho_{a,b} \leq 1$ is the correlation coefficient which is defined as:

$$\rho_{a,b} = \frac{\text{Covar}[a, b]}{\sqrt{\text{Var}[a]\text{Var}[b]}}.$$

Consequently, $1 - \rho^2_{a,b}$ must be less than one and the variance is reduced.

Problem 19.8 The estimate of the gradient in equation 19.34 can be written as:

$$\mathbb{E}_{\boldsymbol{\tau}}\Big[\text{g}[\boldsymbol{\theta}](r[\boldsymbol{\tau}_t] - b)\Big],$$

where

$$\mathrm{g}[\boldsymbol{\theta}, \tau] = \sum_{t=1}^{T} \frac{\partial \log\bigl[Pr(a_t|\mathbf{s}_t, \boldsymbol{\theta})\bigr]}{\partial \boldsymbol{\theta}},$$

and

$$r[\boldsymbol{\tau}] = \sum_{k=t}^{T} r_k.$$

Show that the value of b that minimizes the variance of the gradient estimate is given by:

$$b = \frac{\mathbb{E}\bigl[\mathrm{g}[\boldsymbol{\theta}, \boldsymbol{\tau}]^2 r[\boldsymbol{\tau}]\bigr]}{\mathbb{E}\bigl[\mathrm{g}[\boldsymbol{\theta}, \boldsymbol{\tau}]^2\bigr]}.$$

Chapter 20

Why does deep learning work?

Problem 20.1 Consider the ImageNet image classification task in which the input images contain 224×224×3 RGB values. Consider coarsely quantizing these inputs into ten bins per RGB value and training with $\sim 10^7$ training examples. How many possible inputs are there per training data point?

Problem 20.2 Consider figure 20.1. Why do you think that the algorithm fits the data faster when the pixels are randomized relative to when the labels are randomized?

Problem 20.3 Figure 20.2 shows a non-stochastic fitting process with a fixed learning rate successfully fitting random data. Does this imply that the loss function has no local minima? Does this imply that the function is convex? Justify your answer and give a counter-example if you think either statement is false.

Chapter 21

Deep learning and ethics?

(Answers intentionally missing!)