Universidad Nacional de Colombia
Departarmento de Matemáticas

Matemáticas para el Aprendizaje de Máquina

Juan Antonio Rodríguez

2024

Problems Section 4

1. Problem 4.1 Consider composing the two neural networks in figure 4.8. Draw a plot of the relationship between the input x and output y' for $x \in [-1, 1]$.

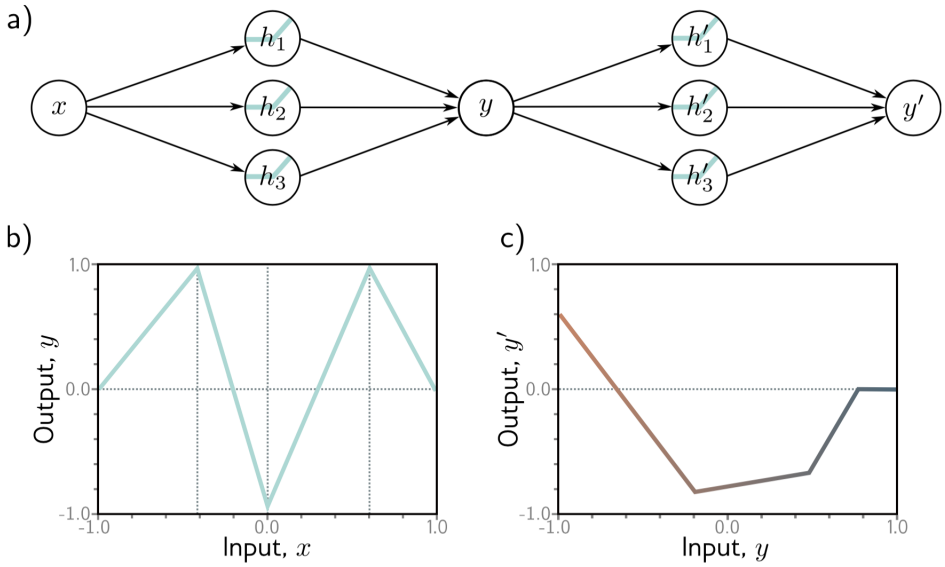
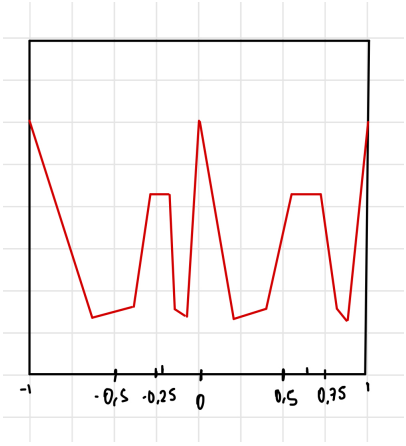


Figure 4.8 Composition of two networks for problem 4.1. a) The output y of the first network becomes the input to the second. b) The first network computes this function with output values $y \in [-1, 1]$. c) The second network computes this function on the input range $y \in [-1, 1]$.

Plot:



2. Problem 4.2 Identify the four hyperparameters in figure 4.6.

   (a) $K = 3$

   (b) $D_1 = 4$

   (c) $D_2 = 2$

   (d) $D_3 = 3$

   - $D_i = 3$
   - $D_o = 2$

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

$$ReLU[\beta_1 + \lambda_1 \cdot \Omega_1 ReLU[\beta_0 + \lambda_0 \cdot \Omega_0 x]] = \lambda_0\lambda_1 \cdot ReLU[\frac{1}{\lambda_0\lambda_1}\beta_1 + \Omega_1 ReLU[\frac{1}{\lambda_0}\beta_0 + \Omega_0 x]]$$

   where λ_0 and λ_1 are non-negative scalars. From this, we see that the weight matrices can be rescaled 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.

   Proof. Suppose λ_0, λ_1 are non-negative scalars and Ω_i, β_i are real parameters (or a matrix of real parameters).

$$\begin{aligned}
&ReLU[\beta_1 + \lambda_1 \cdot \Omega_1 ReLU[\beta_0 + \lambda_0 \cdot \Omega_0 x]] && \text{Using common factor factorization,} \\
&= ReLU[\beta_1 + \lambda_1\Omega_1 ReLU[\lambda_0(\frac{\beta_0}{\lambda_0} + \Omega_0 x)]] && \text{Using the non-negative homogeneity property} \\
&= ReLU[\beta_1 + \lambda_0\lambda_1\Omega_1 ReLU[\frac{1}{\lambda_0}\beta_0 + \Omega_0 x]] && \text{Using common factor factorization,} \\
&= ReLU[\lambda_0\lambda_1(\frac{1}{\lambda_0\lambda_1}\beta_1 + \Omega_1 ReLU[\frac{1}{\lambda_0}\beta_0 + \Omega_0 x)]] && \text{Using the non-negative homogeneity property} \\
&= \lambda_0\lambda_1 \cdot ReLU[\frac{1}{\lambda_0\lambda_1}\beta_1 + \Omega_1 ReLU[\frac{1}{\lambda_0}\beta_0 + \Omega_0 x]]
\end{aligned}$$

   Which is what we wanted, thus ending the proof. □

4. 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 $D1 = 20$, $D2 = 10$, and $D3 = 7$, 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 $\boldsymbol{\beta}_\bullet$?

- Form of equation 4.15

$$\begin{aligned} h_1 &= a[\beta_0 + \Omega_0 x] \\ h_2 &= a[\beta_1 + \Omega_1 h_1] \\ h_3 &= a[\beta_2 + \Omega_2 h_2] \\ y &= \beta_3 + \Omega_3 h_3 \end{aligned}$$

- Form of the equation 4.16

$$y = \beta_3 + \Omega_3 a[\beta_2 + \Omega_2 a[\beta_1 + \Omega_1 a[\beta_0 + \Omega_0 x]]]$$

Dimensions:

- $\Omega_0 : 20 \times 5$
- $\Omega_1 : 10 \times 20$
- $\Omega_2 : 7 \times 10$
- $\Omega_3 : 4 \times 7$
- $\beta_0 : 20 \times 1$
- $\beta_1 : 10 \times 1$
- $\beta_2 : 7 \times 1$
- $\beta_3 : 4 \times 1$

5. 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?

   - Depth: 20 (K)
   - Width: 30 (D_k)

6. Problem 4.6 Consider a network with $D_i = 1$ input, $D_o = 1$ output, and $K = 10$ 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.

   If we were to increase the depth by one, we are adding one layer of hidden units to the network, analyzing the dimensions of the matrices in equation (4.?) we can see that we would be adding $D_k + D_k * D_{k-1}$ new parameters.

If we were to increase the width by one, we are adding one hidden unit to each layer, this would add $D_i + \sum_{j=1}^{k-1} D_j$. If we suppose every layer has the same amount of hidden units, the previous expression would be equal to $D_i + D_k * D_{k-1}$

From the reasoning above, we can conclude that if there are more inputs than hidden units in layer k, increasing the width by one would result in more paramters than incresing the depth by one. Otherwise, incresing the depth by one would result in more parameters than the alternative.

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

   We can choose the values as follow:

   $\phi_0 = 0 \quad \phi_1 = 1 \quad \phi_2 = 1.3 \quad \phi_3 = 0.66 \quad \theta_{10} = 0$

   $\theta_{11} = 1 \quad \theta_{20} = -0.3 \quad \theta_{21} = 0.5 \quad \theta_{30} = -0.3 \quad \theta_{31} = 1$

   The shallow network defined using this parameters, gives us a graph in which the identity function is defined in the range $[0; 0.3]$.

8. 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 $\phi_0 + \phi_1 h1 + \phi_2 h2 + \phi_3 h3$ 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?

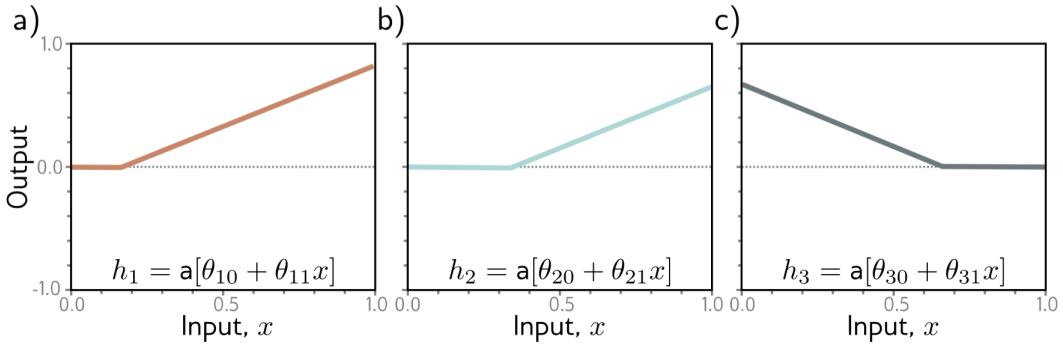
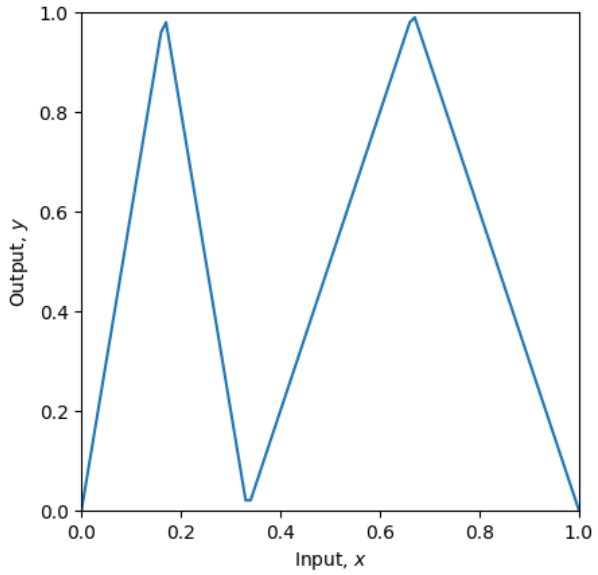


Figure 4.9 Hidden unit activations for problem 4.8. a) First hidden unit has a joint at position $x = 1/6$ and a slope of one in the active region. b) Second hidden unit has a joint at position $x = 2/6$ and a slope of one in the active region. c) Third hidden unit has a joint at position $x = 4/6$ and a slope of minus one in the active region.

$\phi_0 = 4 \quad \phi_1 = -12 \quad \phi_2 = 9 \quad \phi_3 = -6$

This parameters generate the following output:



If we compose this output with itself, the resulting graph would have one copy of the graph above squished into each of the linear regions. Following this, the resulting graph would have 4 linear regions in each of the original 4 linear regions, thus in total it would have $4^2 = 16$ linear regions.

Using the reasoning above, if we compose this output with itself K times, we would have in total 4^K linear regions in the final output graph.

9. 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?

   Yes, the problem above has 3 hidden units creating 4 linear regions. I.e, one linear region more than the number of hidden units. Using appropiate parameters, a shallow network containing 2 hidden units, would create a graph with 3 linear regions, and, depending on the parameters, we could maje the graph oscilate between zero and one just like the graph above. Following this, a shallow network with 4 hidden units would create a graph with 5 linear regions that could oscilate between zero and one.

   In conclusion, one can have a shallow netowrk with n hidden units crate a graph with $n+1$ linear regions that oscilate between zero and one, depending on the parameters chosen.

10. 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.

Proof. Let's analyse the number of parameters of the neural network dividing the network into three parts:

(a) We can start by counting the number of parameters in each layer, not including the first one and the output one. Each layer has $D \times D$ parameters in Ω_i, plus D parameters in each β_i. Since we have k layers, in this part we have $(k-1)D(D+1)$ parameters.

(b) Then we can analyse the first layer, Ω_0 has D parameters, and β_0 has also D parameters. So we have $2D$ parameters.

(c) We finaly analyse the final step (the output one) of the network. For this one, we have that Ω_k has D parameters and β_k has 1 parameter since we have only one output. So, for this part, we have $D+1$ parameters.

Summing everything, we have:

$$(k-1)D(D+1) + 2D + D + 1 = (k-1)D(D+1) + 3D + 1$$

Which is what we wanted to prove, thus concluding this proof. □

11. 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?

Number of parameters:

- Shallow Network.

$$\begin{aligned} N &= (D_i + 1)D + D + 1 \\ &= (2)95 + 95 + 1 \\ &= 286 \end{aligned}$$

- Deep Network.

$$\begin{aligned} N &= 3D + 1 + (k-1)D(D+1) \\ &= 3(5) + 1 + (10)(5)6 \\ &= 316 \end{aligned}$$

Number of regions:

- Shallow Network: (Using Zaslavsky)

$$\begin{aligned} N &= \sum_{j=0}^{D_1} \binom{D}{j} \\ &= \sum_{j=0}^{1} \binom{95}{j} \\ &= \binom{95}{0} + \binom{95}{1} \\ &= 1 + 95 \\ &= 96 \end{aligned}$$

- Deep Network: We have 5 hidden units per layer, so each layer would generate a line that has 6 linear regions. We also hage that the network has 10 layers. Using the folding analogy to do our analysis, we can conclude that in the resukt graph, we can see the last layer being copied onto the previous layer, each linear region of the previous layer receiving one copy of the last layer and so on. Using this analysis, we can conclude that in this deep neural network, the resulting graph would have around 5^{10} linear regions.

Which is faster? One can assume that the shallow one may be faster to do inference, due to having only one layer. However, the number of hidden units is something we have to take into consideration, as well as the number of parameters. Using both numbers as an indication, in this case, the difference in the number of parameters between both networks is not big, the shallow one having less, the deep neural network has less hidden units though.

Taking this things into consideration, we can conclude that the shallow network would be slightly faster becuase of it having to do less claculations despite the fact that it also has more hidden units. However, the deep neural network, despite being slower, provide us greater precision and understanding of the model we want to predict, due to the fact that the number of regions is far greater than if using the shallow network.