Deep Learning **Summer Term 2024**

`http://ml3.leuphana.de/lectures/summer24/DL`
Machine Learning Group, Leuphana University of Lüneburg

Soham Majumder (soham.majumder@leuphana.de)

Classwork

5. The Hyperbolic Tangent Activation Function

Another popular activation function is the hyperbolic tangent $\tanh : \mathbb{R} \to (-1, 1)$ given by

$$\tanh(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)}.$$

(i) Show that the derivative of the hyperbolic tangent with respect to x is

$$\frac{\partial \tanh(x)}{\partial x} = 1 - (\tanh(x))^2 .$$

(ii) Show that the hyperbolic tangent can be written in terms of the sigmoid function as follows:

$$\tanh(x) = 2\sigma(2x) - 1.$$

Solution

(i)

$$\begin{aligned}
\frac{\partial \tanh(x)}{\partial x} &= \frac{\partial}{\partial x} \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)} \\
&= \frac{(\exp(x) + \exp(-x)) \cdot (\exp(x) + \exp(-x)) - (\exp(x) - \exp(-x)) \cdot (\exp(x) - \exp(-x))}{(\exp(x) + \exp(-x))^2} \\
&= \frac{(\exp(x) + \exp(-x)) \cdot (\exp(x) + \exp(-x))}{(\exp(x) + \exp(-x))^2} - \frac{(\exp(x) - \exp(-x)) \cdot (\exp(x) - \exp(-x))}{(\exp(x) + \exp(-x))^2} \\
&= \frac{(\exp(x) + \exp(-x))^2}{(\exp(x) + \exp(-x))^2} - \left(\frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)}\right)^2 \\
&= 1 - (\tanh(x))^2 .
\end{aligned}$$

(ii) Remember that $\sigma(x) = \frac{1}{1+\exp(-x)}$. We can re-write the hyperbolic tangent as follows:

$$\begin{aligned}
\tanh(x) + 1 &= \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)} + 1 = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)} + \frac{\exp(x) + \exp(-x)}{\exp(x) + \exp(-x)} \\
&= \frac{\exp(x) - \exp(-x) + \exp(x) + \exp(-x)}{\exp(x) + \exp(-x)} = \frac{2\exp(x)}{\exp(x) + \exp(-x)} \\
&= 2\frac{1}{1 + \exp(-2x)} = 2\sigma(2x)
\end{aligned}$$

6. Multiclass Classification

Part 1

Let $\mathbf{x} \in \mathbb{R}^d$ be a vector. The softmax function softmax : $\mathbb{R}^d \to (0,1)^d$ is given by

$$\mathbf{p} = \mathrm{softmax}(\mathbf{x}) = \begin{pmatrix} \exp(x_1) \\ \exp(x_2) \\ \vdots \\ \exp(x_d) \end{pmatrix} \Bigg/ \left(\sum_{j=1}^{d} \exp(x_j) \right)$$

and returns a probability distribution $\mathbf{p}$, i.e.,

$$p_j = \frac{\exp(x_j)}{\sum_{k=1}^{d} \exp(x_k)} \geq 0$$

and $\sum_{j=1}^{d} p_j = 1$.

A suitable loss function is the cross-entropy loss. It is given by

$$H(\mathbf{p}, \mathbf{y}) = -\sum_{j=1}^{d} y_j \log(p_j),$$

where $\mathbf{y}$ is a one-hot encoded target vector and $\mathbf{p}$ is the output of the softmax layer.

(i) Show that the derivative of the softmax function with respect to $\mathbf{x}$ is

$$\frac{\partial p_j}{\partial x_i} = p_j(\delta_{ij} - p_i),$$

where δ_{ij} is 1 if $i = j$ and 0 otherwise.

(ii) Show that the derivative of the cross-entropy loss in combination with the softmax function with respect to $\mathbf{x}$ is

$$\frac{\partial H(\mathbf{p}, \mathbf{y})}{\partial \mathbf{x}} = \mathbf{p} - \mathbf{y}.$$

Hint: For the first part, you should do a case distinction ($i = j$ and $i \neq j$) of $\frac{\partial p_j}{\partial x_i}$. In the second part, you need the chain rule when considering $\frac{\partial \log p_j}{\partial x_i}$.

Solution

(i) For $i = j$ we have

$$\begin{aligned}
\frac{\partial p_j}{\partial x_i} &= \frac{\exp(x_j)\left(\sum_k \exp(x_k)\right) - \exp(x_j)\exp(x_i)}{\left(\sum_k \exp(x_k)\right)^2} && \text{(derivative of a rational)} \\
&= \frac{\exp(x_j)}{\sum_k \exp(x_k)} \cdot \frac{\left(\sum_k \exp(x_k)\right) - \exp(x_i)}{\sum_k \exp(x_k)} && \text{(separate } \exp(x_j)\text{)} \\
&= \frac{\exp(x_j)}{\sum_k \exp(x_k)} \cdot \left(\frac{\left(\sum_k \exp(x_k)\right)}{\sum_k \exp(x_k)} - \frac{\exp(x_i)}{\sum_k \exp(x_k)} \right) \\
&= p_j \cdot (1 - p_i)
\end{aligned}$$

and for $i \neq j$ we get

$$\begin{aligned}\frac{\partial p_j}{\partial x_i} &= \frac{0 - \exp(x_j)\exp(x_i)}{\left(\sum_k \exp(x_k)\right)^2} \\ &= \frac{\exp(x_j)}{\sum_k \exp(x_k)} \cdot \frac{-\exp(x_i)}{\sum_k \exp(x_k)} \\ &= p_j \cdot (0 - p_i).\end{aligned}$$

Combined, that yields

$$\frac{\partial p_j}{\partial x_i} = p_j(\delta_{ij} - p_i),$$

where δ_{ij} is 1 if $i = j$ and 0 otherwise.

(ii)

$$\begin{aligned}\frac{\partial H(\mathbf{p}, \mathbf{y})}{\partial x_i} &= \frac{\partial - \sum_{j=1}^{d} y_j \log(p_j)}{\partial x_i} = -\sum_{j=1}^{d} y_j \frac{\partial \log(p_j)}{\partial x_i} = -\sum_{j=1}^{d} y_j \frac{\partial \log(p_j)}{\partial p_j} \frac{\partial p_j}{\partial x_i} \\ &= -\sum_{j=1}^{d} y_j \frac{1}{p_j} p_j(\delta_{ij} - p_i) = -\sum_{j=1}^{d} y_j(\delta_{ij} - p_i) = -\sum_{j=1}^{d} y_j\delta_{ij} - y_j p_i \\ &= -\sum_{j=1}^{d} y_j\delta_{ij} + p_i \sum_{j=1}^{d} y_j = -y_i + p_i\end{aligned}$$

Here, we used the fact that $\mathbf{y}$ is a one-hot encoded target vector, hence $\sum_{j=1}^{d} y_j = 1$.

Part 2

The log-linear model for logistic regression allows us to derive the softmax function to model the probabilities in multiclass classification. For a problem with c classes, start by writing the log-probability of each class as a linear function of the inputs and the partition ("normalization") term $-\log Z$

$$\begin{aligned}\log P(Y = 1|X = x) &= w_1 x + b_1 - \log Z, \\ \log P(Y = 2|X = x) &= w_2 x + b_2 - \log Z, \\ &\vdots \\ \log P(Y = c|X = x) &= w_c x + b_c - \log Z,\end{aligned}$$

and using $\sum_{j=1}^{c} P(Y = j|X = x) = 1$, show how this model is equivalent to modeling the class probabilities with the softmax function.

Solution

First, we rewrite the log-linear models as probabilities by exponentiating both sides:

$$\begin{aligned}
P(Y=1|X=x) &= \frac{1}{Z}\exp(w_1 x + b_1),\\
P(Y=2|X=x) &= \frac{1}{Z}\exp(w_2 x + b_2),\\
&\vdots\\
P(Y=c|X=x) &= \frac{1}{Z}\exp(w_c x + b_c).
\end{aligned}$$

We can now determine Z by using $\sum_{j=1}^{c} P(Y=j|X=x) = 1$:

$$\begin{aligned}
1 &= \sum_{j=1}^{c} \frac{1}{Z}\exp(w_j x + b_j) && \text{(multiplying both sides by } Z\text{)}\\
Z &= \sum_{j=1}^{c} \exp(w_j x + b_j)
\end{aligned}$$

Thus,

$$P(Y=i|X=x) = \frac{\exp(w_i x + b_i)}{\sum_{j=1}^{c}\exp(w_j x + b_j)} = p_i,$$

where p_i is the i-th component of $\operatorname{softmax}((w_1x+b_1, w_2x+b_2, \ldots, w_cx+b_c)^\top)$.

7. Convolutions

Let $\mathbf{x} \in \mathbb{R}^{2\times 3}$ be an input tensor and $\mathbf{f} \in \mathbb{R}^{3\times 1}$ be a filter/kernel. We want to compute the convolution of $\mathbf{x}$ and $\mathbf{f}$ using a padding of $(1, 0)$ on the input tensor.

(i) Let $\mathbf{x} = \begin{pmatrix} 4 & -2 & 1 \\ 3 & 8 & 5 \end{pmatrix}$ and $\mathbf{f} = \begin{pmatrix} 2 \\ 4 \\ -1 \end{pmatrix}$ be specific tensors. Compute the convolution.

(ii) Consider the vectorized form of the input $\mathbf{x}$ and write down the solution of the convolution as a matrix-vector product $\mathbf{wx} = \mathbf{s}$. The filter/kernel $\mathbf{f}$ is now a part of the weight matrix $\mathbf{w}$. Convince yourself that the weight matrix is sparse* and weights are shared.

Hint: The dimensionalities are $\mathbf{x} \in \mathbb{R}^{12}$, $\mathbf{w} \in \mathbb{R}^{6\times 12}$, and $\mathbf{s} \in \mathbb{R}^{6}$.

*In a sparse matrix most of the elements are zero.

Solution

(i) Using a padding of $(1,0)$, the input $\begin{pmatrix} 0 & 0 & 0 \\ 4 & -2 & 1 \\ 3 & 8 & 5 \\ 0 & 0 & 0 \end{pmatrix}$ and the kernel $\begin{pmatrix} 2 \\ 4 \\ -1 \end{pmatrix}$ yield $\begin{pmatrix} 13 & -16 & -1 \\ 20 & 28 & 22 \end{pmatrix}$

(ii) If the vectorization is done row-wise, we obtain

$$\mathbf{x} = \text{vec}\left[\begin{pmatrix} x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 \\ x_7 & x_8 & x_9 \\ x_{10} & x_{11} & x_{12} \end{pmatrix}\right] = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{11} \\ x_{12} \end{pmatrix}$$

$$\mathbf{wx} = \mathbf{s}$$

$$\begin{pmatrix} f_1 & 0 & 0 & f_2 & 0 & 0 & f_3 & 0 & 0 & 0 & 0 & 0 \\ 0 & f_1 & 0 & 0 & f_2 & 0 & 0 & f_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & f_1 & 0 & 0 & f_2 & 0 & 0 & f_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & f_1 & 0 & 0 & f_2 & 0 & 0 & f_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & f_1 & 0 & 0 & f_2 & 0 & 0 & f_3 & 0 \\ 0 & 0 & 0 & 0 & 0 & f_1 & 0 & 0 & f_2 & 0 & 0 & f_3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{11} \\ x_{12} \end{pmatrix} = \begin{pmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \end{pmatrix}$$

In case of a column-wise vectorization, we get

$$\mathbf{x} = \text{vec}\left[\begin{pmatrix} x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 \\ x_7 & x_8 & x_9 \\ x_{10} & x_{11} & x_{12} \end{pmatrix}\right] = \begin{pmatrix} x_1 \\ x_4 \\ x_7 \\ x_{10} \\ \vdots \\ x_9 \\ x_{12} \end{pmatrix}$$

$$\mathbf{wx} = \mathbf{s}$$

$$\begin{pmatrix} f_1 & f_2 & f_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & f_1 & f_2 & f_3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & f_1 & f_2 & f_3 & 0 \\ 0 & f_1 & f_2 & f_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & f_1 & f_2 & f_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & f_1 & f_2 & f_3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_4 \\ x_7 \\ x_{10} \\ \vdots \\ x_9 \\ x_{12} \end{pmatrix} = \begin{pmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \end{pmatrix}$$