Introduction to Deep Learning for Speech and Text Processing
Exercise Sheet 6: Neural Networks

Thang Vu

21st November 2025

1 Warm-Up

We might want to build a social media monitoring tool that constantly analyzes data from Instagram and notifies them if a text is about bananas. Here are some example texts:

1) A banana a day keeps the doctor away.
2) I would rather have a sweet chocolate cookie now.
3) Banana! Banana! Banana!

In order to provide the texts as input to a statistical classifier, they need to map each to a fixed-length vector. A very common approach is to represent each word w_i in the input as a unit vector of the size of the vocabulary, where $e_i = 1 \Leftrightarrow w_i$ is the i-th word in the vocabulary. This is called a **one-hot vector** representation. To obtain a representation for the complete input, i.e., the sequence of words in a text, we can simply sum the one-hot vectors for all words. This is often referred to as a **bag-of-words** representation.

Exercise 1.
Given the 4-word vocabulary
$V = \{v_0 : \text{chocolate}, v_1 : \text{sweet}, v_2 : \text{banana}, v_3 : \text{cookie}\}$,
compute the bag-of-words vector representations for the sample texts (lowercase all words in the texts, ignore the punctuation marks and use 0^4 as out-of-vocabulary vector):

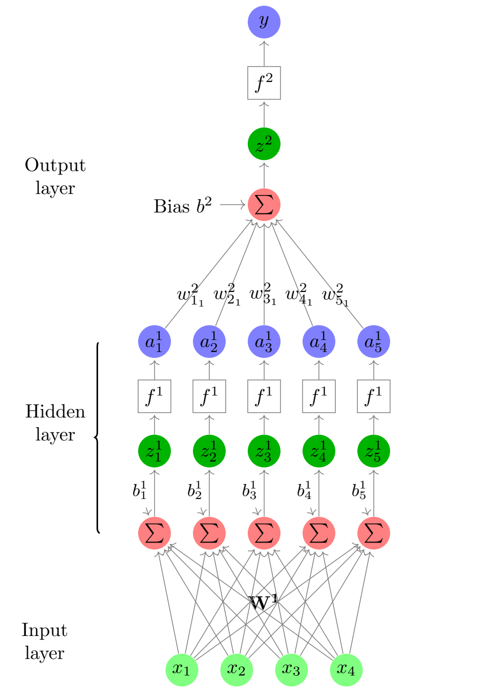
(1) Text 1
(2) Text 2
(3) Text 3

Solution 1.

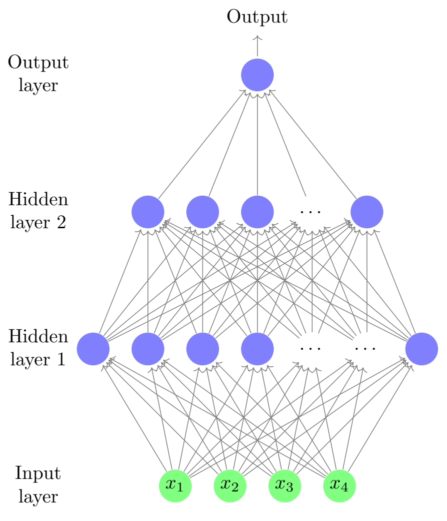
$$x_1 = \begin{bmatrix}0\\0\\0\\0\end{bmatrix} + \begin{bmatrix}0\\0\\1\\0\end{bmatrix} + \begin{bmatrix}0\\0\\0\\0\end{bmatrix} + \begin{bmatrix}0\\0\\0\\0\end{bmatrix} + \begin{bmatrix}0\\0\\0\\0\end{bmatrix} + \begin{bmatrix}0\\0\\0\\0\end{bmatrix} + \begin{bmatrix}0\\0\\0\\0\end{bmatrix} + \begin{bmatrix}0\\0\\0\\0\end{bmatrix} = \begin{bmatrix}0\\0\\1\\0\end{bmatrix}, x_2 = \begin{bmatrix}1\\1\\0\\1\end{bmatrix}, x_3 = \begin{bmatrix}0\\0\\3\\0\end{bmatrix}$$

2 Feed-forward Neural Network

You might come up with the banana network depicted in figure 1b. Note that in the banana network, the hidden layers are displayed conflated. Each hidden layer l consists of first linear activation of the input a^{l-1} with $z^l = W^l a^{l-1} + b^l$ and then non-linear activation, such that $a^l = f^l(z^l)$ as shown in the left figure below.



(a) One-layer feed-forward network for binary classification.



(b) Banana network.

Figure 1: Visualization of a single layer neural network (a) and our banana network (b) consisting of multiple layers.

Exercise 2.

(1) How many parameters does the banana network have in total, if it has 300 units in the first hidden layer and 200 units in the second hidden layer?

(2) Derive the function to compute the output y of this network, given an input vector x.

(3) Compute the output of the network for the first text from Exercise 1. In order to make the computation by hand feasible, use a smaller network with the following parameters:

$$W^1 \in \mathbb{R}^{3\times 4} = \begin{bmatrix} 0 & 1 & -2 & 1 \\ 2 & 3 & 0 & -1 \\ 1 & 0 & -3 & 0 \end{bmatrix}, b^1 \in \mathbb{R}^3 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix},$$

$$W^2 \in \mathbb{R}^{2\times 3} = \begin{bmatrix} 0 & -2 & 1 \\ 2 & 0 & -1 \end{bmatrix}, b^2 \in \mathbb{R}^2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, W^3 \in \mathbb{R}^{1\times 2} = \begin{bmatrix} -1 & 1 \end{bmatrix}, b^3 \in \mathbb{R} = 1$$

For the first two layers, we use the Rectified Linear Unit (ReLU) as activation function, i.e., $f^1(z) = f^2(z) = ReLU(z) = max(0, z)$. The last layer has non-linear activation with the sigmoid, i.e., $f^3(z) = \sigma(z) = \frac{1}{1+e^{-z}}$. Give the resulting network output.

Solution 2.

(1) The parameter set of the network consists of the three weight matrices W^1, W^2 and W^3 and the three biases b^1, b^2, b^3, as each hidden layer and the output layer has a weight matrix and a bias. Given that first hidden layer has 300 units and the input $x \in \mathbb{R}^4$ we obtain $W^1 \in \mathbb{R}^{4\times 300}$. Therefore W^1 has 1200 entries, which are parameters to be learned. The bias in the first layer b^1 has 300 dimensions, i.e. 300 parameters to be learned. Likewise we can see that $W^2 \in \mathbb{R}^{200\times 300}, b^2 \in \mathbb{R}^{200}$ and $W^3 \in \mathbb{R}^{1\times 200}, b^3 \in \mathbb{R}^1$. Therefore the total number of parameters in the network is 1200 + 300 + 60000 + 200 +200 +1 = 61901. 3000 texts are barely enough to train these parameters.

(2)

$$\begin{aligned}
y &= a^3 \\
&= f^3(z^3) \\
&= f^3(W^3 a^2 + b^3) \\
&= f^3(W^3 f^2(z^2) + b^3) \\
&= f^3(W^3 f^2(W^2 a^1 + b^2) + b^3) \\
&= f^3(W^3 f^2(W^2 f^1(z^1) + b^2) + b^3) \\
&= f^3(W^3 f^2(W^2 (f^1(W^1 x + b^1)) + b^2) + b^3)
\end{aligned}$$

(3)

$$\begin{aligned}
y &= f^3(W^3 f^2(W^2 (f^1(W^1 x_a + b^1)) + b^2) + b^3) && (1)\\
&= f^3(W^3 f^2(W^2 f^1(W^1 \begin{bmatrix}0\\0\\1\\0\end{bmatrix}) + b^2)) && (2)\\
&= f^3(W^3 f^2(W^2 f^1 \begin{bmatrix}-2\\0\\-3\end{bmatrix} + b^2)) && (3)\\
&= f^3(W^3 f^2 \begin{bmatrix}1\\1\end{bmatrix}) && (4)\\
&= \sigma(\begin{bmatrix}1\end{bmatrix}) && (5)\\
&= 0.731 && (6)
\end{aligned}$$

banana text detected!

3 Multi-class Classification

Now we want to extend the binary banana-classifier to a multi-class classifier that can detect texts about cookies and bananas at the same time.

Exercise 3.

(1) Extend the network from Exercise 2 to the multi-class case with a single correct class and 3 outputs,

$$y = \begin{bmatrix}\text{p(banana)}\\ \text{p(cookie)}\\ \text{p(other)}\end{bmatrix}.$$

Hint: You only have to change the output layer.

(2) Compute the outputs from the extended network for the first text of Exercise 1 using the bag-of-words representation with vocabulary V, the same weights for W^1, W^2 and bias values b^1, b^2 as in Exercise 2.3 and

$$W^3 = \begin{bmatrix}1 & 1\\ -1 & 2\\ 0 & -1\end{bmatrix},\ b^3 = \begin{bmatrix}0\\0\\0\end{bmatrix}.$$

Solution 3.

(1) In order to perform multiclass classification with 3 output classes, we need three output neurons, i.e. $y \in \mathbb{R}^3$. To achieve this, we have to change the weights in the output layer to a matrix $W^3 \in \mathbb{R}^{3\times 2}$, such that $z^3 \in \mathbb{R}^3$. To ensure that the entries of $y = f(z^3)$ form a probability distribution, i.e. they are all greater than zero and sum to one, we have to change the activation function. The most common choice for the output activation function in neural networks for multiclass classification is the generalization of the logistic sigmoid to multiple inputs, the softmax function. Therefore we set $f^3(z_i^3) = \text{softmax}(z_i^3) = \frac{e^{z_i^3}}{\sum_{j=1}^3 e^{z_j^3}}$.

(2) We can compute the outputs of the network as follows:

$$
\begin{aligned}
y_a &= f^3(W^3 f^2(W^2(f^1(W^1 x_a + b^1)) + b^2) + b^3) && (7)\\
&= f^3(W^3 \begin{bmatrix}1\\1\end{bmatrix}) && (8)\\
&= f^3(\begin{bmatrix}2\\1\\-1\end{bmatrix}) && (9)\\
&= \begin{bmatrix}0.71\\0.26\\0.04\end{bmatrix} && (10)
\end{aligned}
$$

⇒ According to the classifier, text 1 is most likely about bananas.

$$
\begin{aligned}
y_b &= f^3(W^3 f^2(W^2(f^1(W^1 x_b + b^1)) + b^2) + b^3) && (11)\\
&= f^3(W^3 f^2(W^2(f^1(W^1 \begin{bmatrix}1\\1\\0\\1\end{bmatrix} + b^1)) + b^2) + b^3) && (12)\\
&= \text{softmax}(\begin{bmatrix}4\\8\\-4\end{bmatrix}) = \begin{bmatrix}0.02\\0.98\\0.00\end{bmatrix} && (13)
\end{aligned}
$$

⇒ According to the classifier, text 2 is most likely about cookies.

4 Error Computation

Now we want a numerical measure to assess how well their classifier does. We can do this by the means of a cost or error function C that compares the network's predicted probability distribution over N classes $\hat{y} \in \mathbb{R}^N$ with a ground truth $y \in \mathbb{R}^N$ and yields a scalar c, the cost. Here, we use cross-entropy as a common choice for classification:

Cross-Entropy (CE) $c_{\text{CE}} = -\sum_{i=0}^{N-1} y_i \ln \hat{y}_i = -\ln \hat{y}_t$, where t is the correct class

Exercise 4.

(1) Compute the loss for the output of the multiclass classification network for the first text from exercise 3.2 assuming 'banana' as correct class.

(2) For the backward pass, we successively compute the derivatives starting from the last layer. Compute δ^L for the previously computed loss.

Solution 4.

(1) **Cross-entropy loss for the first text.**

From Exercise 3.2, for the first text we obtained the logits

$$z = \begin{bmatrix}2\\1\\-1\end{bmatrix}$$

and the corresponding softmax output

$$\hat{y} = \begin{bmatrix}0.71\\0.26\\0.04\end{bmatrix},$$

where the three entries correspond to the classes banana, cookie, other. The correct class is banana, so the target (one-hot) vector is

$$y = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}.$$

The cross-entropy loss for one-hot targets is

$$c_{\text{CE}} = -\sum_{i=0}^{2} y_i \ln \hat{y}_i = -\ln \hat{y}_t,$$

where t is the index of the correct class. Here, t is the index of the banana class, so

$$c_{\text{CE}} = -\ln \hat{y}_{\text{banana}} = -\ln \hat{y}_0.$$

Using the logits explicitly, the softmax for the banana class is

$$\hat{y}_0 = \frac{e^{z_0}}{\sum_{j=0}^{2} e^{z_j}} = \frac{e^2}{e^2 + e^1 + e^{-1}}.$$

Hence

$$c_{\text{CE}} = -\ln\left(\frac{e^2}{e^2 + e^1 + e^{-1}}\right) \approx 0.35.$$

(2) **Output error signal δ^L (last layer).**

We want the error signal at the last layer,

$$\delta^L = \frac{\partial c_{\text{CE}}}{\partial z^L} \in R^3.$$

Let us denote the logits of the last layer by z_i and the softmax outputs by $\hat{y}_i$ (for $i = 0, 1, 2$).

Step 1: derivative of the cross-entropy w.r.t. $\hat{y}_i$.

The loss is

$$c_{\text{CE}} = -\sum_{i=0}^{2} y_i \ln \hat{y}_i.$$

Differentiating with respect to $\hat{y}_i$:

$$\frac{\partial c_{\text{CE}}}{\partial \hat{y}_i} = \frac{\partial}{\partial \hat{y}_i}\left(-y_i \ln \hat{y}_i\right) = -y_i \frac{1}{\hat{y}_i} = -\frac{y_i}{\hat{y}_i}.$$

Step 2: derivative of softmax w.r.t. logits z_k.

The softmax is defined as

$$\hat{y}_i = \frac{e^{z_i}}{\sum_j e^{z_j}}.$$

Define $S = \sum_j e^{z_j}$, so $\hat{y}_i = \frac{e^{z_i}}{S}$.

We compute $\frac{\partial \hat{y}_i}{\partial z_k}$ and distinguish two cases.

Case 1: $i = k$.

Using the quotient rule,

$$\frac{\partial \hat{y}_k}{\partial z_k} = \frac{e^{z_k} S - e^{z_k} e^{z_k}}{S^2} = \frac{e^{z_k}}{S} \cdot \frac{S - e^{z_k}}{S}.$$

Recognising $\hat{y}_k = \frac{e^{z_k}}{S}$, we obtain

$$\frac{\partial \hat{y}_k}{\partial z_k} = \hat{y}_k(1 - \hat{y}_k).$$

Case 2: $i \neq k$.
Here the numerator e^{z_i} does not depend on z_k, so its derivative is zero:

$$\frac{\partial \hat{y}_i}{\partial z_k} = \frac{0 \cdot S - e^{z_i} \cdot \frac{\partial S}{\partial z_k}}{S^2} = -\frac{e^{z_i} e^{z_k}}{S^2} = -\left(\frac{e^{z_i}}{S}\right)\left(\frac{e^{z_k}}{S}\right) = -\hat{y}_i \hat{y}_k.$$

Summarising,

$$\frac{\partial \hat{y}_i}{\partial z_k} = \begin{cases} \hat{y}_k(1 - \hat{y}_k), & i = k, \\ -\hat{y}_i \hat{y}_k, & i \neq k. \end{cases}$$

Step 3: chain rule for $\frac{\partial c_{\mathbf{CE}}}{\partial z_k}$.
Applying the chain rule:

$$\frac{\partial c_{\text{CE}}}{\partial z_k} = \sum_i \frac{\partial c_{\text{CE}}}{\partial \hat{y}_i} \frac{\partial \hat{y}_i}{\partial z_k}.$$

Split the sum into the term $i = k$ and the terms $i \neq k$:

$$\frac{\partial c_{\text{CE}}}{\partial z_k} = \frac{\partial c_{\text{CE}}}{\partial \hat{y}_k} \frac{\partial \hat{y}_k}{\partial z_k} + \sum_{i \neq k} \frac{\partial c_{\text{CE}}}{\partial \hat{y}_i} \frac{\partial \hat{y}_i}{\partial z_k}.$$

Substitute the expressions from Steps 1 and 2:

$$\begin{aligned} \frac{\partial c_{\text{CE}}}{\partial z_k} &= \left(-\frac{y_k}{\hat{y}_k}\right) \hat{y}_k(1 - \hat{y}_k) + \sum_{i \neq k} \left(-\frac{y_i}{\hat{y}_i}\right)(-\hat{y}_i \hat{y}_k) \\ &= -y_k(1 - \hat{y}_k) + \hat{y}_k \sum_{i \neq k} y_i. \end{aligned}$$

Since y is one-hot encoded, we have $\sum_i y_i = 1$, thus

$$\sum_{i \neq k} y_i = 1 - y_k.$$

Therefore,

$$\begin{aligned} \frac{\partial c_{\text{CE}}}{\partial z_k} &= -y_k(1 - \hat{y}_k) + \hat{y}_k(1 - y_k) \\ &= -y_k + y_k \hat{y}_k + \hat{y}_k - \hat{y}_k y_k \\ &= -y_k + \hat{y}_k. \end{aligned}$$

So we obtain the well-known simplification

$$\boxed{\frac{\partial c_{\text{CE}}}{\partial z_k} = \hat{y}_k - y_k}$$

and, in vector form,

$$\boxed{\delta^L = \hat{y} - y.}$$

Step 4: numerical value of δ^L for the first text.

For the first text we have

$$\hat{y} = \begin{bmatrix} 0.71 \\ 0.26 \\ 0.04 \end{bmatrix}, \quad y = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}.$$

Therefore,

$$\delta^L = \hat{y} - y = \begin{bmatrix} 0.71 - 1 \\ 0.26 - 0 \\ 0.04 - 0 \end{bmatrix} = \boxed{\begin{bmatrix} -0.29 \\ 0.26 \\ 0.04 \end{bmatrix}}.$$

5 Stochastic Gradient Descent

We will now perform the backward pass:

Exercise 5.

(1) For the parameters W^3

  (1) compute the gradient of W^3 with the result being the matrix $\begin{bmatrix} \frac{\partial c}{\partial w^3_{11}} & \frac{\partial c}{\partial w^3_{12}} \\ \frac{\partial c}{\partial w^3_{21}} & \frac{\partial c}{\partial w^3_{22}} \\ \frac{\partial c}{\partial w^3_{31}} & \frac{\partial c}{\partial w^3_{32}} \end{bmatrix}$

  (2) perform a gradient update step using these derivatives with $\eta = 0.1$.

(2) For the parameters W^2

  (1) compute the gradient of W^2 analog to the previous task.

  (2) perform a gradient update step using these derivatives with $\eta = 0.1$.

(3) For the parameters W^1

  (1) compute the gradient of W^1 analog to the previous task.

  (2) perform a gradient update step using these derivatives with $\eta = 0.1$.

Solution 5.

(1) For the parameters W^3

  (1) Compute the gradient of W^3:

$$\nabla_{W^3} c = \begin{bmatrix} \frac{\partial c}{\partial w^3_{11}} & \frac{\partial c}{\partial w^3_{12}} \\ \frac{\partial c}{\partial w^3_{21}} & \frac{\partial c}{\partial w^3_{22}} \\ \frac{\partial c}{\partial w^3_{31}} & \frac{\partial c}{\partial w^3_{32}} \end{bmatrix} = \frac{\partial c}{\partial z^3_i} \frac{\partial z^3_i}{\partial W^3_{ij}} = \boldsymbol{\delta}^3 (\mathbf{a}^2)^\top = \begin{bmatrix} -0.29 \\ 0.26 \\ 0.04 \end{bmatrix} \begin{bmatrix} 1 & 1 \end{bmatrix} = \begin{bmatrix} -0.29 & -0.29 \\ 0.26 & 0.26 \\ 0.04 & 0.04 \end{bmatrix}$$

  (2) Perform a gradient update step with learning rate $\eta = 0.1$:

$$W^3_{\text{new}} = W^3_{\text{old}} - 0.1\,\nabla_{W^3} c = \begin{bmatrix} 1 & 1 \\ -1 & 2 \\ 0 & -1 \end{bmatrix} - 0.1 \begin{bmatrix} -0.29 & -0.29 \\ 0.26 & 0.26 \\ 0.04 & 0.04 \end{bmatrix} = \begin{bmatrix} 1.029 & 1.029 \\ -1.026 & 1.974 \\ -0.004 & -1.004 \end{bmatrix}$$

(2) For the parameters W^2

(1) Compute the gradient analogously:

$$\delta^{(2)} = \mathrm{ReLU}'\big(z^{(2)}\big) \odot \big(W^{(3)}\big)^\top \delta^{(3)} = \begin{bmatrix}1\\1\end{bmatrix} \odot \begin{bmatrix}1 & -1 & 0\\1 & 2 & -1\end{bmatrix} \begin{bmatrix}-0.29\\0.26\\0.04\end{bmatrix} = \begin{bmatrix}-0.55\\0.19\end{bmatrix}$$

$$\nabla_{W^2} c = \boldsymbol{\delta}^2 (\mathbf{a}^1)^\top = \begin{bmatrix}-0.55\\0.19\end{bmatrix} \begin{bmatrix}0 & 0 & 0\end{bmatrix} = \begin{bmatrix}0 & 0 & 0\\0 & 0 & 0\end{bmatrix}$$

(2) Gradient update with $\eta = 0.1$:

$$W^2_{\text{new}} = W^2_{\text{old}} - 0.1\,\nabla_{W^2} c = \begin{bmatrix}0 & -2 & 1\\2 & 0 & -1\end{bmatrix} - 0.1 \begin{bmatrix}0 & 0 & 0\\0 & 0 & 0\end{bmatrix} = \begin{bmatrix}0 & -2 & 1\\2 & 0 & -1\end{bmatrix}$$

(3) For the parameters W^1

(1) Compute the gradient analogously:

$$\delta^{(1)} = \mathrm{ReLU}'\big(z^{(1)}\big) \odot \big(W^{(2)}\big)^\top \delta^{(2)} = \begin{bmatrix}0\\0\\0\end{bmatrix} \odot \begin{bmatrix}0 & 2\\-2 & 0\\1 & -1\end{bmatrix} \begin{bmatrix}-0.55\\0.19\end{bmatrix} = \begin{bmatrix}0\\0\\0\end{bmatrix}$$

$$\nabla_{W^1} c = \boldsymbol{\delta}^1 \mathbf{x}^\top = \begin{bmatrix}0\\0\\0\end{bmatrix} \begin{bmatrix}0 & 0 & 1 & 0\end{bmatrix} = \begin{bmatrix}0 & 0 & 0 & 0\\0 & 0 & 0 & 0\\0 & 0 & 0 & 0\end{bmatrix}$$

(2) Gradient update with $\eta = 0.1$:

$$W^1_{\text{new}} = W^1_{\text{old}} - 0.1\,\nabla_{W^1} c = \begin{bmatrix}0 & 1 & -2 & 1\\2 & 3 & 0 & -1\\1 & 0 & -3 & 0\end{bmatrix} - 0.1 \begin{bmatrix}0 & 0 & 0 & 0\\0 & 0 & 0 & 0\\0 & 0 & 0 & 0\end{bmatrix} = \begin{bmatrix}0 & 1 & -2 & 1\\2 & 3 & 0 & -1\\1 & 0 & -3 & 0\end{bmatrix}$$