Benchmarking Large Language Models and Privacy Protection (OPC Project 2024-25)
Notes on Neural Networks

April 1, 2025

Contents

1 Basics of neural networks

1.1 Introduction

Training Large Language Models requires working with sophisticated neural networks. Therefore, in this document we present a very mild introduction to neural networks, with some simple illustrating examples.

The main goal is as follows: Given the input data X and the output data Y, find the best model that links X to Y. The "best" model is chosen according to a particular loss function. In the context of Large Language Models, X could be for example database of email texts, while Y is a 0-1 variable indicating whether the given text should be treated as spam or not.

1.1.1 Network architecture

A neural network consists of an input layer, several hidden layers, and an output layer. Each layer consists of several neurons. The goal is as follows: given the input X and the output Y, we want to find a set of **weights** W such that the difference

$$Y - XW$$

is as small as possible. Typically, Y is a vector, X is a matrix, and hence W is a matrix of the appropriate dimension.

1.1.2 Notation

- L: Total number of layers (excluding the input layer).
- l: Layer index, where $l = 1, 2, \ldots, L$.
- N_l: Number of neurons in layer l.
- $A^{(l)}$: Activation vector at layer l, of the dimension $N_l \times 1$.
- $Z^{(l)}$: Pre-activation vector at layer l, of the dimension $N_l \times 1$.
- $W^{(l)}$: Weight matrix for layer l, of the dimension $N_l \times N_{l-1}$.
- $b^{(l)}$: Bias vector for layer l, of the dimension $N_l \times 1$.
- $f^{(l)}$: Activation function at layer l, applied element-wise, of the dimension $N_l \times 1$.
- $f'^{(l)}$: Derivative of the activation function at layer l, of the dimension $N_l \times 1$ vector.

- $X = A^{(0)}$: Input; typically a matrix of dimension $N \times p$.
- Y: True output vector.
- $\widehat{Y} = A^{(L)}$: Output vector of the neural network.
- $\mathcal{L}$: Loss function
$$\mathcal{L} = \frac{1}{2}\|A^{(L)} - Y\|^2.$$

1.1.3 Forward Propagation

For each layer $l = 1, 2, \ldots, L$:

1. **Pre-activation:**
$$Z^{(l)} = W^{(l)} A^{(l-1)} + b^{(l)}.$$
2. **Activation:**
$$A^{(l)} = f^{(l)}(Z^{(l)}).$$

1.1.4 Gradient Calculation Using the Chain Rule

The goal is to find the optimal weights. For this, we find the minimum of the loss function. As such, we aim to compute the gradient of the loss function L with respect to the weights $W^{(l)}$ at layer l:

$$\frac{\partial \mathcal{L}}{\partial W^{(l)}} = \left(\frac{\partial \mathcal{L}}{\partial A^{(L)}} \cdot \prod_{k=l+1}^{L} \left(\frac{\partial A^{(k)}}{\partial Z^{(k)}} \cdot \frac{\partial Z^{(k)}}{\partial A^{(k-1)}} \right) \cdot \frac{\partial A^{(l)}}{\partial Z^{(l)}} \right) \cdot \frac{\partial Z^{(l)}}{\partial W^{(l)}}.$$

We will compute each term in this expression, providing both the **matrix-wise** and **element-wise** forms.

- The gradient of the loss function with respect to $A^{(L)}$ is:
$$\frac{\partial \mathcal{L}}{\partial A^{(L)}} = A^{(L)} - Y.$$
Dimension: $N_L \times 1$. The derivatives are taken coordinatewise.
- Since $A^{(k)} = f^{(k)}(Z^{(k)})$,
$$\frac{\partial A^{(k)}}{\partial Z^{(k)}} = \operatorname{diag}\left(f'^{(k)}(Z^{(k)}) \right).$$
Dimension: $N_k \times N_k$.
- Since $Z^{(k)} = W^{(k)} A^{(k-1)} + b^{(k)}$,
$$\frac{\partial Z^{(k)}}{\partial A^{(k-1)}} = W^{(k)}.$$
Dimension: $N_k \times N_{k-1}$.

1.2 Case 1: Neural Network with No Hidden Layer

In this simple example, we will assume our data come from a linear model.

Network Architecture. We consider the neural network consisting of an input layer and an output layer with no hidden layers.

- **Input**: Assume that we have a two-dimensional input (X_{1n}, X_{2n}) for $n = 1, 2, \ldots, N$;
- **Weights**: w_1 and w_2;
- **Bias**: Ignored for simplicity;
- **Activation Function**: Identity function (linear activation).

The **predicted output** for each data point n is:

$$\widehat{Y}_n = w_1 X_{1n} + w_2 X_{2n}.$$

The goal is to find the weights w_1, w_2.

1.2.1 Detailed calculation

- The Mean Squared Error (MSE) loss function:

$$\mathcal{L} = \frac{1}{2N} \sum_{n=1}^{N} \left(\widehat{Y}_n - Y_n\right)^2.$$

- Gradient Calculation:

$$\frac{\partial \mathcal{L}}{\partial w_1} = \frac{1}{N} \sum_{n=1}^{N} \left(\widehat{Y}_n - Y_n\right) X_{1n},$$
$$\frac{\partial \mathcal{L}}{\partial w_2} = \frac{1}{N} \sum_{n=1}^{N} \left(\widehat{Y}_n - Y_n\right) X_{2n}.$$

Alternatively, we can express the gradients in vector form. Let

$$\mathbf{W} = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix},$$

$$\mathbf{Y} = (Y_1, \ldots, Y_N)^\top, \quad \mathbf{X}_n = \begin{bmatrix} X_{1n} \\ X_{2n} \end{bmatrix}, \quad n = 1, \ldots, N$$

and $\mathbf{X}$ be the matrix $N \times 2$ whose n-th row is $\mathbf{X}_n^\top$. Then

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}} = \frac{1}{N}\mathbf{X}^\top(\widehat{\mathbf{Y}} - \mathbf{Y}),$$

where

$$\mathbf{Y} = \mathbf{X}\mathbf{W}.$$

- To find the weights $\mathbf{W}$ that minimize the loss function $\mathcal{L}$, we set the gradient to zero:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}} = 0 \implies \mathbf{X}^\top(\mathbf{X}\mathbf{W} - \mathbf{Y}) = 0.$$

Assuming $\mathbf{X}^\top\mathbf{X}$ is invertible, we can solve for $\mathbf{W}$:

$$\mathbf{W} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{Y}. \tag{1}$$

This solution gives the theoretical optimal weights that minimize the loss function.

Neural network with no hidden layer and linear activation function is equivalent to solving the linear regression problem.

1.3 Case 2: Neural Network with One Hidden Layer

In this section, we explore a neural network architecture comprising of one hidden layer and one neuron. We examine two activation functions for the hidden layer:

1. **Subcase 1**: Linear Activation Function.
2. **Subcase 2**: Sigmoid Activation Function.

In both cases we assume

$$\mathbf{Y} = (Y_1, \ldots, Y_N)^\top, \quad \mathbf{X}_n = \begin{bmatrix} X_{1n} \\ X_{2n} \end{bmatrix}, \quad n = 1, \ldots, N$$

and $\mathbf{X}$ is the matrix $N \times 2$ whose n-th row is $\mathbf{X}_n^\top$.

1.3.1 Subcase 1: Linear Activation Function

Predicted output. For each data point n ($n = 1, 2, \ldots, N$):

$$\begin{aligned} Z_n &= w_1 X_{1n} + w_2 X_{2n}, \\ A_n &= f^{(1)}(z_n) = Z_n = w_1 X_{1n} + w_2 X_{2n}, \\ \widehat{Y}_n &= w_3 A_n = w_3(w_1 X_{1n} + w_2 X_{2n}). \end{aligned}$$

Again, we will ignore the bias.

Detailed calculation.

- The Mean Squared Error (MSE) loss function:

$$\mathcal{L} = \frac{1}{2N} \sum_{n=1}^{N} (\widehat{Y}_n - Y_n)^2.$$

- Gradient Calculation:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial w_1} &= \frac{1}{N} \sum_{n=1}^{N} (\widehat{Y}_n - Y_n) \cdot w_3 X_{1n} = \frac{1}{N} \sum_{n=1}^{N} (w_3(w_1 X_{1n} + w_2 X_{2n}) - Y_n) \cdot w_3 X_{1n}, \\ \frac{\partial \mathcal{L}}{\partial w_2} &= \frac{1}{N} \sum_{n=1}^{N} (\widehat{Y}_n - Y_n) \cdot w_3 X_{2n} = \frac{1}{N} \sum_{n=1}^{N} (w_3(w_1 X_{1n} + w_2 X_{2n}) - Y_n) \cdot w_3 X_{2n}, \\ \frac{\partial \mathcal{L}}{\partial w_3} &= \frac{1}{N} \sum_{n=1}^{N} (\widehat{Y}_n - Y_n) \cdot (w_1 X_{1n} + w_2 X_{2n}) \\ &= \frac{1}{N} \sum_{n=1}^{N} (w_3(w_1 X_{1n} + w_2 X_{2n}) - Y_n) \cdot (w_1 X_{1n} + w_2 X_{2n}). \end{aligned}$$

- For each $w_3 \neq 0$, let

$$\begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = \frac{1}{w_3} \mathbf{W}_{\text{eq}},$$

  where $\mathbf{W}_{\text{eq}}$ are the weights as in (1). Thus, w_1 **and** w_2 **are not uniquely determined.**

1.3.2 Subcase 2: Sigmoid Activation Function

Predicted output. The activation function for the hidden layer is the **sigmoid** function:

$$f^{(1)}(z) = \frac{1}{1 + e^{-z}}.$$

Its derivative is:

$$f'^{(1)}(z) = \frac{1}{1+e^{-z}}\left(1 - \frac{1}{1+e^{-z}}\right)$$

For each data point n ($n = 1, 2, \ldots, N$):

$$\begin{aligned}
Z_n &= w_1 X_{1n} + w_2 X_{2n} \\
A_n &= f^{(1)}(Z_n) = \frac{1}{1+e^{-Z_n}} \\
\widehat{Y}_n &= w_3 A_n = w_3 \cdot \frac{1}{1+e^{-Z_n}} = w_3 \cdot \frac{1}{1+e^{-(w_1 X_{1n} + w_2 X_{2n})}}.
\end{aligned}$$

Detailed calculation.

- Gradients:

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial w_1} &= \frac{1}{N}\sum_{n=1}^{N}(\widehat{Y}_n - Y_n) \cdot w_3 \cdot f'^{(1)}(Z_n) \cdot X_{1n} \\
\frac{\partial \mathcal{L}}{\partial w_2} &= \frac{1}{N}\sum_{n=1}^{N}(\widehat{Y}_n - Y_n) \cdot w_3 \cdot f'^{(1)}(Z_n) \cdot X_{2n} \\
\frac{\partial \mathcal{L}}{\partial w_3} &= \frac{1}{N}\sum_{n=1}^{N}(\widehat{Y}_n - Y_n) \cdot A_n = \frac{1}{N}\sum_{n=1}^{N}(\widehat{Y}_n - Y_n) \cdot f^{(1)}(Z_n).
\end{aligned}$$

- We do not have a closed-form solution. The optimal weights are obtained numerically using the gradient descent.

  To find the theoretical optimal weights, we would need to solve the equations:

$$\frac{\partial \mathcal{L}}{\partial w_1} = 0, \quad \frac{\partial \mathcal{L}}{\partial w_2} = 0, \quad \frac{\partial \mathcal{L}}{\partial w_3} = 0.$$

  That is:

$$\begin{aligned}
0 &= \frac{1}{N}\sum_{n=1}^{N}\left(w_3 \cdot \frac{1}{1+e^{-Z_n}} - Y_n\right) \cdot w_3 \cdot f'^{(1)}(Z_n) \cdot X_{1n} \\
0 &= \frac{1}{N}\sum_{n=1}^{N}\left(w_3 \cdot \frac{1}{1+e^{-Z_n}} - Y_n\right) \cdot w_3 \cdot f'^{(1)}(Z_n) \cdot X_{2n} \\
0 &= \frac{1}{N}\sum_{n=1}^{N}\left(w_3 \cdot \frac{1}{1+e^{-Z_n}} - Y_n\right) \cdot \frac{1}{1+e^{-Z_n}}
\end{aligned}$$

1.4 Example: Gradient computation using batch Gradient Descent with 5 data points

We will demonstrate the gradient computation using batch gradient descent on a subset of 5 data points. We will explicitly write out the derivatives using the chain rule.

Data. We consider the following dataset:

Table 1: Data

x_1	x_2	y
0.5	1.0	1.5
1.5	0.5	2.0
0.0	1.0	1.0
1.0	1.5	2.5
1.0	0.0	1.0

Network architecture and parameters. The neural network has:

- Input Layer: 2 neurons;
- Hidden Layer 1: 3 neurons;
- Hidden Layer 2: 4 neurons;
- Output Layer: 1 neuron;
- Activation Functions: Linear ($f(z) = z$).

Weights and biases are initialized as:

$$W^{(1)} = \begin{bmatrix} 0.2 & -0.1 \\ 0.4 & 0.3 \\ -0.5 & 0.2 \end{bmatrix}, \quad b^{(1)} = \begin{bmatrix} 0.1 \\ -0.2 \\ 0.0 \end{bmatrix},$$

$$W^{(2)} = \begin{bmatrix} 0.3 & -0.2 & 0.1 \\ -0.1 & 0.5 & 0.2 \\ 0.4 & -0.3 & 0.6 \\ -0.2 & 0.1 & -0.4 \end{bmatrix}, \quad b^{(2)} = \begin{bmatrix} 0.0 \\ 0.1 \\ -0.1 \\ 0.2 \end{bmatrix},$$

$$W^{(3)} = \begin{bmatrix} 0.2 & -0.3 & 0.5 & -0.1 \end{bmatrix}, \quad b^{(3)} = 0.05.$$

Forward Propagation. We perform forward propagation.

- Input layer ($A^{(0)}$):

$$A^{(0)} = \begin{bmatrix} 0.5 & 1.5 & 0.0 & 1.0 & 1.0 \\ 1.0 & 0.5 & 1.0 & 1.5 & 0.0 \end{bmatrix}$$

- First hidden layer We compute $Z^{(1)} = W^{(1)}A^{(0)} + b^{(1)}$:

$$Z^{(1)} = \begin{bmatrix} 0.2 & -0.1 \\ 0.4 & 0.3 \\ -0.5 & 0.2 \end{bmatrix} \begin{bmatrix} 0.5 & 1.5 & 0.0 & 1.0 & 1.0 \\ 1.0 & 0.5 & 1.0 & 1.5 & 0.0 \end{bmatrix} + \begin{bmatrix} 0.1 \\ -0.2 \\ 0.0 \end{bmatrix}.$$

  Thus

$$Z^{(1)} = \begin{bmatrix} 0.1 & 0.2 & -0.05 & 0.3 & 0.2 \\ 0.3 & 0.75 & 0.0 & 0.55 & 0.3 \\ -0.05 & 0.1 & 0.25 & -0.2 & -0.5 \end{bmatrix}.$$

  Since the activation function is linear, $A^{(1)} = Z^{(1)}$.

- Second hidden layer. We compute $Z^{(2)} = W^{(2)}A^{(1)} + b^{(2)}$:

$$Z^{(2)} = \begin{bmatrix} 0.3 & -0.2 & 0.1 \\ -0.1 & 0.5 & 0.2 \\ 0.4 & -0.3 & 0.6 \\ -0.2 & 0.1 & -0.4 \end{bmatrix} \begin{bmatrix} 0.1 & 0.2 & -0.05 & 0.3 & 0.2 \\ 0.3 & 0.75 & 0.0 & 0.55 & 0.3 \\ -0.05 & 0.1 & 0.25 & -0.2 & -0.5 \end{bmatrix} + \begin{bmatrix} 0.0 \\ 0.1 \\ -0.1 \\ 0.2 \end{bmatrix}.$$

  Thus

$$Z^{(2)} = \begin{bmatrix} -0.035 & 0.015 & -0.025 & 0.015 & 0.05 \\ 0.23 & 0.395 & 0.05 & 0.365 & 0.11 \\ -0.18 & 0.12 & 0.05 & 0.14 & 0.38 \\ 0.23 & 0.095 & -0.05 & 0.07 & 0.05 \end{bmatrix}.$$

  Since the activation function is linear, $A^{(2)} = Z^{(2)}$.

- Output layer. We compute $Z^{(3)} = W^{(3)}A^{(2)} + b^{(3)}$:

$$Z^{(3)} = \begin{bmatrix} 0.2 & -0.3 & 0.5 & -0.1 \end{bmatrix} \begin{bmatrix} -0.035 & 0.015 & -0.025 & 0.015 & 0.05 \\ 0.23 & 0.395 & 0.05 & 0.365 & 0.11 \\ -0.18 & 0.12 & 0.05 & 0.14 & 0.38 \\ 0.23 & 0.095 & -0.05 & 0.07 & 0.05 \end{bmatrix} + 0.05.$$

  Thus

$$Z^{(3)} = \begin{bmatrix} -0.139 & -0.105 & 0.0 & 0.05 & 0.0 \end{bmatrix}.$$

  Since the activation function is linear, $A^{(3)} = Z^{(3)}$.

  Hence,

$$A^{(3)} - Y = \begin{bmatrix} -1.639 & -2.105 & -1.0 & -2.45 & -1.0 \end{bmatrix}.$$

Gradient computation using the chain rule. We will compute the gradients of the total loss $\mathcal{L}$ with respect to the weights $W^{(3)}, W^{(2)}$, and $W^{(1)}$ using the chain rule.

- Gradient: $\frac{\partial \mathcal{L}}{\partial W^{(3)}}$:

$$\frac{\partial \mathcal{L}}{\partial W^{(3)}} = \frac{\partial \mathcal{L}}{\partial Z^{(3)}} \cdot \frac{\partial Z^{(3)}}{\partial W^{(3)}} = \frac{1}{5}(A^{(3)} - Y)(A^{(2)})^T.$$

$$A^{(2)} = \begin{bmatrix} -0.035 & 0.015 & -0.025 & 0.015 & 0.05 \\ 0.23 & 0.395 & 0.05 & 0.365 & 0.11 \\ -0.18 & 0.12 & 0.05 & 0.14 & 0.38 \\ 0.23 & 0.095 & -0.05 & 0.07 & 0.05 \end{bmatrix},$$

$$\frac{\partial \mathcal{L}}{\partial W^{(3)}} = \begin{bmatrix} -0.162057 & -0.241662 & 0.001645 & -0.220982 \end{bmatrix}.$$

- Gradient: $\frac{\partial \mathcal{L}}{\partial W^{(2)}}$:

$$\frac{\partial \mathcal{L}}{\partial W^{(2)}} = \frac{\partial \mathcal{L}}{\partial Z^{(3)}} \cdot \frac{\partial Z^{(3)}}{\partial A^{(3)}} \cdot \frac{\partial A^{(3)}}{\partial Z^{(2)}} \cdot \frac{\partial Z^{(2)}}{\partial W^{(2)}}.$$

We have:

$$\frac{\partial \mathcal{L}}{\partial Z^{(3)}} = \frac{\partial \mathcal{L}}{\partial A^{(3)}} = \frac{1}{5}(A^{(3)} - Y) = \frac{1}{5}\begin{bmatrix} -1.639 & -2.105 & -1.0 & -2.45 & -1.0 \end{bmatrix},$$

$$\frac{\partial Z^{(3)}}{\partial A^{(3)}} = (W^{(3)})^T,$$

$$\frac{\partial A^{(3)}}{\partial Z^{(2)}} = \frac{\partial A^{(3)}}{\partial Z^{(3)}} \cdot \frac{\partial Z^{(3)}}{\partial A^{(2)}} \cdot \frac{\partial A^{(2)}}{\partial Z^{(2)}} = (W^{(3)})^T,$$

$$\frac{\partial Z^{(2)}}{\partial W^{(2)}} = (W^{(1)})^T,$$

$$\frac{\partial \mathcal{L}}{\partial W^{(2)}} = \frac{1}{N}(A^{(3)} - Y) \cdot (W^{(3)})^T \cdot (W^{(1)})^T.$$

We have

$$A^{(3)} - Y = \begin{bmatrix} -1.639 & -2.105 & -1.0 & -2.45 & -1.0 \end{bmatrix},$$

- Gradient: $\frac{\partial \mathcal{L}}{\partial W^{(1)}}$:

$$\frac{\partial \mathcal{L}}{\partial W^{(1)}} = \frac{\partial \mathcal{L}}{\partial Z^{(1)}} \cdot \frac{\partial Z^{(1)}}{\partial W^{(1)}} = \frac{1}{5}(W^{(2)})^T \frac{\partial \mathcal{L}}{\partial Z^{(2)}}(A^{(0)})^T.$$

Updating the Weights. Choose a learning rate η. For demonstration, let $\eta = 0.01$. Then

$$W^{(3)}_{\text{new}} = W^{(3)} - \eta \frac{\partial \mathcal{L}}{\partial W^{(3)}}$$
$$= \begin{bmatrix} 0.2 & -0.3 & 0.5 & -0.1 \end{bmatrix} - 0.01 \times \begin{bmatrix} -0.162057 & -0.241662 & 0.001645 & -0.220982 \end{bmatrix}.$$

Similarly, update $W^{(2)}_{\text{new}}$ and $W^{(1)}_{\text{new}}$ using the computed gradients:

$$W^{(2)}_{\text{new}} = W^{(2)} - \eta \frac{\partial \mathcal{L}}{\partial W^{(2)}},$$

and

$$W^{(1)}_{\text{new}} = W^{(1)} - \eta \frac{\partial \mathcal{L}}{\partial W^{(1)}}.$$