Homework 1: Introduction to Deep Learning

Daryoush Seif Ashrafy

October 2025

1 Theoretical Problems

Problem 1. Optimizers

(a) **Momentum** represents an exponentially weighted sum of previous gradients.
It acts like a "moving object with momentum," intending to continue its motion along the path where it has already gained speed—i.e., in the direction of previous gradients (previous updates).
Using momentum, our aim is to avoid oscillations in which the direction constantly flips signs, instead moving in a more stable and consistent direction.
Formulation:

$$\Delta W^{(k)} = \beta \Delta W^{(k-1)} + \eta \nabla_W (W^{(k-1)})$$
$$W^{(k)} = W^{(k-1)} - \Delta W^{(k)}$$

(b) Both Nesterov Accelerated Gradient (NAG / Nesterov momentum) and the momentum involve two main components:
The current gradient (which indicates the direction towards the optimal point based on the current position) and the previous gradients (which represent the accumulated experience of moving toward the optimum in previous steps).
In the momentum method, the current gradient is first calculated (showing the direction toward the optimal point from the current position) and then combined — in a weighted manner — with the sum of previous gradients.
However, the NAG method works in a look-ahead manner. This means that it first moves in the direction of the accumulated previous gradients and then, based on the new "look-ahead" position, computes the gradient to move toward the optimal point.
Momentum formula:

$$\Delta W^{(k)} = \beta \Delta W^{(k-1)} + \eta \nabla_W (W^{(k-1)})$$

NAG formula:

$$\Delta W^{(k)} = \beta \Delta W^{(k-1)} + \eta \nabla_W (W^{(k-1)} + \beta \Delta W^{(k-1)})$$

(c) In adaptive learning rate methods, we assume that the direction in which we are oscillating has larger gradient magnitudes, meaning the movement in that direction is significantly larger than in directions where the movement is stable.

The core idea of **Adagrad** to address this issue is to normalize the update step by dividing the movement update by the square root of the sum of squared previous gradients (past updates). This reduces the intensity of movement in directions with oscillations, leading to a more stable trajectory toward the optimal point.

Formula:

$$s^{(k)} = \beta s^{(k-1)} + (1-\beta)(\partial_w(E))^{2^{(k)}}$$

$$w^{(k+1)} = w^{(k)} - \frac{\eta}{\sqrt{s^{(k)} + \epsilon}}(\partial_w(E))^{2^{(k)}}$$

Main Issue: The term $s^{(k)}$ (sum of squared gradients) is always positive and monotonically increasing with each update. As a result, with each step, $s^{(k)}$ grows larger, causing the learning rate to shrink progressively. This can hinder convergence to the optimal point after several steps, as the updates become too small.

(d) **RMSProp** works similarly to **AdaGrad**, but solves its key limitation.
AdaGrad Issue: Uses the cumulative sum of squared gradients, which grows unbounded over time, causing the learning rate to shrink to nearly zero.
RMSProp Solution: Instead of the total sum, it uses the exponentially weighted moving average of squared gradients. This prevents unbounded growth and keeps the learning rate adaptive but stable.
Formula:

$$v^{(k)} = \beta v^{(k-1)} + (1-\beta)(\partial_w(E))^{2^{(k)}}$$

$$w^{(k+1)} = w^{(k)} - \frac{\eta}{\sqrt{v^{(k)} + \epsilon}}(\partial_w(E))^{2^{(k)}}$$

Where $v^{(k)} = \overline{(\partial_w E)^{2^{(k)}}}$

(e) **Adam** is the result of combining bias-corrected first and second momentum (RMSProp):

(a) Bias-corrected first moment (exponentially weighted average of gradients, i.e. Momentum)

(b) Bias-corrected second moment (exponentially weighted average of squared gradients, i.e. RMSProp)

First momentum:

$$m^{(k)} = \beta_1 m^{(k)} + (1-\beta_1)(\partial_w E)^{(k)}$$

RMSProb:

$$v^{(k)} = \beta_2 v^{(k)} + (1-\beta_2)(\partial_w E)^{2^{(k)}}$$

Bias correction:

$$\hat{m}^{(k)} = \frac{m^{(k)}}{1-\beta_1}, \quad \hat{v}^{(k)} = \frac{v^{(k)}}{1-\beta_2}$$

Adam formula:

$$w^{(k+1)} = w^{(k)} - \frac{\eta}{\sqrt{\hat{v}^{(k)}+\epsilon}}\hat{m}^{(k)}$$

Why Bias Correction? Early in training, moments $m^{(k)}$ and $v^{(k)}$ are biased toward zero due to exponential weighting. Bias correction normalizes them for more accurate adaptation from the start.

Problem 2. Kaiming He

(a) **Zero mean**

$$\mathbb{E}[z_i^{(h)}] = 0$$

Proof. We begin with the definition:

$$z^{(h)} = W^{(h)}x^{(h)} \quad \Rightarrow \quad z_i^{(h)} = \sum_j W_{ij}^{(h)} x_j^{(h)}$$

Taking the expectation:

$$\mathbb{E}[z_i^{(h)}] = \mathbb{E}\left[\sum_j W_{ij}^{(h)} x_j^{(h)}\right] = \sum_j \mathbb{E}[W_{ij}^{(h)} x_j^{(h)}]$$

Using the identity for the expectation of a product:

$$\mathbb{E}[W_{ij}^{(h)} x_j^{(h)}] = \mathbb{E}[W_{ij}^{(h)}] \cdot \mathbb{E}[x_j^{(h)}] + \mathrm{Cov}(W_{ij}^{(h)}, x_j^{(h)})$$

Now we analyze each term:

(i) **Independence:** Since weights are initialized independently of the activations from previous layers:

$$\mathrm{Cov}(W_{ij}^{(h)}, x_j^{(h)}) = 0$$

(ii) **Zero-mean weights:** By initialization assumption:

$$\mathbb{E}[W_{ij}^{(h)}] = 0 \quad \Rightarrow \quad \mathbb{E}[W_{ij}^{(h)}] \cdot \mathbb{E}[x_j^{(h)}] = 0$$

Therefore:

$$\mathbb{E}[W_{ij}^{(h)} x_j^{(h)}] = 0 + 0 = 0$$

And consequently:

$$\mathbb{E}[z_i^{(h)}] = \sum_j 0 = 0$$

□

(b) **Variance expansion**

$$Var(z_i^{(h)}) = d_h \sigma^2 \mathbb{E}[(x_j^{(h)})^2]$$

Proof. We begin with the definition:

$$z^{(h)} = W^{(h)} x^{(h)} \quad \Rightarrow \quad z_i^{(h)} = \sum_j W_{ij}^{(h)} x_j^{(h)}$$

Taking the variance:

$$\text{Var}(z_i^{(h)}) = \text{Var}\left(\sum_j W_{ij}^{(h)} x_j^{(h)} \right)$$

Using the variance sum formula:

$$\text{Var}(a + b) = \text{Var}(a) + \text{Var}(b) + 2\text{Cov}(a, b)$$

By the independence assumption across index j, the covariance terms vanish, giving:

$$\text{Var}\left(\sum_j W_{ij}^{(h)} x_j^{(h)} \right) = \sum_j \text{Var}(W_{ij}^{(h)} x_j^{(h)})$$

Now compute each term using the variance definition:

$$\text{Var}(W_{ij}^{(h)} x_j^{(h)}) = \mathbb{E}[(W_{ij}^{(h)} x_j^{(h)})^2] - \mathbb{E}[W_{ij}^{(h)} x_j^{(h)}]^2$$

From part (a), we have $\mathbb{E}[W_{ij}^{(h)} x_j^{(h)}] = 0$, so:

$$\text{Var}(W_{ij}^{(h)} x_j^{(h)}) = \mathbb{E}[(W_{ij}^{(h)} x_j^{(h)})^2] = \mathbb{E}[W_{ij}^{(h)2} \cdot x_j^{(h)2}]$$

Expanding the expectation:

$$\mathbb{E}[W_{ij}^{(h)2} \cdot x_j^{(h)2}] = \mathbb{E}[W_{ij}^{(h)2}] \cdot \mathbb{E}[x_j^{(h)2}] + \text{Cov}(W_{ij}^{(h)2}, x_j^{(h)2})$$

By independence, $\text{Cov}(W_{ij}^{(h)2}, x_j^{(h)2}) = 0$, leaving:

$$\mathbb{E}[W_{ij}^{(h)2} \cdot x_j^{(h)2}] = \mathbb{E}[W_{ij}^{(h)2}] \cdot \mathbb{E}[x_j^{(h)2}] = \sigma_h^2 \cdot \mathbb{E}[x_j^{(h)2}]$$

Substituting back:

$$\text{Var}(z_i^{(h)}) = \sum_j \text{Var}(W_{ij}^{(h)} x_j^{(h)}) = \sum_j \sigma_h^2 \cdot \mathbb{E}[x_j^{(h)2}]$$

Since the distribution of $x_j^{(h)}$ is identical for all j, we have:

$$\text{Var}(z_i^{(h)}) = h_d \cdot \sigma_h^2 \cdot \mathbb{E}[x_j^{(h)2}]$$

□

(c) **ReLU second moment**

$$\mathbb{E}[(x_j^{(h)})^2] = \frac{1}{2}\text{Var}(z_j^{(h-1)})$$

Proof. We know that:

$$x_j^{(h)} = \text{ReLU}(z_j^{(h-1)}) = \max(0, z_j^{(h-1)})$$

Taking the expectation of the square:

$$\mathbb{E}[(x_j^{(h)})^2] = \mathbb{E}[(\max(0, z_j^{(h-1)}))^2]$$

Recall the definition of expectation for continuous random variables:

$$\mathbb{E}[g(a)] = \int_{-\infty}^{\infty} g(a) f(a) da$$

where $f(a)$ is the probability density function.
Applying this to our case:

$$\mathbb{E}[(\max(0, z_j^{(h-1)}))^2] = \int_{-\infty}^{\infty} (\max(0, z))^2 p(z) dz = \int_0^{\infty} z^2 p(z) dz$$

Now consider the variance of $z_j^{(h-1)}$:

$$\text{Var}(z_j^{(h-1)}) = \mathbb{E}[(z_j^{(h-1)})^2] - \mathbb{E}[z_j^{(h-1)}]^2$$

From part (a), we have $\mathbb{E}[z_j^{(h-1)}] = 0$, so:

$$\text{Var}(z_j^{(h-1)}) = \mathbb{E}[(z_j^{(h-1)})^2] = \int_{-\infty}^{\infty} z^2 p(z) dz$$

Assuming the distribution of $z_j^{(h-1)}$ is symmetric about zero, we have:

$$\int_{-\infty}^{\infty} z^2 p(z) dz = 2 \int_{0}^{\infty} z^2 p(z) dz$$

Therefore:

$$\mathbb{E}[(x_j^{(h)})^2] = \int_{0}^{\infty} z^2 p(z) dz = \frac{1}{2} \int_{-\infty}^{\infty} z^2 p(z) dz = \frac{1}{2} \text{Var}(z_j^{(h-1)})$$

Thus:

$$\mathbb{E}[(x_j^{(h)})^2] = \frac{1}{2} \text{Var}(z_j^{(h-1)})$$

□

(d) **Combining Results**

From part (b), we obtained:

$$\text{Var}(z_i^{(h)}) = h_d \cdot \sigma_h^2 \cdot \mathbb{E}[(x_j^{(h)})^2]$$

From part (c), we derived:

$$\mathbb{E}[(x_j^{(h)})^2] = \frac{1}{2} \text{Var}(z_j^{(h-1)})$$

Substituting the result from (c) into (b):

$$\text{Var}(z_i^{(h)}) = h_d \cdot \sigma_h^2 \cdot \frac{1}{2} \text{Var}(z_j^{(h-1)})$$

To preserve the variance of activations across layers (maintain stable signal propagation), we require:

$$\text{Var}(z_j^{(h-1)}) = \text{Var}(z_i^{(h)})$$

Substituting this equality into the previous equation:

$$\text{Var}(z_i^{(h)}) = h_d \cdot \sigma_h^2 \cdot \frac{1}{2} \text{Var}(z_i^{(h)})$$

Assuming $\text{Var}(z_i^{(h)}) \neq 0$, we can divide both sides by $\text{Var}(z_i^{(h)})$:

$$1 = h_d \cdot \sigma_h^2 \cdot \frac{1}{2}$$

Solving for σ_h^2:

$$\sigma_h^2 = \frac{2}{h_d}$$

Problem 3. War Zone

We can apply the **Universal Approximation Theorem** in both parts and solve the problem using only a single hidden layer. However, since there is no limitation on the number of layers, we can design a deeper network to make the representation more structured and modular.

(a) Our non-convex shape is composed of two convex subregions — the upper triangle and the lower square.
In the first layer, we model each convex region separately by using neurons that act as linear separators to capture their boundaries.
Then, in the second layer, we combine the outputs of these two neurons using an OR gate–like neuron, which merges the two convex regions into a single non-convex output region.

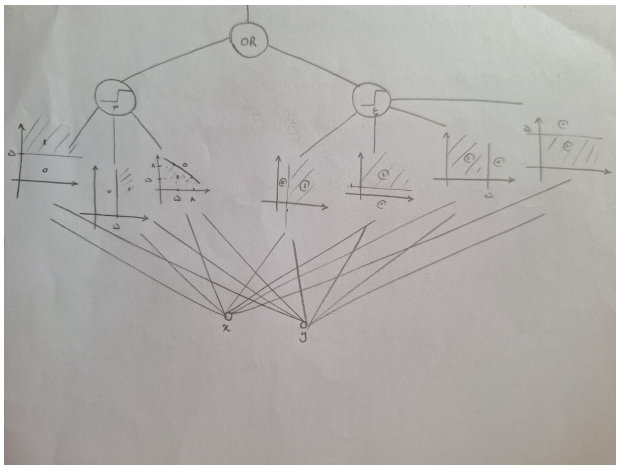

Figure 1: 3-a

(b) Since the surface in this case is spherical, using linear separators is no longer suitable, as a sphere cannot be represented by a single linear boundary. To handle this, we can map the inputs into a two-dimensional (or higher) feature space, where the circular or spherical boundary can be represented more easily. After this transformation, the target region — the area between two circles — can be expressed as the intersection of two conditions:

(i) inside the outer circle

(ii) outside the inner circle

Therefore, the final layer can combine these two conditions using an AND gate–like neuron to represent the desired region.

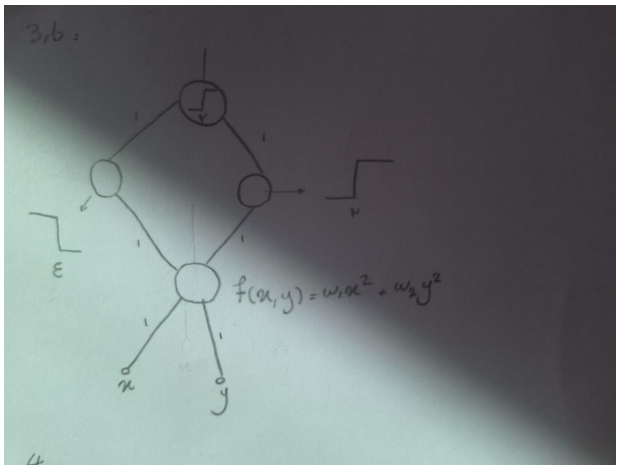

Figure 2: 3-b

Problem 4. Divergence

(a) *Proof.* Second-order Taylor expansion of $E(w)$ around $w = w_k$:

$$E(w) \approx E(w^{(k)}) + E'(w^{(k)})(w - w^{(k)}) + \frac{1}{2}E''(w^{(k)})(w - w^{(k)})^2$$

Key Property: Exact equality holds if $E(w)$ is quadratic.

Optimality Condition:
To find $\hat{w} = \arg\min_w E(w)$, set the gradient of the approximation to zero:

$$\frac{\partial E(w)}{\partial w} = E'(w^{(k)}) + E''(w^{(k)})(w - w^{(k)}) = 0$$

Differentiate each term

(i) Derivative of $E(w^{(k)})$: 0 (it's a constant)
(ii) Derivative of $E'(w^{(k)})(w - w^{(k)})$: $E'(w^{(k)})$
(iii) Derivative of $\frac{1}{2}E''(w^{(k)})(w - w^{(k)})^2$: $E''(w^{(k)})(w - w^{(k)})$

Multiply through by the inverse Hessian $E''(w^{(k)})^{-1}$:

$$E''(w^{(k)})^{-1}E'(w^{(k)}) + w - w^{(k)} = 0$$

Rearrange to obtain the **Newton update rule**:

$$w = w^{(k)} - E''(w^{(k)})^{-1}E'(w^{(k)})$$

Thus:

$$\eta_{\text{opt}} = E''(w^{(k)})^{-1}$$

□

(b) The derived relationship holds specifically for quadratic loss functions; for general non-quadratic objectives, this equation is not necessarily valid. Consequently, the second-order derivative no longer directly determines the optimal learning rate.

Consider, for example, the non-quadratic loss function:

$$E(w) = e^w + \sin(w)$$

Its first-order derivative:

$$E'(w) = e^w + \cos(w)$$

And its second-order derivative:

$$E''(w) = e^w - \sin(w)$$

(c) *Proof.* We know that the optimal learning rate is the step size that moves us exactly to the optimal point in one iteration. Therefore, the distance to the optimum is equal to one optimal learning rate step.
Now, suppose we move one step to reach the optimum, and then continue moving by the same amount again.
In that case, we will have moved twice the optimal learning rate, which means we are now back at the same distance from the optimum as our initial point, but on the opposite side.
If we go even further than twice the optimal learning rate, our distance from the starting point will increase — hence, the process will diverge when the step size exceeds two times the optimal learning rate. □

(d) *Proof.* Similarly, if the learning rate is between the optimal value and twice the optimal learning rate, we will first overshoot the optimal point, moving slightly beyond it.
However, since the step size is less than twice the optimal rate, our distance from the starting point will be smaller than before.
As a result, in each iteration, we move closer to the optimum, but still cross over it slightly, causing the updates to oscillate around the optimal point. □

Problem 5. Backpropagation

$$\begin{aligned} s_1 &= w_1x_1 + w_2x_2, & s_2 &= w_3x_3 + w_4x_4 \\ h_1 &= \sigma(s_1), & h_2 &= \sigma(s_2) \\ s_3 &= w_5h_1 + w_6h_2 \\ \hat{y} &= \sigma(s_3) \end{aligned}$$

(a) **Forward Pass**

Given inputs: $x_1 = 1.7$, $x_2 = 0.1$, $x_3 = -0.6$, $x_4 = -1.8$
Weights: $w_1 = -0.7$, $w_2 = 1.2$, $w_3 = 1.1$, $w_4 = -2.0$

$$
\begin{aligned}
s_1 &= (-0.7)(-1.7) + (1.2)(0.1) = 1.19 + 0.12 = 1.31 \\
s_2 &= (1.1)(-0.6) + (-2.0)(-1.8) = -0.66 + 3.6 = 2.94 \\
h_1 &= \sigma(1.31) \approx 0.79 \\
h_2 &= \sigma(2.94) \approx 0.95 \\
s_3 &= w_5 h_1 + w_6 h_2 = (-0.2)(0.79) + (0.5)(0.95) = 0.317 \\
\hat{y} &= \sigma(s_3) \approx 0.58
\end{aligned}
$$

(b) **Backward Pass**

Loss function: $L(y, \hat{y}) = (y - \hat{y})^2$

$$L = (0.5 - 0.85)^2 = (-0.35)^2 = 0.1225$$

Chain Rule Derivatives:

$$
\begin{aligned}
\frac{\partial L}{\partial w_5} &= \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial s_3} \cdot \frac{\partial s_3}{\partial w_5} \\
\frac{\partial L}{\partial w_6} &= \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial s_3} \cdot \frac{\partial s_3}{\partial w_6} \\
\frac{\partial L}{\partial w_1} &= \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial s_3} \cdot \frac{\partial s_3}{\partial h_1} \cdot \frac{\partial h_1}{\partial s_1} \cdot \frac{\partial s_1}{\partial w_1} \\
\frac{\partial L}{\partial w_2} &= \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial s_3} \cdot \frac{\partial s_3}{\partial h_1} \cdot \frac{\partial h_1}{\partial s_1} \cdot \frac{\partial s_1}{\partial w_2} \\
\frac{\partial L}{\partial w_3} &= \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial s_3} \cdot \frac{\partial s_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial s_2} \cdot \frac{\partial s_2}{\partial w_3} \\
\frac{\partial L}{\partial w_4} &= \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial s_3} \cdot \frac{\partial s_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial s_2} \cdot \frac{\partial s_2}{\partial w_4}
\end{aligned}
$$

1. Output layer:

$$\frac{\partial L}{\partial \hat{y}} = -2(y - \hat{y}) = -2(0.5 - 0.58) = -0.16$$

$$\frac{\partial \hat{y}}{\partial s_3} = \hat{y}(1 - \hat{y}) \approx 0.58 \times 0.42 = 0.2436$$

Let

$$\delta_3 = \frac{\partial L}{\partial s_3} = \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial s_3} \approx -0.16 \times 0.2436 = 0.038976$$

2. Weights w_5, w_6:

$$\frac{\partial L}{\partial w_5} = \delta_3 \cdot h_1 \approx 0.038976 \times 0.79 \approx 0.03079104$$

$$\frac{\partial L}{\partial w_6} = \delta_3 \cdot h_2 \approx 0.038976 \times 0.95 \approx 0.0370272$$

3. Hidden layer errors:

$$\delta_1 = \delta_3 \cdot w_5 \cdot h_1(1-h_1) \approx 0.038976 \times (-0.2) \times (0.79 \times 0.21) \approx 0.00129322368$$

$$\delta_2 = \delta_3 \cdot w_6 \cdot h_2(1-h_2) \approx 0.038976 \times 0.5 \times (0.95 \times 0.05) \approx 0.00092568$$

4. Weights w_1, w_2, w_3, w_4:

$$\frac{\partial L}{\partial w_1} = \delta_1 \cdot x_1 = 0.00129322368 \times (-0.7)$$

$$\frac{\partial L}{\partial w_2} = \delta_1 \cdot x_2 = 0.00129322368 \times 1.2$$

$$\frac{\partial L}{\partial w_3} = \delta_2 \cdot x_3 = 0.00092568 \times 1.1$$

$$\frac{\partial L}{\partial w_4} = \delta_2 \cdot x_4 = 0.00092568 \times (-2)$$

Problem 6. Dropout

(a) Show that

$$\mathbb{E}[R_{ni}R_{nj}] = \delta_{ij}\rho + (1-\delta_{ij})\rho^2$$

Proof. We begin with the expectation of the product:

$$\mathbb{E}[R_{ni}R_{nj}] = \mathbb{E}[R_{ni}]\mathbb{E}[R_{nj}] + \mathrm{Cov}(R_{ni}, R_{nj}) = \rho^2 + \mathrm{Cov}(R_{ni}, R_{nj})$$

Now consider the covariance in two cases:

(i) For $i \neq j$:

$$\mathrm{Cov}(R_{ni}, R_{nj}) = 0$$

(ii) For $i = j$:

$$\mathrm{Cov}(R_{ni}, R_{nj}) = \mathrm{Var}(R_{ni}) = \mathbb{E}[R_{ni}^2] - \mathbb{E}[R_{ni}]^2 = \rho - \rho^2$$

Define the Kronecker delta:

$$\delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

Combining both cases:

$$\mathbb{E}[R_{ni}R_{nj}] = \delta_{ij}\rho + (1-\delta_{ij})\rho^2$$

□

(b) Show that the following is the expectation of the error function defined above:

$$\mathbb{E}[E(\mathbf{W})] = \sum_{n=1}^{N}\sum_{k=1}^{K}\left\{y_{nk} - \rho\sum_{i=1}^{D} w_{ki}x_{ni}\right\}^2 + \rho(1-\rho)\sum_{n=1}^{N}\sum_{k=1}^{K}\sum_{i=1}^{D} w_{ki}^2 x_{ni}^2$$

Proof. We begin with the objective function:

$$E(\mathbf{W}) = \sum_{n=1}^{N}\sum_{k=1}^{K}\left\{y_{nk} - \sum_{i=1}^{D} w_{ki}R_{ni}x_{ni}\right\}^2$$

Taking the expectation:

$$\mathbb{E}[E(\mathbf{W})] = \mathbb{E}\left[\sum_{n=1}^{N}\sum_{k=1}^{K}\left(y_{nk} - \sum_{i=1}^{D} w_{ki}R_{ni}x_{ni}\right)^2\right]$$

Using the identity $\mathbb{E}[Z^2] = \mathrm{Var}(Z) + \mathbb{E}[Z]^2$ for each term:

$$\mathbb{E}[E(\mathbf{W})] = \sum_{n=1}^{N}\sum_{k=1}^{K}\left\{\mathrm{Var}\left(y_{nk} - \sum_{i=1}^{D} w_{ki}R_{ni}x_{ni}\right) + \mathbb{E}\left[y_{nk} - \sum_{i=1}^{D} w_{ki}R_{ni}x_{ni}\right]^2\right\}$$

Expectation term:

$$\mathbb{E}\left[y_{nk} - \sum_{i=1}^{D} w_{ki}R_{ni}x_{ni}\right] = y_{nk} - \sum_{i=1}^{D} w_{ki}\mathbb{E}[R_{ni}]x_{ni}$$

Since $\mathbb{E}[R_{ni}] = \rho$, we have:

$$\mathbb{E}\left[y_{nk} - \sum_{i=1}^{D} w_{ki}R_{ni}x_{ni}\right]^2 = \left(y_{nk} - \rho\sum_{i=1}^{D} w_{ki}x_{ni}\right)^2$$

Variance term: Since y_{nk} is constant:

$$\mathrm{Var}\left(y_{nk} - \sum_{i=1}^{D} w_{ki}R_{ni}x_{ni}\right) = \mathrm{Var}\left(\sum_{i=1}^{D} w_{ki}R_{ni}x_{ni}\right)$$

For Bernoulli random variables R_{ni} with $\mathbb{E}[R_{ni}] = \rho$:

$$\mathbb{E}[R_{ni}^2] = \rho, \quad \mathrm{Var}(R_{ni}) = \rho(1-\rho)$$

Assuming independence across i:

$$\mathrm{Var}\left(\sum_{i=1}^{D} w_{ki}R_{ni}x_{ni}\right) = \sum_{i=1}^{D} w_{ki}^2 x_{ni}^2 \mathrm{Var}(R_{ni}) = \rho(1-\rho)\sum_{i=1}^{D} w_{ki}^2 x_{ni}^2$$

Combining both terms:

$$\mathbb{E}[E(\mathbf{W})] = \sum_{n=1}^{N}\sum_{k=1}^{K}\left\{\left(y_{nk} - \rho\sum_{i=1}^{D} w_{ki}x_{ni}\right)^2 + \rho(1-\rho)\sum_{i=1}^{D} w_{ki}^2 x_{ni}^2\right\}$$

Rearranging:

$$\mathbb{E}[E(\mathbf{W})] = \sum_{n=1}^{N}\sum_{k=1}^{K}\left(y_{nk} - \rho\sum_{i=1}^{D} w_{ki}x_{ni}\right)^2 + \rho(1-\rho)\sum_{n=1}^{N}\sum_{k=1}^{K}\sum_{i=1}^{D} w_{ki}^2 x_{ni}^2$$

□

(c) What does the second term of the above expectation resemble?

The second term in this expression represents a **regularization term** applied to our loss function.
It functions similarly to L2 regularization because it includes the square of the weights w_{ki}^2, which penalizes large weight values and encourages the model to maintain simpler weights with smaller magnitudes. This helps prevent overfitting by promoting sparsity and stability in the parameters.