Christopher Zanoli
Student Number: 23-942-394

1. **EM algorithm.** You and your friends decide to participate in the SOLA relay race (`https://www.asvz.ch/event/412-sola-stafette`). In this race, every runner is assigned to one of N tracks and the tracks are then ran in sequence from 1 to N. For simplicity we assume that all tracks are identical (a **very** unrealistic assumption!) and that team member i runs the i-th track. We denote the time it takes the i-th team member to finish their track by the random variable X_i.

   Before the race, each member roughly estimates the time it will take them to finish their track (this is useful for planning), let's denote this time by $c_i > 0$. All team members are, in truth, equally good runners, i.e. $X_i \overset{i.i.d.}{\sim} \exp(\lambda)$, but some runners over and under-estimate their times, i.e. the predicted times c_i vary quite a bit.

   Denote by $Y_i = \mathbf{1}_{X_i \le c_i}$ the random variable that is 1 if member i under-estimated their time (i.e. they went faster than predicted) and 0 for the opposite case.

   If you only have access to $Y_1, \dots, Y_N$ as well as $c_1, \dots, c_N$ (i.e. X_i are unobserved) but you know that all runners are equally skilled (i.e. $X_i \overset{i.i.d.}{\sim} \exp(\lambda)$ for some unknown λ), how would you estimate the average pace of the team?

   Hint 1: For a random variable X and an event A it holds that $\mathbb{E}[X|A] = \frac{1}{\mathbb{P}(A)}\mathbb{E}[X\mathbf{1}_A]$ where $\mathbf{1}_A(x)$ is 1 if $x \in A$ otherwise 0.

   Hint 2: $\int_a^b xe^{-\lambda x}dx = \frac{1}{\lambda^2}\left(\lambda a e^{-\lambda a} - \lambda b e^{-\lambda b} + e^{-\lambda a} - e^{-\lambda b}\right)$

   Answer:

   In order to perform the E-step of the EM algorithm we need to compute $\mathbb{E}[X_i \mid Y_i = 1]$ and $\mathbb{E}[X_i \mid Y_i = 0]$

   - For $\mathbb{E}[X_i \mid Y_i = 1]$:
     From *Hint 1*:

     $$\mathbb{E}[X_i \mid Y_i = 1] = \frac{\mathbb{E}[X_i \mathbf{1}_{\{X_i \le c_i\}}]}{\mathbb{P}(X_i \le c_i)} = \frac{\int_0^{c_i} x\lambda e^{-\lambda x}\,dx}{\mathbb{P}(X_i \le c_i)}$$

     (a) Let's start with the numerator:
     From *Hint 2*:

     $$\int_0^{c_i} x\lambda e^{-\lambda x}\,dx = \lambda \int_0^{c_i} xe^{-\lambda x}\,dx = \lambda\,\frac{1}{\lambda^2}\left(\lambda 0 e^{-\lambda 0} - \lambda c_i e^{-\lambda c_i} + e^{-\lambda 0} - e^{-\lambda c_i}\right) =$$

     $$= \lambda\,\frac{1}{\lambda^2}\left(-\lambda c_i e^{-\lambda c_i} + 1 - e^{-\lambda c_i}\right) = \frac{1}{\lambda} - e^{-\lambda c_i}(c_i + \frac{1}{\lambda})$$

(b) Let's continue with the denominator:
The probability density function of an exponential distribution is:

$$f(x;\lambda) = \begin{cases} \lambda e^{-\lambda x} & x \geq 0, \\ 0 & x < 0. \end{cases}$$

$$\Rightarrow \mathbb{P}(Y_i = 1) = \mathbb{P}(X_i \leq c_i) = \int_0^{c_i} \lambda e^{-\lambda x}\, dx$$

$$\text{Let } u = -\lambda x \Rightarrow du = -\lambda dx \Rightarrow dx = -\frac{du}{\lambda}$$

Therefore, we can compute the former integral by substitution:

$$\int_0^{c_i} \lambda e^{-\lambda x}\, dx = \int_0^{-\lambda c_i} \lambda e^u (-\frac{du}{\lambda}) = \int_0^{-\lambda c_i} -e^u\, du =$$

$$= [-e^u]_0^{-\lambda c_i} = -e^{-\lambda c_i} - (-e^0) = 1 - e^{-\lambda c_i}$$

$$\Rightarrow \mathbb{P}(Y_i = 1) = 1 - e^{-\lambda c_i}$$

Putting them together:

$$\mathbb{E}[X_i \mid Y_i = 1] = \frac{\frac{1}{\lambda} - e^{-\lambda c_i}(c_i + \frac{1}{\lambda})}{1 - e^{-\lambda c_i}} = \frac{\frac{1}{\lambda}(1 - e^{-\lambda c_i}) - e^{-\lambda c_i} c_i}{1 - e^{-\lambda c_i}} =$$

$$= \frac{\frac{1}{\lambda}(1 - e^{-\lambda c_i})}{1 - e^{-\lambda c_i}} - \frac{e^{-\lambda c_i} c_i}{1 - e^{-\lambda c_i}} = \frac{1}{\lambda} - \frac{e^{-\lambda c_i} c_i}{1 - e^{-\lambda c_i}}$$

- For $\mathbb{E}[X_i \mid Y_i = 0]$:
  From *Hint 1*:

$$\mathbb{E}[X_i \mid Y_i = 0] = \frac{\mathbb{E}[X_i \mathbf{1}_{\{X_i > c_i\}}]}{\mathbb{P}(X_i > c_i)} = \frac{\int_{c_i}^{\infty} x\lambda e^{-\lambda x}\, dx}{\mathbb{P}(X_i > c_i)}$$

(a) Let's start with the numerator:
From *Hint 2*:

$$\int_{c_i}^{\infty} x\lambda e^{-\lambda x}\, dx = \lambda \int_{c_i}^{\infty} x e^{-\lambda x}\, dx = \lambda\, \frac{1}{\lambda^2} \left(\lambda c_i e^{-\lambda c_i} - \lambda\infty e^{-\lambda\infty} + e^{-\lambda c_i} - e^{-\lambda\infty}\right) =$$

$$= c_i e^{-\lambda c_i} + \frac{1}{\lambda} e^{-\lambda c_i}$$

(b) Let's continue with the denominator:

$$\mathbb{P}(Y_i = 0) = P(X_i > c_i) = 1 - \mathbb{P}(Y_i = 1) = 1 - (1 - e^{-\lambda c_i}) = e^{-\lambda c_i}$$

Putting them together:

$$\mathbb{E}[X_i \mid Y_i = 0] = \frac{c_i e^{-\lambda c_i} + \frac{1}{\lambda} e^{-\lambda c_i}}{e^{-\lambda c_i}} = \frac{(\frac{1}{\lambda} + c_i) e^{-\lambda c_i}}{e^{-\lambda c_i}} = \frac{1}{\lambda} + c_i$$

Since we are asked the average pace of the team and X_i is exponentially distributed, it is known that $\mathbb{E}[X_i] = \frac{1}{\lambda}$. At this point we have everything needed to perform the EM-algorithm, that allows us to estimate λ:

Step 1: Initialize λ

We can start with an initial guess for λ, denoted as $\lambda^{(0)}$. As we need to estimate the average pace of the team, i.e. $\mathbb{E}[X_i]$, we can initialize $\lambda^{(0)} = \frac{\sum_{i=1}^{N} c_i}{N}$

Step 2: E-step

In this step we compute the conditional expectations according to the formulas derived previously, given the current $\lambda^{(t)}$:

$$\mathbb{E}[X_i \mid Y_i = 1, \lambda^{(t)}] = \frac{1}{\lambda^{(t)}} - \frac{e^{-\lambda^{(t)} c_i} c_i}{1 - e^{-\lambda^{(t)} c_i}}$$

$$\mathbb{E}[X_i \mid Y_i = 0, \lambda^{(t)}] = \frac{1}{\lambda^{(t)}} + c_i$$

Step 3: M-step

For a the random variable X_i that follows an exponential distribution parameterized by λ, the probability density function is $f(X_i; \lambda) = \lambda e^{-\lambda X_i}$ (for $X_i \geq 0$), and the log-likelihood for X_i is $\log(\lambda) - \lambda X_i$.

In this step, we use the expectations computed in the E-step to update λ. We can define $Q(\lambda^{(t+1)} \mid \lambda^{(t)})$ as the expected value of the log likelihood function of $\lambda^{(t+1)}$, with respect to the current conditional distribution of X_i given Y_i and the current estimate of $\lambda^{(t)}$:

$$\begin{aligned}
Q(\lambda^{(t+1)} \mid \lambda^{(\mathrm{t})}) &= \sum_{i=1}^{N} \left(\log(\lambda^{(t+1)}) - \lambda^{(t+1)} \mathbb{E}[X_i | Y_i, \lambda^{(t)}]\right) \\
&= \sum_{i=1}^{N} \left(\log(\lambda^{(t+1)}) - \lambda^{(t+1)} \left(Y_i \ \mathbb{E}[X_i \mid Y_i = 1, \lambda^{(\mathrm{t})}] + (1 - Y_i) \ \mathbb{E}[X_i \mid Y_i = 0, \lambda^{(\mathrm{t})}]\right)\right) \\
&= N \log(\lambda^{(t+1)}) - \lambda^{(t+1)} \sum_{i=1}^{N} \left(Y_i \mathbb{E}[X_i \mid Y_i = 1, \lambda^{(\mathrm{t})}] + (1 - Y_i) \mathbb{E}[X_i \mid Y_i = 0, \lambda^{(\mathrm{t})}]\right)
\end{aligned}$$

To maximize this function we differentiate with respect to $\lambda^{(t+1)}$, setting the derivative to zero, and solve for $\lambda^{(t+1)}$:

$$\frac{\partial}{\partial \lambda^{(t+1)}} Q(\lambda^{(t+1)} | \lambda^{(\mathrm{t})}) =$$

$$= \frac{\partial}{\partial \lambda^{(t+1)}} \left(N \log(\lambda^{(t+1)}) - \lambda^{(t+1)} \sum_{i=1}^{N} \left(Y_i \mathbb{E}[X_i | Y_i = 1, \lambda^{(\mathrm{t})}] + (1 - Y_i) \mathbb{E}[X_i | Y_i = 0, \lambda^{(\mathrm{t})}]\right) \right)$$

$$= \frac{N}{\lambda^{(t+1)}} - \sum_{i=1}^{N} \left(Y_i \mathbb{E}[X_i | Y_i = 1, \lambda^{(\mathrm{t})}] + (1 - Y_i) \mathbb{E}[X_i | Y_i = 0, \lambda^{(\mathrm{t})}]\right)$$

$$\frac{N}{\lambda^{(t+1)}} - \sum_{i=1}^{N} \left(Y_i \mathbb{E}[X_i | Y_i = 1, \lambda^{(\text{t})}] + (1 - Y_i) \mathbb{E}[X_i | Y_i = 0, \lambda^{(\text{t})}]\right) \overset{!}{=} 0$$

$$\Rightarrow \lambda^{(t+1)} = \frac{N}{\sum_{i=1}^{N} (Y_i \mathbb{E}[X_i | Y_i = 1, \lambda^{(t)}] + (1 - Y_i) \mathbb{E}[X_i | Y_i = 0, \lambda^{(t)}])}$$

Step 4: Iterate

Now we can repeat the E-step and M-step until convergence, i.e. until when the changes in λ are below a small threshold ϵ.

Once the algorithm have converged we end up with a λ^{final} that can be used to estimate the average pace of the team (that is comprised of N teammates): $\mathbb{E}[\text{team}] = \sum_{i=1}^{N} \mathbb{E}[X_i] = \frac{N}{\lambda^{final}}$

3 pts □

2. **Matrix-matrix backpropagation.** In Exercise 9.4, we learnt how to backpropagate through matrix multiplication. But to be precise, we "only" learnt how to backpropagate through a matrix-vector product, i.e. for a vector $\boldsymbol{x} \in \mathbb{R}^D$, a matrix $\boldsymbol{W} \in \mathbb{R}^{E \times D}$ and the resulting vector $\boldsymbol{z} = \boldsymbol{W}\boldsymbol{x}$, we derived how to compute $\frac{\partial L}{\partial \boldsymbol{x}}$ and $\frac{\partial L}{\partial \boldsymbol{W}}$. In deep learning, we rarely just work with a single datapoint $\boldsymbol{x} \in \mathbb{R}^D$ at once, but rather with a batch $\boldsymbol{X} \in \mathbb{R}^{B \times D}$ in order to speed up computation. We then perform a matrix-matrix product "in parallel",

$$\boldsymbol{Z} = \boldsymbol{X}\boldsymbol{W}$$

Careful: The weights now have dimension $\boldsymbol{W} \in \mathbb{R}^{D \times E}$ and the result is also a matrix, $\boldsymbol{Z} \in \mathbb{R}^{B \times E}$, not a vector anymore.

Assume the exact same setup as in the exercise sheet, i.e. that L is some scalar-valued loss function and that we already computed $\frac{\partial L}{\partial \boldsymbol{Z}}$. Derive $\frac{\partial L}{\partial \boldsymbol{X}}$ and $\frac{\partial L}{\partial \boldsymbol{W}}$ as a function of $\frac{\partial L}{\partial \boldsymbol{Z}}$, $\boldsymbol{X}$ and $\boldsymbol{W}$. Express your end result in terms of matrices.

Hint: It might be easier to derive gradients entrywise (e.g. $\frac{\partial L}{\partial W_{ij}}$) and then in the end go back to the matrix level.

Answer: Given the setup of exercise 9.4 with the input now being $\boldsymbol{X} \in \mathbb{R}^{B \times D}$, the new setup is the following:

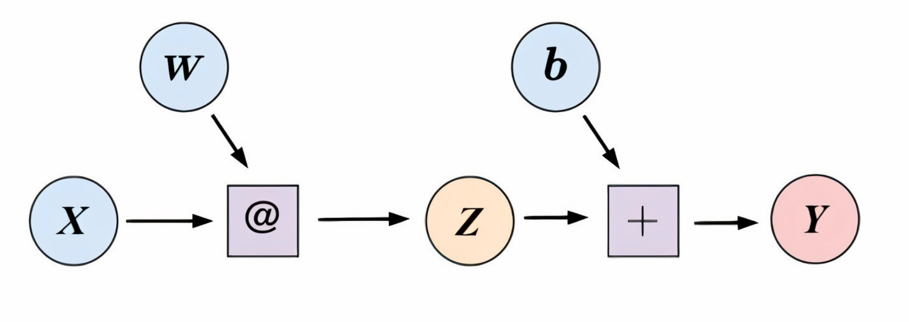

Figure 1: Computational Graph.

As stated by the request we assume we already computed $\frac{\partial L}{\partial \boldsymbol{Z}}$, thus we already performed the forward pass, so we have:

$$\boldsymbol{Z} \in \mathbb{R}^{B\times E} = \boldsymbol{X}\boldsymbol{W} = \begin{pmatrix} x_{1,1} & \cdots & x_{1,D} \\ \vdots & \ddots & \vdots \\ x_{B,1} & \cdots & x_{B,D} \end{pmatrix} \begin{pmatrix} w_{1,1} & \cdots & w_{1,E} \\ \vdots & \ddots & \vdots \\ w_{D,1} & \cdots & w_{D,E} \end{pmatrix}$$

$$= \begin{pmatrix} x_{1,1}w_{1,1} + \cdots + x_{1,D}w_{D,1} & \cdots & x_{1,1}w_{1,E} + \cdots + x_{1,D}w_{D,E} \\ \vdots & \ddots & \vdots \\ x_{B,1}w_{1,1} + \cdots + x_{B,D}w_{D,1} & \cdots & x_{B,1}w_{1,E} + \cdots + x_{B,D}w_{D,E} \end{pmatrix}$$

Now, for the backward pass, we need to find $\frac{\partial L}{\partial \boldsymbol{X}}$ and $\frac{\partial L}{\partial \boldsymbol{W}}$ as a function of $\frac{\partial L}{\partial \boldsymbol{Z}}$, $\boldsymbol{Z}$ and $\boldsymbol{W}$.

- Let's compute $\frac{\partial L}{\partial \boldsymbol{X}}$:

  First, we can notice that since L is a scalar and $\boldsymbol{X} \in \mathbb{R}^{B\times D} \Rightarrow \frac{\partial L}{\partial \boldsymbol{X}} \in \mathbb{R}^{B\times D}$:

$$\frac{\partial L}{\partial \boldsymbol{X}} = \begin{pmatrix} \frac{\partial L}{\partial x_{1,1}} & \frac{\partial L}{\partial x_{1,2}} & \cdots & \frac{\partial L}{\partial x_{1,D}} \\ \frac{\partial L}{\partial x_{2,1}} & \frac{\partial L}{\partial x_{2,2}} & \cdots & \frac{\partial L}{\partial x_{2,D}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial L}{\partial x_{B,1}} & \frac{\partial L}{\partial x_{B,2}} & \cdots & \frac{\partial L}{\partial x_{B,D}} \end{pmatrix}$$

  From the chain rule:

$$\frac{\partial L}{\partial \boldsymbol{X}} = \frac{\partial L}{\partial \boldsymbol{Z}} \cdot \frac{\partial \boldsymbol{Z}}{\partial \boldsymbol{X}}$$

  Now, since L is a scalar and $\boldsymbol{Z} \in \mathbb{R}^{B\times E} \Rightarrow \frac{\partial L}{\partial \boldsymbol{Z}} \in \mathbb{R}^{B\times E}$:

$$\frac{\partial L}{\partial \boldsymbol{Z}} = \begin{pmatrix} \frac{\partial L}{\partial z_{1,1}} & \frac{\partial L}{\partial z_{1,2}} & \cdots & \frac{\partial L}{\partial z_{1,E}} \\ \frac{\partial L}{\partial z_{2,1}} & \frac{\partial L}{\partial z_{2,2}} & \cdots & \frac{\partial L}{\partial z_{2,E}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial L}{\partial z_{B,1}} & \frac{\partial L}{\partial z_{B,2}} & \cdots & \frac{\partial L}{\partial z_{B,E}} \end{pmatrix}$$

We can fix two indices i and j to derive multivariate gradients on an element-wise basis, i.e. $\frac{\partial L}{\partial X_{i,j}}$ instead of directly characterizing $\frac{\partial L}{\partial \boldsymbol{X}}$:

$$\frac{\partial L}{\partial X_{ij}} = \sum_{k=1}^{B}\sum_{l=1}^{E} \frac{\partial L}{\partial Z_{kl}} \frac{\partial Z_{kl}}{\partial X_{ij}} = \sum_{k=1}^{B}\sum_{l=1}^{E} \frac{\partial \left(\sum_{t=1}^{D} X_{kt} W_{tl} \right)}{\partial X_{ij}} =$$

$$= \sum_{k=1}^{B}\sum_{l=1}^{E} \frac{\partial L}{\partial Z_{kl}} \sum_{t=1}^{D} \mathbb{1}_{\{k=i\}} \mathbb{1}_{\{t=j\}} W_{tl} =$$

$$= \sum_{k=1}^{B}\sum_{l=1}^{E} \frac{\partial L}{\partial Z_{kl}} \mathbb{1}_{\{k=i\}} W_{jl} = \sum_{k=1}^{E} \frac{\partial L}{\partial Z_{il}} W_{jl} =$$

$$= \sum_{k=1}^{E} \frac{\partial L}{\partial Z_{il}} \left(\boldsymbol{W}^{T} \right)_{lj} = \left(\frac{\partial L}{\partial \boldsymbol{Z}} \boldsymbol{W^{T}} \right)_{ij}$$

Thus, in matrix form we can express it as follows:

$$\frac{\partial L}{\partial \boldsymbol{X}} = \frac{\partial L}{\partial \boldsymbol{Z}} \boldsymbol{W}^{T}$$

- Now let's compute $\frac{\partial L}{\partial \boldsymbol{W}}$:

First, we can notice that since L is a scalar and $\boldsymbol{W} \in \mathbb{R}^{D \times E} \Rightarrow \frac{\partial L}{\partial \boldsymbol{W}} \in \mathbb{R}^{D \times E}$:

$$\frac{\partial L}{\partial \boldsymbol{W}} = \begin{pmatrix} \frac{\partial L}{\partial w_{1,1}} & \frac{\partial L}{\partial w_{1,2}} & \cdots & \frac{\partial L}{\partial w_{1,E}} \\ \frac{\partial L}{\partial w_{2,1}} & \frac{\partial L}{\partial w_{2,2}} & \cdots & \frac{\partial L}{\partial w_{2,E}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial L}{\partial w_{D,1}} & \frac{\partial L}{\partial w_{D,2}} & \cdots & \frac{\partial L}{\partial w_{D,E}} \end{pmatrix}$$

From the chain rule:

$$\frac{\partial L}{\partial \boldsymbol{W}} = \frac{\partial L}{\partial \boldsymbol{Z}} \cdot \frac{\partial \boldsymbol{Z}}{\partial \boldsymbol{W}}$$

Now, since L is a scalar and $\boldsymbol{Z} \in \mathbb{R}^{B \times E} \Rightarrow \frac{\partial L}{\partial \boldsymbol{Z}} \in \mathbb{R}^{B \times E}$:

$$\frac{\partial L}{\partial \boldsymbol{Z}} = \begin{pmatrix} \frac{\partial L}{\partial z_{1,1}} & \frac{\partial L}{\partial z_{1,2}} & \cdots & \frac{\partial L}{\partial z_{1,E}} \\ \frac{\partial L}{\partial z_{2,1}} & \frac{\partial L}{\partial z_{2,2}} & \cdots & \frac{\partial L}{\partial z_{2,E}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial L}{\partial z_{B,1}} & \frac{\partial L}{\partial z_{B,2}} & \cdots & \frac{\partial L}{\partial z_{B,E}} \end{pmatrix}$$

Again, we can fix two indices i and j to derive multivariate gradients on an element-wise basis, i.e. $\frac{\partial L}{\partial W_{i,j}}$ instead of directly characterizing $\frac{\partial L}{\partial \boldsymbol{W}}$:

$$\frac{\partial L}{\partial W_{ij}} = \sum_{k=1}^{B}\sum_{l=1}^{E} \frac{\partial L}{\partial Z_{kl}} \frac{\partial Z_{kl}}{\partial W_{ij}} = \sum_{k=1}^{B}\sum_{l=1}^{E} \frac{\partial \left(\sum_{t=1}^{D} X_{kt} W_{tl} \right)}{\partial W_{ij}} =$$

$$= \sum_{k=1}^{B}\sum_{l=1}^{E}\frac{\partial L}{\partial Z_{kl}}\sum_{t=1}^{D}\mathbb{1}_{\{i=t\}}\mathbb{1}_{\{j=l\}}X_{kt} =$$

$$= \sum_{k=1}^{B}\sum_{l=1}^{E}\frac{\partial L}{\partial Z_{kl}}\mathbb{1}_{\{j=l\}}X_{ki} = \sum_{k=1}^{B}\frac{\partial L}{\partial Z_{kj}}X_{ki} =$$

$$= \sum_{k=1}^{B}\left(\boldsymbol{X}^T\right)_{ik}\frac{\partial L}{\partial Z_{kj}} = \left(\boldsymbol{X}^T\frac{\partial L}{\partial \boldsymbol{Z}}\right)_{ij}$$

Thus, in matrix form we can express it as follows:

$$\frac{\partial L}{\partial \boldsymbol{W}} = \boldsymbol{X}^T\frac{\partial L}{\partial \boldsymbol{Z}}$$

3 pts □

3. **Softmax backpropagation.** Another essential building block in deep learning (especially with the rise of Transformers) is the `softmax` function, which, given a vector $\boldsymbol{x} \in \mathbb{R}^C$, is defined as

$$\boldsymbol{z} = \texttt{softmax}(\boldsymbol{x}) = \frac{\exp(\boldsymbol{x})}{\sum_c^C \exp(x_c)}$$

or element-wise, $\texttt{softmax}(\boldsymbol{x})_i = \frac{\exp(x_i)}{\sum_c^C \exp(x_c)}$. Again the goal of this exercise is to calculate $\frac{\partial L}{\partial \boldsymbol{x}}$ as a function of $\frac{\partial L}{\partial \boldsymbol{z}}$ and $\boldsymbol{x}$. Instead of calculating the gradient in one go, we will break this into rather easy sub-problems.

a. Draw a "computational graph" style diagram of the `softmax` function, in the spirit of exercise 9.2.

Answer: The figure below shows the computational graph of the softmax function. Each circle represents a scalar value (as the convention used in exercise 9.2 of the exercise sheet). These are the steps performed by the computational graph:

(a) `exp()`: It is a unary operator (it takes a single input and return a single output). It takes the element x_i of the vector $\boldsymbol{x}$ as input and computes $q_i = e^{x_i}$ as output.

(b) `[+]`: It takes all the q_i ($\forall i = 1 \cdots C$) and compute the summation s

(c) `[/]`: This is a binary operator (it takes two inputs and returns a single output). It returns z_i ($\forall i = 1 \cdots C$) as output, i.e. each i_{th} element of the vector $\boldsymbol{z}$. It takes q_i ($\forall i = 1 \cdots C$) as the numerator and s as the denominator.

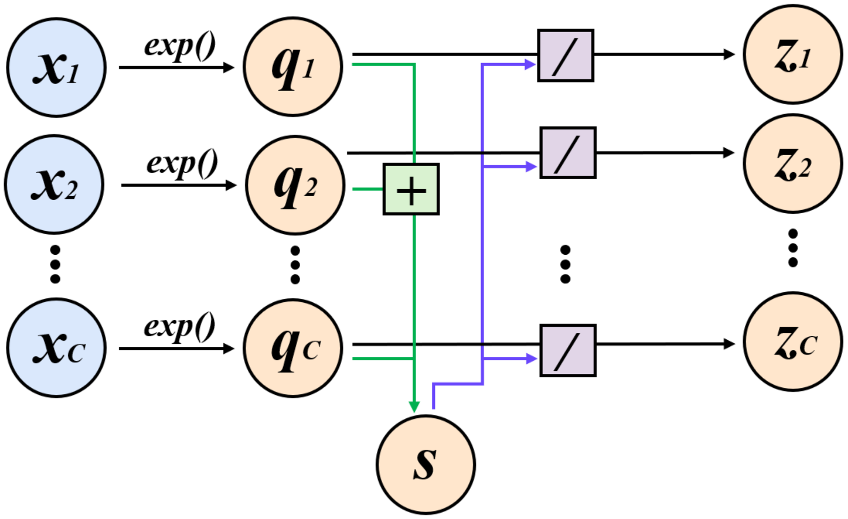

Figure 2: Computational Graph of the Softmax Function.

0.5 pts □

b. Derive the backprop rule for elementwise exponentiation, $\boldsymbol{z} = \exp(\boldsymbol{x})$ for $\boldsymbol{x} \in \mathbb{R}^C$, i.e. calculate $\frac{\partial L}{\partial \boldsymbol{x}}$ assuming that you know $\frac{\partial L}{\partial \boldsymbol{z}}$.

Answer: In order to derive the rule for element-wise exponentiation we can first start by noticing that due to the derivation rule $\exp(x) = x$, then:

$$\frac{\partial z_i}{\partial x_i} = \frac{\partial \exp(x_i)}{\partial x_i} = \exp(x_i) = z_i$$

Since we need to compute $\frac{\partial L}{\partial \boldsymbol{x}}$ and we know $\frac{\partial L}{\partial \boldsymbol{z}}$, then:

$$\frac{\partial L}{\partial x_i} = \frac{\partial L}{\partial z_i}\frac{\partial z_i}{\partial x_i} = \frac{\partial L}{\partial z_i} z_i$$

Finally, we can write the above formula in the vectorized form by leveraging the element-wise product as:

$$\frac{\partial L}{\partial \boldsymbol{x}} = \frac{\partial L}{\partial \boldsymbol{z}} \odot \boldsymbol{z}$$

Where $\boldsymbol{z} = \exp(\boldsymbol{x})$.

1 pts □

c. For two vectors $\boldsymbol{x} \in \mathbb{R}^C$ and $\boldsymbol{y} \in \mathbb{R}^C$ consider the element-wise division

$$\boldsymbol{z} = \frac{\boldsymbol{x}}{\boldsymbol{y}}$$

Again derive $\frac{\partial L}{\partial \boldsymbol{x}}$ and $\frac{\partial L}{\partial \boldsymbol{y}}$ assuming that you know $\frac{\partial L}{\partial \boldsymbol{z}}$.

Answer: Again let's start by considering just the i_{th} element. From the chain rule:

$$\frac{\partial L}{\partial x_i} = \frac{\partial L}{\partial z_i}\frac{\partial z_i}{\partial x_i}$$

$$\frac{\partial L}{\partial y_i} = \frac{\partial L}{\partial z_i}\frac{\partial z_i}{\partial y_i}$$

Since we assume we know $\frac{\partial L}{\partial \boldsymbol{z}}$, we can just compute $\frac{\partial z_i}{\partial x_i}$ and $\frac{\partial z_i}{\partial y_i}$ as follows:

$$\frac{\partial z_i}{\partial x_i} = \frac{\partial(\frac{x_i}{y_i})}{\partial x_i} = \frac{1}{y_i}\frac{\partial x_i}{\partial x_i} = \frac{1}{y_i}$$

$$\frac{\partial z_i}{\partial y_i} = \frac{\partial(\frac{x_i}{y_i})}{\partial y_i} = x_i\frac{\partial(y_i^{-1})}{\partial y_i} = x_i(-y_i^{-2}) = -\frac{x_i}{y_i^2}$$

$$\Rightarrow \frac{\partial L}{\partial x_i} = \frac{\partial L}{\partial z_i}\left(\frac{1}{y_i}\right)$$

$$\Rightarrow \frac{\partial L}{\partial y_i} = \frac{\partial L}{\partial z_i}\left(-\frac{x_i}{y_i^2}\right)$$

Again we can express them in the vectorized form by leveraging the element-wise product as:

$$\frac{\partial L}{\partial \boldsymbol{x}} = \frac{\partial L}{\partial \boldsymbol{z}} \odot \frac{\mathbf{1}}{\boldsymbol{y}}$$

Where $\mathbf{1}$ is a vector of all ones and $\frac{\mathbf{1}}{\boldsymbol{y}}$ is the element-wise division.

$$\frac{\partial L}{\partial \boldsymbol{y}} = -\frac{\partial L}{\partial \boldsymbol{z}} \odot \frac{\boldsymbol{x}}{\boldsymbol{y}\odot\boldsymbol{y}}$$

Where $\frac{\boldsymbol{x}}{\boldsymbol{y}\odot\boldsymbol{y}}$ is the element-wise division.

1 pts □

d. Finally, for a vector $\boldsymbol{x} \in \mathbb{R}^C$ consider the sum-reduction

$$z = \sum_c^C x_i$$

Again derive $\frac{\partial L}{\partial \boldsymbol{x}}$ assuming that you know $\frac{\partial L}{\partial z}$.

Answer: Again let's start by considering just the i_{th} element. From the chain rule:

$$\frac{\partial L}{\partial x_i} = \frac{\partial L}{\partial z}\frac{\partial z}{\partial x_i}$$

Again, assuming we know $\frac{\partial L}{\partial z}$, we can just compute $\frac{\partial z}{\partial x_i}$ as follows:

$$\frac{\partial z}{\partial x_i} = \frac{\partial \sum_c^C x_c}{\partial x_i}$$

We can notice that when $c \neq i$ the partial derivative involving the summation is 0. When $c = i$, then the partial derivative involving the summation is 1.

$$\Rightarrow \frac{\partial L}{\partial x_i} = \frac{\partial L}{\partial z}$$

Again, we can express it in vectorized form as:

$$\frac{\partial L}{\partial \boldsymbol{x}} = \frac{\partial L}{\partial z}\mathbf{1}$$

where $\mathbf{1} \in \mathbb{R}^C$ is a vector of all ones.

1 pts □

e. Explain now in words how we can backpropagate through the `softmax` function by leveraging all these "smaller" backprop steps we just derived. (You can even solve this if you did not solve all the previous problems!)

Answer: We can observe that the chain rule involved during the backpropagation of the `softmax` function requires exactly the smaller steps performed in point b), c) and d) of the question. Since the `softmax` function is defined as:

$$\boldsymbol{z} = \texttt{softmax}(\boldsymbol{x}) = \frac{\exp(\boldsymbol{x})}{\sum_c^C \exp(x_c)}$$

We should first compute the partial derivative of the loss function with respect to both the numerator and the denominator, exactly as done in point c). In this case numerator is $\exp(\boldsymbol{x})$ while the denominator is $\sum_c^C \exp(x_c)$.

Then we should compute the partial derivative of the loss function with respect to $\sum_c^C \exp(x_c)$. This is exactly what we have done in point d).

Finally, once we have performed the aforementioned steps, we have everything we need to compute the partial derivative of the loss function with respect to $\exp(\boldsymbol{x})$. This is exactly what we have done in point a).

0.5 pts □