Working with Tensors: Index Gymnastics

Review

TA Team

Informatics Institute
University of Amsterdam

Deep Learning I, Fall 2022

Table of Contents

Table of Contents

Motivation

Again?... Here's why:

- Refresher on notation: combining linear algebra with calculus

Motivation

Again?... Here's why:

- Refresher on notation: combining linear algebra with calculus
- We will see that convention **does not matter** in index notation!

Motivation

Again?... Here's why:

- Refresher on notation: combining linear algebra with calculus
- We will see that convention **does not matter** in index notation!
- Goal: Code an MLP from **scratch**

Motivation

Again?... Here's why:

- Refresher on notation: combining linear algebra with calculus
- We will see that convention **does not matter** in index notation!
- Goal: Code an MLP from **scratch**
- Object Oriented Programming $\rightarrow$ **modular** approach

Motivation

Again?... Here's why:

- Refresher on notation: combining linear algebra with calculus
- We will see that convention **does not matter** in index notation!
- Goal: Code an MLP from **scratch**
- Object Oriented Programming $\rightarrow$ **modular** approach
- Backpropagation equations are actually **simpler**!

$h = \Phi(a_h)$ is a_h. The differences between the pre-activation and post-activation values within a neuron are shown in Figure 1.7. Therefore, instead of Equation 1.23, one can use the following chain rule:

$$\frac{\partial L}{\partial w_{(h_{r-1},h_r)}} = \underbrace{\frac{\partial L}{\partial o} \cdot \Phi'(a_o) \cdot \left[\sum_{[h_r, h_{r+1}, \ldots h_k, o] \in \mathcal{P}} \frac{\partial a_o}{\partial a_{h_k}} \prod_{i=r}^{k-1} \frac{\partial a_{h_{i+1}}}{\partial a_{h_i}} \right]}_{\text{Backpropagation computes } \delta(h_r, o) = \frac{\partial L}{\partial a_{h_r}}} \underbrace{\frac{\partial a_{h_r}}{\partial w_{(h_{r-1},h_r)}}}_{h_{r-1}} \tag{1.28}$$

Here, we have introduced the notation $\delta(h_r, o) = \frac{\partial L}{\partial a_{h_r}}$ instead of $\Delta(h_r, o) = \frac{\partial L}{\partial h_r}$ for setting up the recursive equation. The value of $\delta(o, o) = \frac{\partial L}{\partial a_o}$ is initialized as follows:

$$\delta(o, o) = \frac{\partial L}{\partial a_o} = \Phi'(a_o) \cdot \frac{\partial L}{\partial o} \tag{1.29}$$

Then, one can use the multivariable chain rule to set up a similar recursion:

$$\delta(h_r, o) = \frac{\partial L}{\partial a_{h_r}} = \sum_{h: h_r \Rightarrow h} \overbrace{\frac{\partial L}{\partial a_h}}^{\delta(h,o)} \underbrace{\frac{\partial a_h}{\partial a_{h_r}}}_{\Phi'(a_{h_r}) w_{(h_r,h)}} = \Phi'(a_{h_r}) \sum_{h: h_r \Rightarrow h} w_{(h_r,h)} \cdot \delta(h, o) \tag{1.30}$$

This recursion condition is found more commonly in textbooks discussing backpropagation

Figure: Backpropagation Equations in textbooks (Aggarwal)

Rank

The rank of an array refers to the dimensionality of its inherent structure. *Note the number of independent indices!*

- scalar s has rank 0
- vector $\mathbf{v}$ has rank 1 (v_i)
- matrix $\mathbf{M}$ has a rank of 2 (M_{ij})
- object $\mathbf{T}$ with elements T_{ijk} is a 3-rank tensor
- etc.

Rank

The rank of an array refers to the dimensionality of its inherent structure. *Note the number of independent indices!*

- scalar s has rank 0
- vector $\mathbf{v}$ has rank 1 (v_i)
- matrix $\mathbf{M}$ has a rank of 2 (M_{ij})
- object $\mathbf{T}$ with elements T_{ijk} is a 3-rank tensor
- etc.

An array of higher rank is often simply referred to as a **tensor**.

Rank

The rank of an array refers to the dimensionality of its inherent structure. *Note the number of independent indices!*

- scalar s has rank 0
- vector $\mathbf{v}$ has rank 1 (v_i)
- matrix $\mathbf{M}$ has a rank of 2 (M_{ij})
- object $\mathbf{T}$ with elements T_{ijk} is a 3-rank tensor
- etc.

An array of higher rank is often simply referred to as a **tensor**. The most important takeaway of working with tensors is to keep *good algebraic hygiene* throughout your calculations.

Kronecker Delta and Sifting

We can introduce "if-statements" into our calculations:

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

Kronecker Delta and Sifting

We can introduce "if-statements" into our calculations:

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

Sifting property:

$$\sum_{i=1}^{n} a_i \delta_{i{\color{red}k}}$$

Kronecker Delta and Sifting

We can introduce "if-statements" into our calculations:

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

Sifting property:

$$\sum_{i=1}^{n} a_i \delta_{i\textcolor{red}{k}} = a_1\delta_{1k} + \ldots + a_k\delta_{kk} + \ldots + a_n\delta_{nk}$$

Kronecker Delta and Sifting

We can introduce "if-statements" into our calculations:

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

Sifting property:

$$\sum_{i=1}^{n} a_i \delta_{i\color{red}{k}} = a_1 \delta_{1k} + \ldots + a_k \delta_{kk} + \ldots + a_n \delta_{nk} = a_{\color{red}{k}}$$

Kronecker Delta and Sifting

We can introduce "if-statements" into our calculations:

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

Sifting property:

$$\sum_{i=1}^{n} a_i \delta_{i\color{red}{k}} = a_1 \delta_{1k} + \ldots + a_k \delta_{kk} + \ldots + a_n \delta_{nk} = a_{\color{red}{k}}$$

Note the dummy index i and free index $\color{red}{k}$.

Kronecker Delta and Sifting

We can introduce "if-statements" into our calculations:

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

Sifting property:

$$\sum_{i=1}^{n} a_i \delta_{i\textcolor{red}{k}} = a_1 \delta_{1k} + \ldots + a_k \delta_{kk} + \ldots + a_n \delta_{nk} = a_{\textcolor{red}{k}}$$

Note the dummy index i and free index $\textcolor{red}{k}$. For calculus:

$$\frac{\partial x_i}{\partial x_j} = \delta_{ij}.$$

Kronecker Delta and Sifting

We can introduce "if-statements" into our calculations:

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

Sifting property:

$$\sum_{i=1}^{n} a_i \delta_{ik} = a_1 \delta_{1k} + \ldots + a_k \delta_{kk} + \ldots + a_n \delta_{nk} = a_k$$

Note the dummy index i and free index k. For calculus:

$$\frac{\partial x_i}{\partial x_j} = \delta_{ij}.$$

The Kronecker Delta is also the result of *indexing*[1] the identity matrix:

$$[\mathbf{I}]_{ij} = \delta_{ij}.$$

[1]For generic matrix $\mathbf{M}$ we write $[\mathbf{M}]_{ij} = M_{ij}$

More Tools

Another useful piece of notation is that for the *trace* of a square matrix $\mathbf{S} \in \mathbb{R}^{m \times m}$,

$$tr(S) := \sum_i S_{ii}.$$

Another useful piece of notation is that for the *trace* of a square matrix $\mathbf{S} \in \mathbb{R}^{m \times m}$,

$$tr(S) := \sum_i S_{ii}.$$

Sometimes it will be useful to introduce the *ones-vector* $\mathbf{1}$, which simply has all components equal to unity:

$$[\mathbf{1}]_i = 1.$$

More Tools

Another useful piece of notation is that for the *trace* of a square matrix $\mathbf{S} \in \mathbb{R}^{m \times m}$,

$$tr(S) := \sum_i S_{ii}.$$

Sometimes it will be useful to introduce the *ones-vector* $\mathbf{1}$, which simply has all components equal to unity:

$$[\mathbf{1}]_i = 1.$$

The *Hadamard product* or element-wise product between two matrices of identical size is given by $\mathbf{A} \circ \mathbf{B}$. The elements of the result are

$$[\mathbf{A} \circ \mathbf{B}]_{ij} = A_{ij} B_{ij}.$$

Table of Contents

Examples

Recall the definition of *matrix-multiplication*:

$$[\mathbf{A}]_{ij} = [\mathbf{BC}]_{ij}$$
$$A_{ij} = \sum_{p} B_{ip} C_{pj}.$$

Example 1

Question: Let $r = \mathbf{x} \cdot \mathbf{a} \in \mathbb{R}$ for vectors $\mathbf{x}, \mathbf{a} \in \mathbb{R}^n$. What is $\frac{\partial r}{\partial \mathbf{x}}$?

Examples

Recall the definition of *matrix-multiplication*:

$$[\mathbf{A}]_{ij} = [\mathbf{BC}]_{ij}$$
$$A_{ij} = \sum_{p} B_{ip} C_{pj}.$$

Example 1

Question: Let $r = \mathbf{x} \cdot \mathbf{a} \in \mathbb{R}$ for vectors $\mathbf{x}, \mathbf{a} \in \mathbb{R}^n$. What is $\frac{\partial r}{\partial \mathbf{x}}$?

Example 2

Question: Consider the scalar $s = \mathbf{b}^\top \mathbf{X} \mathbf{c}$, where $\mathbf{b} \in \mathbb{R}^m$, $\mathbf{c} \in \mathbb{R}^n$, and $\mathbf{X} \in \mathbb{R}^{m \times n}$. Find $\frac{\partial s}{\partial \mathbf{X}}$.

Your Turn

Remember that during coding, you can always print the shapes of your tensors with `numpy.shape` or `torch.size` to check that your calculations correspond to what is happening under the hood!

Exercise 1

Question: For vector $\mathbf{x} \in \mathbb{R}^n$ and square matrix $\mathbf{B} \in \mathbb{R}^{n \times n}$, evaluate $\frac{\partial \mathbf{x}^\top \mathbf{B} \mathbf{x}}{\partial \mathbf{x}}$.
Answer: $(\mathbf{B} + \mathbf{B}^\top)\mathbf{x}$.

Your Turn

Remember that during coding, you can always print the shapes of your tensors with `numpy.shape` or `torch.size` to check that your calculations correspond to what is happening under the hood!

Exercise 1

Question: For vector $\mathbf{x} \in \mathbb{R}^n$ and square matrix $\mathbf{B} \in \mathbb{R}^{n \times n}$, evaluate $\frac{\partial \mathbf{x}^\top \mathbf{B} \mathbf{x}}{\partial \mathbf{x}}$.
Answer: $(\mathbf{B} + \mathbf{B}^\top)\mathbf{x}$.

Exercise 2

Question: Given matrices $\mathbf{V}$ and $\mathbf{W}$. Find an expression for $\frac{\partial \mathrm{tr}(\mathbf{VXW})}{\partial \mathbf{X}}$.
Answer: $\mathbf{V}^\top \mathbf{W}^\top$.

Exercise 1

Solution: Note that the object being evaluated is a 1-rank tensor. Hence,

$$
\begin{aligned}
\frac{\partial \mathbf{x}^\top \mathbf{B}\mathbf{x}}{\partial x_i} &= \frac{\partial}{\partial x_i}\sum_{p,q} x_p B_{pq} x_q = \sum_{p,q} \frac{\partial x_p}{\partial x_i} B_{pq} x_q + \sum_{p,q} x_p B_{pq} \frac{\partial x_q}{\partial x_i} \\
&= \sum_{p,q} \delta_{pi} B_{pq} x_q + \sum_{p,q} x_p B_{pq} \delta_{qi} = \sum_q B_{iq} x_q + \sum_p x_p B_{pi} \\
&= [\mathbf{B}\mathbf{x}]_i + [\mathbf{x}^\top \mathbf{B}]_i .
\end{aligned}
$$

Exercise 1

Solution: Note that the object being evaluated is a 1-rank tensor. Hence,

$$\begin{aligned}
\frac{\partial \mathbf{x}^\top \mathbf{B}\mathbf{x}}{\partial x_i} &= \frac{\partial}{\partial x_i} \sum_{p,q} x_p B_{pq} x_q = \sum_{p,q} \frac{\partial x_p}{\partial x_i} B_{pq} x_q + \sum_{p,q} x_p B_{pq} \frac{\partial x_q}{\partial x_i} \\
&= \sum_{p,q} \delta_{pi} B_{pq} x_q + \sum_{p,q} x_p B_{pq} \delta_{qi} = \sum_q B_{iq} x_q + \sum_p x_p B_{pi} \\
&= [\mathbf{B}\mathbf{x}]_i + [\mathbf{x}^\top \mathbf{B}]_i .
\end{aligned}$$

Let us choose the column-vector representation and use the following observation: $[\mathbf{v}]_j = [\mathbf{v}^\top]_j$. Finally,

$$\left[\frac{\partial \mathbf{x}^\top \mathbf{B}\mathbf{x}}{\partial \mathbf{x}}\right]_i = [\mathbf{B}\mathbf{x}]_i + [\mathbf{x}^\top \mathbf{B}]_i = [\mathbf{B}\mathbf{x}]_i + [\mathbf{B}^\top \mathbf{x}]_i = [(\mathbf{B} + \mathbf{B}^\top)\mathbf{x}]_i .$$

Exercise 2

Solution: This is a 2-rank object. Index and expand to obtain:

$$
\begin{aligned}
\frac{\partial \mathrm{tr}(\mathbf{VXW})}{\partial X_{ij}} &= \frac{\partial}{\partial X_{ij}} \sum_t [\mathbf{VXW}]_{tt} = \frac{\partial}{\partial X_{ij}} \sum_{t,p,q} V_{tp} X_{pq} W_{qt} \\
&= \sum_{t,p,q} V_{tp} \frac{\partial X_{pq}}{\partial X_{ij}} W_{qt} = \sum_{t,p,q} V_{tp} \delta_{pi} \delta_{qj} W_{qt} = \sum_{t,q} V_{ti} \delta_{qj} W_{qt} \\
&= \sum_t V_{ti} W_{jt} = \sum_t V_{it}^\top W_{tj}^\top = \sum_t [\mathbf{V}^\top]_{it} [\mathbf{W}^\top]_{tj} \\
&= [\mathbf{V}^\top \mathbf{W}^\top]_{ij}
\end{aligned}
$$

Recall: The product of three matrices can be resolved as follows

$$[\mathbf{VXW}]_{mn} = \sum_{r,s} V_{mr} X_{rs} W_{sn}.$$

Example 3

Question: Find an expression for $\frac{\partial \mathbf{Q}^\top \mathbf{Q}}{\partial \mathbf{Q}}$, where $\mathbf{Q} \in \mathbb{R}^{p \times q}$.

Example 3

Question: Find an expression for $\frac{\partial \mathbf{Q}^\top \mathbf{Q}}{\partial \mathbf{Q}}$, where $\mathbf{Q} \in \mathbb{R}^{p\times q}$.

It helps to rename the product such that $\mathbf{R} := \mathbf{Q}^\top \mathbf{Q}$, then the task is to evaluate $\frac{\partial \mathbf{R}}{\partial \mathbf{Q}}$. This object has four indices, i.e. it is a 4-rank tensor.

Example 3

Question: Find an expression for $\frac{\partial \mathbf{Q}^\top \mathbf{Q}}{\partial \mathbf{Q}}$, where $\mathbf{Q} \in \mathbb{R}^{p \times q}$.

It helps to rename the product such that $\mathbf{R} := \mathbf{Q}^\top \mathbf{Q}$, then the task is to evaluate $\frac{\partial \mathbf{R}}{\partial \mathbf{Q}}$. This object has four indices, i.e. it is a 4-rank tensor. (We now have a $q \times q$ size matrix in the "numerator", and a $p \times q$ size matrix in the "denominator". So there are four free indices and the object has pq^3 entries.)

Example 3

Question: Find an expression for $\frac{\partial \mathbf{Q}^\top \mathbf{Q}}{\partial \mathbf{Q}}$, where $\mathbf{Q} \in \mathbb{R}^{p \times q}$.

It helps to rename the product such that $\mathbf{R} := \mathbf{Q}^\top \mathbf{Q}$, then the task is to evaluate $\frac{\partial \mathbf{R}}{\partial \mathbf{Q}}$. This object has four indices, i.e. it is a 4-rank tensor. (We now have a $q \times q$ size matrix in the "numerator", and a $p \times q$ size matrix in the "denominator". So there are four free indices and the object has pq^3 entries.) Then:

$$\frac{\partial R_{ij}}{\partial Q_{mn}} = \frac{\partial [\mathbf{Q}^\top \mathbf{Q}]_{ij}}{\partial Q_{mn}} = \frac{\partial}{\partial Q_{mn}} \sum_k Q^\top_{ik} Q_{kj} = \sum_k \frac{\partial}{\partial Q_{mn}} (Q_{ki} Q_{kj})$$

$$= \sum_k \frac{\partial Q_{ki}}{\partial Q_{mn}} Q_{kj} + \sum_k Q_{ki} \frac{\partial Q_{kj}}{\partial Q_{mn}} = \sum_k \delta_{km} \delta_{in} Q_{kj} + \sum_k Q_{ki} \delta_{km} \delta_{jn}$$

$$= \delta_{in} Q_{mj} + \delta_{jn} Q_{mi}.$$

Additional Exercises

Exercise 3

Question: For a vector $\mathbf{w} \in \mathbb{R}^n$ and its Euclidean norm $\|\mathbf{w}\| := \sqrt{\mathbf{w}^\top \mathbf{w}}$, calculate $\frac{\partial \|\mathbf{w}\|}{\partial \mathbf{w}}$.
Answer: $\frac{\mathbf{w}}{\|\mathbf{w}\|}$.

Exercise 4

Question: Let $\mathbf{S}$ be a square matrix, find an expression for $\frac{\partial tr(\mathbf{S})}{\partial \mathbf{S}}$.
Answer: $\mathbf{I}$.

Exercise 3

Question: For a vector $\mathbf{w} \in \mathbb{R}^n$ and its Euclidean norm $\|\mathbf{w}\| := \sqrt{\mathbf{w}^\top \mathbf{w}}$, calculate $\frac{\partial \|\mathbf{w}\|}{\partial \mathbf{w}}$.

Solution: The norm is nothing but a function of n variables, i.e., $f(w_1, \ldots, w_n) = \sqrt{\sum_i w_i^2}$. As usual, we evaluate the derivative component-wise:

Exercise 3

Question: For a vector $\mathbf{w} \in \mathbb{R}^n$ and its Euclidean norm $\|\mathbf{w}\| := \sqrt{\mathbf{w}^\top \mathbf{w}}$, calculate $\frac{\partial \|\mathbf{w}\|}{\partial \mathbf{w}}$.

Solution: The norm is nothing but a function of n variables, i.e., $f(w_1, \ldots, w_n) = \sqrt{\sum_i w_i^2}$. As usual, we evaluate the derivative component-wise:

$$\frac{\partial \|\mathbf{w}\|}{\partial w_k} = \frac{\partial}{\partial w_k} \sqrt{\sum_i w_i^2} = \frac{1}{2\|\mathbf{w}\|} \frac{\partial}{\partial w_k} \sum_i w_i^2 = \frac{1}{2\|\mathbf{w}\|} \sum_i \frac{\partial}{\partial w_k} w_i^2$$
$$= \frac{1}{2\|\mathbf{w}\|} \sum_i 2 w_i \frac{\partial w_i}{\partial w_k} = \frac{1}{2\|\mathbf{w}\|} \sum_i 2 w_i \delta_{ik} = \frac{w_k}{\|\mathbf{w}\|}.$$

Exercise 4

Question: Let $\mathbf{S}$ be a square matrix, find an expression for $\frac{\partial \mathrm{tr}(\mathbf{S})}{\partial \mathbf{S}}$.

Solution: The object has a rank of 2 due to the matrix in the "denominator". We index and expand:

$$\left[\frac{\partial \mathrm{tr}(\mathbf{S})}{\partial \mathbf{S}}\right]_{ij} = \frac{\partial \mathrm{tr}(\mathbf{S})}{\partial S_{ij}} = \frac{\partial}{\partial S_{ij}} \sum_n S_{nn} = \sum_n \frac{\partial S_{nn}}{\partial S_{ij}}$$
$$= \sum_n \delta_{ni}\delta_{nj} = \delta_{ii}\delta_{ij} = \delta_{ij} = [\mathbf{I}]_{ij}.$$

Table of Contents

Chain Rule

Performing the chain rule over a matrix requires to sum over all its elements. Let there be a matrix $\mathbf{M}$ with some dependence on a scalar variable t. Then, for some well-defined and continuous function $g : \mathbb{R}^{m \times n} \to \mathbb{R}$, we have:

$$\frac{\partial g(\mathbf{M})}{\partial t} = \sum_{ij} \frac{\partial g(\mathbf{M})}{\partial M_{ij}} \frac{\partial M_{ij}}{\partial t}.$$

Einstein Summation Convention

You might encounter the notion of the *Einstein summation convention*. Simply put, this alleviates the need to write the summation sign at the front of an expression. The key to working with this convention is to look for **repeated indices**, which indicates that the index is a dummy index.

Einstein Summation Convention

You might encounter the notion of the *Einstein summation convention*. Simply put, this alleviates the need to write the summation sign at the front of an expression. The key to working with this convention is to look for **repeated indices**, which indicates that the index is a dummy index.

We will **not** use it in this course as it will not provide additional clarity in solving the problems. *You are expected to keep summation signs in all expressions in your work.*

Einstein Summation Convention

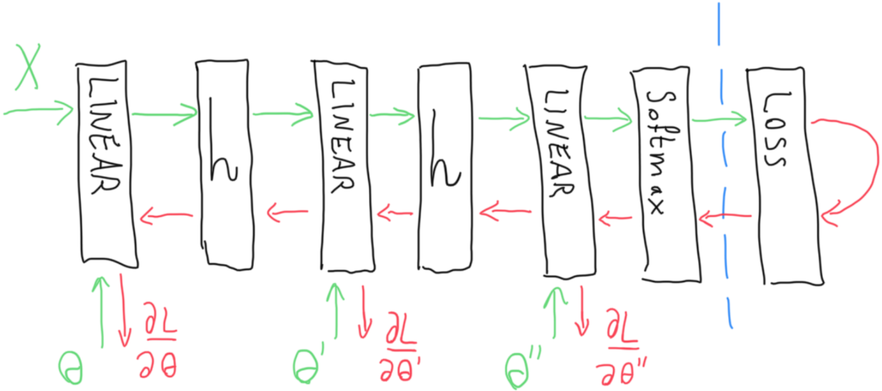
You might encounter the notion of the *Einstein summation convention*. Simply put, this alleviates the need to write the summation sign at the front of an expression. The key to working with this convention is to look for **repeated indices**, which indicates that the index is a dummy index.

We will **not** use it in this course as it will not provide additional clarity in solving the problems. *You are expected to keep summation signs in all expressions in your work.*

In NumPy, however, there is a handy implementation of `einsum` which could be useful for removing loops from your calculations.

Table of Contents

MLP Backpropagation



Question 1.1 a) Linear Module

Consider a linear module $\mathbf{Y} = \mathbf{X}\mathbf{W}^\top + \mathbf{B}$. The input and output features are $\mathbf{X}$ and $\mathbf{Y}$, respectively. Find closed form expressions for

$$\frac{\partial L}{\partial \mathbf{W}}, \frac{\partial L}{\partial \mathbf{b}}, \frac{\partial L}{\partial \mathbf{X}}$$

in terms of the gradients of the loss with respect to the output features $\frac{\partial L}{\partial \mathbf{Y}}$ provided by the next module during backpropagation.

Assume the gradients have the same shape as the object with respect to which is being differentiated. E.g. $\frac{\partial L}{\partial \mathbf{W}}$ should have the same shape as $\mathbf{W}$, $\frac{\partial L}{\partial \mathbf{b}}$ should be a row-vector just like $\mathbf{b}$ etc.

Question 1.1 b) Activation Module

Consider an *element-wise* activation function h. The activation module has input $\mathbf{X}$ and output $\mathbf{Y}$. I.e. $\mathbf{Y} = h(\mathbf{X}) \Rightarrow Y_{ij} = h(X_{ij})$. Find a closed form expression for

$$\frac{\partial L}{\partial \mathbf{X}}$$

in terms of the gradient of the loss with respect to the output features $\frac{\partial L}{\partial \mathbf{Y}}$ provided by the next module. Again, assume the gradient has the same shape as $\mathbf{X}$.

Question 1.1 c) Softmax and Loss Modules

1. Consider a module such that $Y_{ij} = [\text{softmax}(\mathbf{X})]_{ij}$, for input $\mathbf{X}$ and output $\mathbf{Y}$. Find a closed-form expression for $\frac{\partial L}{\partial \mathbf{X}}$ in terms of $\frac{\partial L}{\partial \mathbf{Y}}$. *[Hint: The answer might require using an all-ones matrix.]*
2. The loss module for the categorical cross entropy takes as input $\mathbf{X}$ and returns $L = \frac{1}{S}\sum_i L_i = -\frac{1}{S}\sum_{ik} T_{ik}\log(X_{ik})$. Find a closed form expression for $\frac{\partial L}{\partial \mathbf{X}}$. Write your answer in terms of matrix operations. *[Hint: You may use element-wise operations.]*
3. One can combine these into a single module with the following gradient: $\frac{\partial L}{\partial \mathbf{X}} = \alpha \mathbf{M}$. Find expressions for the positive scalar $\alpha \in \mathbb{R}^+$ and the matrix $\mathbf{M} \in \mathbb{R}^{S \times C}$ in terms of $\mathbf{Y}$, $\mathbf{T}$ and S.

Question 1.1 d) Residual Blocks

A residual connection has been introduced across a linear-activation-linear module. It adds $\mathbf{X} \in \mathbb{R}^{S \times F}$ to the output of the module element-wise.

1. Which constraints does the residual connection place on N_1 and N_2, the numbers of neurons in the two linear layers of the LAL module?
2. How does adding the residual connection change $\frac{\partial L}{\partial \mathbf{X}}$?
3. Briefly explain how your answer to (ii) improves the stability of training a deep neural network made up of many such residual blocks, also known as a *ResNet*.