Backpropagation - A Perspective of Matrix Calculus

Yu Chen

Technical University of Munich

November 20, 2023

Table of contents

0. Motivation & Background
1. Basis of Matrix Calculus
2. The Application of Matrix Calculus
3. Backpropagation in Matrix Form
4. Matrix-Form Backpropagation in Deep Learning Models

Guided Map

0. Motivation & Background
1. Basics of Matrix Calculus
2. The Application of Matrix Calculus
3. Backpropagation in Matrix Form
4. Matrix-Form Backpropagation in Deep Models

Motivation

Example: $L(W^1, \cdots, W^h, \cdots, W^H)$ is a categorical cross-entropy loss function of a Fully Connected Neural Network. $W^h \in \mathbb{R}^{D_h \times D_{h-1}}$ is the weight parameter matrix in the h^{th} layer.

- How to represent the backpropagation process in a closed-matrix form when we do $\min_{\{W^i\}_{i=1}^H} L(\cdot)$?

Background – Numerical Optimization

$$\min_{\vec{x}} f(\vec{x}), \quad \vec{x} \in \mathbb{R}^D \quad \text{and} \quad f(\vec{x}) \in \mathbb{R}. \tag{1}$$

solved by:

$$\vec{x}_{t+1} := \vec{x}_t + \alpha_t \vec{d}_t \tag{2}$$

where $\|\vec{d}\|_2 = 1$ and α_t is the step size at $\vec{x}_t$.

How to decide $\vec{d}_t$ and α_t?

Background – Intuition Behind Gradient Descent

$$\min_{\vec{d}_t, \alpha_t} f(\vec{x}_t + \alpha_t \vec{d}_t) \quad \text{s.t.} \quad \|\vec{d}_t\|_2^2 = 1.$$

- $f(\vec{x}_t + \alpha_t \vec{d}_t) = f(\vec{x}_t) + \alpha_t \vec{d}_t^T \nabla_{\vec{x}_t} f(\vec{x}) + o(\alpha_t)$
- Gradient Descent $f(\vec{x}_t + \alpha_t \vec{d}_t) \approx f(\vec{x}_t) + \alpha_t \vec{d}_t^T \nabla_{\vec{x}_t} f(\vec{x})$
- $$\vec{d}_t^* = \text{argmin}_{\vec{d}_t} f(\vec{x}_t) + \alpha_t \vec{d}_t^T \nabla_{\vec{x}_t} f(\vec{x}) \quad \text{s.t.} \quad \|\vec{d}_t\|_2^2 = 1 \qquad (3)$$
- $$\alpha_t^* = \text{argmin}_{\alpha_t} f(\vec{x}_t + \alpha_t \vec{d}_t^*) \qquad (4)$$

Background – Intuition Behind Gradient Descent

- α_t^*: Line Search Methods.
- $\vec{d}_t^*$: Lagrange Multiplier Method.

$$
\begin{aligned}
\vec{d}_t^* &= \operatorname{argmin}_{\vec{d}_t} L(\vec{d}_t, \lambda_t) \\
&:= f(\vec{x}_t) + \alpha_t \vec{d}_t^T \nabla_{\vec{x}_t} f(\vec{x}) + \lambda_t (\|\vec{d}_t\|_2^2 - 1) \\
&= -\frac{\nabla_{\vec{x}_t} f(\vec{x})}{\|\nabla_{\vec{x}_t} f(\vec{x})\|_2}
\end{aligned}
$$

Background – Gradient Descent Algorithm

Step 1: initialize $\vec{x}_0$;

Step 2: calculate $\vec{d}_t^* = -\frac{\nabla_{\vec{x}_t} f(\vec{x})}{\|\nabla_{\vec{x}_t} f(\vec{x})\|_2}$;

Step 3: calculate α_t^* using line search or set $\alpha_t^* = 0.0001$;

Step 4: $\vec{x}_{t+1} := \vec{x}_t - \alpha_t^* \vec{d}_t^*$;

Step 5: repeat Step 2, 3, and 4 to approximate $\vec{x}^*$.

Backpropagation is applied to compute $\nabla_{\vec{x}} f(\vec{x})$ in machine learning especially when $f(\cdot)$ is complex.

Guided Map

0. Motivation & Background
1. Basics of Matrix Calculus
2. The Application of Matrix Calculus
3. Backpropagation in Matrix Form
4. Matrix-Form Backpropagation in Deep Models

Gradient Computation - Example

Example 1-1

$f(\vec{x}, A, \vec{y}) = \vec{x}^T A\vec{y}$ where $A \in \mathbb{R}^{M\times N}$,$\vec{x} \in \mathbb{R}^M$, and $\vec{y} \in \mathbb{R}^N$. How to derive gradients $\nabla_{\vec{x}} f(\cdot)$, $\nabla_{\vec{y}} f(\cdot)$, and $\nabla_A f(\cdot)$ in closed forms?

1. Intuitively, $\nabla_{\vec{x}} f(\cdot) = A\vec{y}$ and $\nabla_{\vec{y}} f(\cdot) = A^T\vec{x}$, but why and how?
2. How to derive $\nabla_A f(\cdot)$ systematically rather than intuitively?
3. If $\vec{y} = \sigma \odot (B^{-1}\vec{z} + \vec{b})$ where $B^{-1} \in \mathbb{R}^{N\times N}$ is the inverse of B, $\vec{z}, \vec{b} \in \mathbb{R}^N$, and $\sigma \odot (\vec{y}) := [\sigma(y_0), \sigma(y_1), \cdots, \sigma(y_N)]^T$ is an element-wise Sigmoid function with $\sigma(y_i) := \frac{1}{1+\exp(-y_i)}$, how to derive $\nabla_B f(\cdot)$?

Definition of Gradient

Suppose $f_1(x) : \mathbb{R} \to \mathbb{R}$, $f_2(\vec{x}) : \mathbb{R}^D \to \mathbb{R}$, and $f_3(X) : \mathbb{R}^{M\times N} \to \mathbb{R}$ where $x \in \mathbb{R}$, $\vec{x} := [x_1, \cdots, x_D]^T \in \mathbb{R}^D$, and $X := [X_{ij}] \in \mathbb{R}^{M\times N}$.

Definition (1-1)

The derivative of $f_1(x)$ w.r.t. x is defined as $f_1'(x) = \frac{\partial f_1(x)}{\partial x} := \lim\limits_{\Delta x\to 0} \frac{f_1(x+\Delta x)-f_1(x)}{\Delta x}$.

The gradient of $f_2(\vec{x})$ w.r.t. $\vec{x}$ is defined as $\nabla_{\vec{x}} f_2(\cdot) = \frac{\partial f_2}{\partial \vec{x}} := [\frac{\partial f_2}{\partial x_1}, \cdots, \frac{\partial f_2}{\partial x_D}]^T \in \mathbb{R}^D$

The gradient of $f_3(X)$ w.r.t. X is defined as $\nabla_X f_3(\cdot) = \frac{\partial f_3}{\partial X} := [\frac{\partial f_3}{\partial X_{ij}}] \in \mathbb{R}^{M\times N}$

where $\frac{\partial f_3}{\partial X_{ij}}$ is the partial derivative of f_3 w.r.t. the matrix entry X_{ij} defined as $\frac{\partial f_3}{\partial X_{ij}} := \lim\limits_{\Delta X_{ij}\to 0} \frac{f_3(X_{ij}+\Delta X_{ij})-f_3(X_{ij})}{\Delta X_{ij}}$.

Differential

Differential is used to derive the gradient of a function w.r.t. a vector or a matrix in a closed-matrix form .

Definition (1-2)

$$\partial f_1 := f_1'(x)\,\partial x$$

$$\partial f_2 := \sum_{i=1}^{D} \frac{\partial f_2}{\partial x_i}\,\partial x_i = \left(\frac{\partial f_2}{\partial \vec{x}}\right)^T \partial \vec{x}$$

$$\partial f_3 := \sum_{i=1}^{M}\sum_{j=1}^{N} \frac{\partial f_3}{\partial X_{ij}}\,\partial X_{ij} = \mathrm{Tr}\left(\frac{\partial f_3}{\partial X}^T \partial X\right)$$

where Tr() represents the trace operation defined as

$\mathrm{Tr}(A) := \sum_{i=1}^{N} A_{ii}$ where $A \in \mathbb{R}^{N\times N}$.

→ the laws of matrix differential operations and properties of trace help!

Laws of Matrix Differential

Theorem (1-1)

Assume $A, B \in \mathbb{R}^{M\times N}$, $C \in \mathbb{R}^{N\times M}$, and $D \in \mathbb{R}^{N\times N}$ is invertible. $|D|$ is the determinant and D^* is the adjugate matrix.

1. $\partial(A \pm B) = \partial A \pm \partial B$
2. $\partial(AC) = (\partial A)C + A(\partial C)$
3. $\partial(A^T) = (\partial A)^T$
4. $\partial \mathrm{Tr}(D) = \mathrm{Tr}(\partial D)$
5. $\partial D^{-1} = -D^{-1}(\partial D)D^{-1}$
6. $\partial|D| = \frac{1}{N}\mathrm{Tr}(D^* \partial D) = \frac{1}{N}|D|\mathrm{Tr}(D^{-1} \partial D)$
7. $\partial(A \odot B) = (\partial A) \odot B + A \odot \partial B$
8. $\partial f \odot (A) = f'(A) \odot \partial A$ where $f'(A) := [\frac{\partial f}{\partial A_{ij}}] \in \mathbb{R}^{M\times N}$

Properties of Trace

Theorem (1-2)

Assume $a \in \mathbb{R}$ is a real number, matrices $A, B \in \mathbb{R}^{N\times N}$, and $C, D, F \in \mathbb{R}^{M\times N}$.

1. $a = \mathrm{Tr}(a)$
2. $\mathrm{Tr}(A^T) = \mathrm{Tr}(A)$
3. $\mathrm{Tr}(A \pm B) = \mathrm{Tr}(A) \pm \mathrm{Tr}(B)$
4. $\mathrm{Tr}(CD^T) = \mathrm{Tr}(D^T C)$
5. $\mathrm{Tr}(C^T(D \odot F)) = \mathrm{Tr}((C \odot D)^T F)$

With the laws of matrix differential and trace properties, we can derive some closed-matrix form gradients for some functions.

Guided Map

1. Motivation & Background
2. Basics of Matrix Calculus
3. The Application of Matrix Calculus
4. Backpropagation in Matrix Form
5. Matrix-Form Backpropagation in Deep Models

Simple Example

Example 2-1-1

$f(\vec{x}, A, \vec{y}) = \vec{x}^T A\vec{y}$ where $A \in \mathbb{R}^{M\times N}$,$\vec{x} \in \mathbb{R}^M$, and $\vec{y} \in \mathbb{R}^N$. Derive $\nabla_{\vec{x}} f(\cdot)$ in its closed-matrix form.

Solution:

1. $\partial f = (\partial\vec{x}^T)A\vec{y} + \vec{x}^T\,\partial(A\vec{y}) = (\partial\vec{x}^T)A\vec{y}$
2. $\mathrm{Tr}(\partial f) = \mathrm{Tr}((\partial\vec{x}^T)A\vec{y}) = \mathrm{Tr}(\vec{y}A^T\,\partial\vec{x}) = \mathrm{Tr}((A\vec{y})^T\,\partial\vec{x})$
3. $\partial f = \mathrm{Tr}((A\vec{y})^T\,\partial\vec{x})$
4. So, $\nabla_{\vec{x}} f(\cdot) = \frac{\partial f}{\partial\vec{x}} = A\vec{y}$ compared with Definition 2.

Simple Example

Example 2-1-2

$f(\vec{x}, A, \vec{y}) = \vec{x}^T A \vec{y}$ where $A \in \mathbb{R}^{M \times N}$,$\vec{x} \in \mathbb{R}^M$, and $\vec{y} \in \mathbb{R}^N$. Derive $\nabla_{\vec{y}} f(\cdot)$ in its closed-matrix form.

Solution:

1. $\partial f = (\partial \vec{x}^T A)\vec{y} + \vec{x}^T A\, \partial(\vec{y}) = (\vec{x}^T A)\, \partial \vec{y}$
2. $\mathrm{Tr}(\partial f) = \mathrm{Tr}((\vec{x}^T A)\, \partial \vec{y}) = \mathrm{Tr}((A^T \vec{x})^T\, \partial \vec{y})$
3. $\partial f = \mathrm{Tr}((A^T \vec{x})^T\, \partial \vec{y})$
4. So, $\nabla_{\vec{y}} f(\cdot) = \frac{\partial f}{\partial \vec{y}} = A^T \vec{x}$ compared with Definition 2.

Simple Example

Example 2-1-3

$f(\vec{x}, A, \vec{y}) = \vec{x}^T A \vec{y}$ where $A \in \mathbb{R}^{M\times N}$,$\vec{x} \in \mathbb{R}^M$, and $\vec{y} \in \mathbb{R}^N$. Derive $\nabla_A f(\cdot)$ in its closed-matrix form.

Solution:

1. $\partial f = \vec{x}^T \, \partial(A)\vec{y}$
2. $\mathrm{Tr}(\partial f) = \mathrm{Tr}(\vec{x}^T \, \partial(A)\vec{y}) = \mathrm{Tr}(\vec{y}\vec{x}^T \, \partial A) = \mathrm{Tr}((\vec{x}\vec{y}^T)^T \, \partial A)$
3. $\partial f = \mathrm{Tr}((\vec{x}\vec{y}^T)^T \, \partial A)$
4. So, $\nabla_A f(\cdot) = \frac{\partial f}{\partial A} = \vec{x}\vec{y}^T$ compared with Definition 2.

Linear Regression

Example 2-2

Suppose $L(\vec{w}) = ||X\vec{w} - \vec{y}||^2$ where $\vec{y} \in \mathbb{R}^M$, $\vec{w} \in \mathbb{R}^N$, and $X \in \mathbb{R}^{M\times N}$. Solving $\min_{\vec{w}} L(\vec{w})$ requires $\nabla_{\vec{w}} L(\vec{w}) = 0$, so we need to derive the closed-matrix form of $\nabla_{\vec{w}} L(\vec{w})$.

Solution 2-2:

1. Apply laws of matrix differential:

$$\begin{aligned}\partial L &= \partial((X\vec{w} - \vec{y})^T (X\vec{w} - \vec{y})) \\ &= \partial(X\vec{w} - \vec{y})^T (X\vec{w} - \vec{y}) + (X\vec{w} - \vec{y})^T \, \partial(X\vec{w} - \vec{y}) \\ &= (X\,\partial w)^T (X\vec{w} - \vec{y}) + (X\vec{w} - \vec{y})^T X\,\partial\vec{w}\end{aligned}$$

2. Add trace operation and apply properties of trace:

$$\mathrm{Tr}(\partial L) = \mathrm{Tr}((X\,\partial w)^T (X\vec{w} - \vec{y}) + (X\vec{w} - \vec{y})^T X\,\partial\vec{w})$$

Linear Regression

Solution 2-2:

2. Apply properties of trace

$$\begin{aligned}
\mathrm{Tr}(\partial \mathrm{L}) &= \mathrm{Tr}((\mathrm{X}\,\partial\vec{\mathrm{w}})^{\mathrm{T}}(\mathrm{X}\vec{\mathrm{w}}-\vec{\mathrm{y}}) + (\mathrm{X}\vec{\mathrm{w}}-\vec{\mathrm{y}})^{\mathrm{T}}\mathrm{X}\,\partial\vec{\mathrm{w}}) \\
&= \mathrm{Tr}((\mathrm{X}\,\partial\vec{\mathrm{w}})^{\mathrm{T}}(\mathrm{X}\vec{\mathrm{w}}-\vec{\mathrm{y}})) + \mathrm{Tr}((\mathrm{X}\vec{\mathrm{w}}-\vec{\mathrm{y}})^{\mathrm{T}}\mathrm{X}\,\partial\vec{\mathrm{w}})) \\
&= \mathrm{Tr}((\partial\vec{\mathrm{w}})^{\mathrm{T}}\mathrm{X}^{\mathrm{T}}(\mathrm{X}\vec{\mathrm{w}}-\vec{\mathrm{y}})) + \mathrm{Tr}((\mathrm{X}\vec{\mathrm{w}}-\vec{\mathrm{y}})^{\mathrm{T}}\mathrm{X}\,\partial\vec{\mathrm{w}}) \\
&= \mathrm{Tr}((\mathrm{X}\vec{\mathrm{w}}-\vec{\mathrm{y}})^{\mathrm{T}}\mathrm{X}\,\partial\vec{\mathrm{w}}) + \mathrm{Tr}((\mathrm{X}\vec{\mathrm{w}}-\vec{\mathrm{y}})^{\mathrm{T}}\mathrm{X}\,\partial\vec{\mathrm{w}}) \\
&= \mathrm{Tr}(2(\mathrm{X}\vec{\mathrm{w}}-\vec{\mathrm{y}})^{\mathrm{T}}\mathrm{X}\,\partial\vec{\mathrm{w}}) \\
&= \mathrm{Tr}(2(\mathrm{X}^{\mathrm{T}}(\mathrm{X}\vec{\mathrm{w}}-\vec{\mathrm{y}}))^{\mathrm{T}}\,\partial\vec{\mathrm{w}})
\end{aligned}$$

3. So, $\nabla_{\vec{\mathrm{w}}}\mathrm{L} = 2\mathrm{X}^{\mathrm{T}}(\mathrm{X}\vec{\mathrm{w}}-\vec{\mathrm{y}})$

Maximum Likelihood Estimation

Example 2-3

Suppose the data set $\{\vec{x}_i\}_{i=1}^N$ with $\vec{x}_i \in \mathbb{R}^D$ where

$\vec{x}_i \sim P(\vec{x}_i|\mu, \Sigma) := (2\pi)^{-\frac{D}{2}}|\Sigma|^{-\frac{1}{2}}e^{-\frac{1}{2}(\vec{x}_i-\vec{\mu})^T\Sigma^{-1}(\vec{x}_i-\vec{\mu})}$. Then,

$$\max_{\vec{\mu},\Sigma} \prod_{i=1}^{N} P(\vec{x}_i|\vec{\mu}, \Sigma) \iff$$

$$\max_{\vec{\mu},\Sigma} -\frac{ND}{2}\log(2\pi) - \frac{N}{2}\log(|\Sigma|) - \frac{1}{2}\sum_{i=1}^{N}(\vec{x}_i - \vec{\mu})^T\Sigma^{-1}(\vec{x}_i - \vec{\mu}) \iff$$

$$\min_{\vec{\mu},\Sigma} L(\vec{\mu}, \Sigma) = N\log(|\Sigma|) + \sum_{i=1}^{N}(\vec{x}_i - \vec{\mu})^T\Sigma^{-1}(\vec{x}_i - \vec{\mu}).$$

The minimization of L w.r.t. $\vec{\mu}$ is independent to that of Σ.

The MLE of Σ requires $\nabla_\Sigma L = 0$, so we need to derive $\nabla_\Sigma L$.

Maximum Likelihood Estimation

Example 2-3

$\min_{\vec{\mu},\Sigma} L(\vec{\mu}, \Sigma) = N\log(|\Sigma|) + \sum_{i=1}^{N}(\vec{x}_i - \vec{\mu})^T\Sigma^{-1}(\vec{x}_i - \vec{\mu})$. Derive $\nabla_{\Sigma}L$.

Solution 2-3:

1. Apply laws of matrix differential:

$$\begin{aligned}
\partial L &= N\,\partial(\log(|\Sigma|)) + \sum_{i=1}^{N}(\vec{x}_i - \vec{\mu})^T\,\partial(\Sigma^{-1})(\vec{x}_i - \vec{\mu}) \\
&= N(|\Sigma|)^{-1}\,\partial|\Sigma| - \sum_{i=1}^{N}(\vec{x}_i - \vec{\mu})^T\Sigma^{-1}\,\partial(\Sigma)\Sigma^{-1}(\vec{x}_i - \vec{\mu})
\end{aligned}$$

Maximum Likelihood Estimation

Solution 2-3:

2. Add trace operation and apply properties of trace:

$$\mathrm{Tr}(\partial L) = \mathrm{Tr}(N(|\Sigma|)^{-1}\,\partial|\Sigma|) -$$
$$\mathrm{Tr}(\sum_{i=1}^{N}(\vec{x}_i - \vec{\mu})^T\Sigma^{-1}\,\partial(\Sigma)\Sigma^{-1}(\vec{x}_i - \vec{\mu}))$$
$$= N/D|\Sigma|^{-1}|\Sigma|\mathrm{Tr}(\Sigma^{-1}\,\partial\Sigma) -$$
$$\mathrm{Tr}(\sum_{i=1}^{N}\Sigma^{-1}(\vec{x}_i - \vec{\mu})(\vec{x}_i - \vec{\mu})^T\Sigma^{-1}\,\partial\Sigma)$$
$$= \mathrm{Tr}(N/D\Sigma^{-1}\,\partial\Sigma - \sum_{i=1}^{N}\Sigma^{-1}(\vec{x}_i - \vec{\mu})(\vec{x}_i - \vec{\mu})^T\Sigma^{-1}\,\partial\Sigma)$$

3. So, $\nabla_\Sigma L = N/D\Sigma^{-1} - \sum_{i=1}^{N}\Sigma^{-1}(\vec{x}_i - \vec{\mu})(\vec{x}_i - \vec{\mu})^T\Sigma^{-1}$

Guided Map

0. Motivation & Background
1. Basics of Matrix Calculus
2. The Application of Matrix Calculus
3. Backpropagation in Matrix Form
4. Matrix-Form Backpropagation in Deep Models

Backpropagation in Matrix Form

Example 3-1

$f = \vec{x}^T A \vec{y}$ with $\vec{y} = \sigma \odot (\vec{h})$ and $\vec{h} = B^{-1}\vec{z} + \vec{b}$ where $\vec{y} \in \mathbb{R}^N$, $A \in \mathbb{R}^{M\times N}$,$\vec{x} \in \mathbb{R}^M$, $B^{-1} \in \mathbb{R}^{N\times N}$,$\vec{z}, \vec{b} \in \mathbb{R}^N$, and $\sigma \odot (\vec{y}) := [\sigma(y_0), \cdots, \sigma(y_N)]^T$ is an element-wise Sigmoid function. Derive $\nabla_B f(\cdot)$ in its closed-matrix form.

Solution:

1. $\partial f = \vec{x}^T A\, \partial\vec{y}$. After adding $\mathrm{Tr}(\cdot)$ operation on both side:
2. $\mathrm{Tr}(\partial f) = \mathrm{Tr}(\vec{x}^T A\, \partial(\vec{y})) = \mathrm{Tr}((A^T\vec{x})^T\, \partial\vec{y})$ leads to:
   2.1. $\nabla_{\vec{y}} f = A^T\vec{x}$
3. $\mathrm{Tr}(\partial f) = \mathrm{Tr}(\nabla_{\vec{y}} f^T\, \partial(\vec{y}))$
   3.1. $\partial(\vec{y}) = \partial\{\sigma \odot (\vec{h})\} = \sigma'(\vec{h}) \odot \partial\vec{h}$
   3.2. $\mathrm{Tr}(\partial f) = \mathrm{Tr}(\nabla_{\vec{y}} f^T(\sigma'(\vec{h}) \odot \partial\vec{h})) = \mathrm{Tr}((\nabla_{\vec{y}} f \odot \sigma'(\vec{h}))^T\, \partial\vec{h})$ leads to
   3.3. $\nabla_{\vec{h}} f = \nabla_{\vec{y}} f \odot \sigma'(\vec{h})$

Backpropagation in Matrix Form

Solution:

4. $\mathrm{Tr}(\nabla_{\mathrm{h}}\mathrm{f}) = \mathrm{Tr}(\nabla_{\vec{\mathrm{h}}}\mathrm{f}^{\mathrm{T}}\,\partial\vec{\mathrm{h}})$
   4.1. $\partial\vec{\mathrm{h}} = \partial(\mathrm{B}^{-1}\vec{\mathrm{z}} + \vec{\mathrm{b}}) = \partial(\mathrm{B}^{-1})\vec{\mathrm{z}} = -\mathrm{B}^{-1}(\partial\mathrm{B})\mathrm{B}^{-1}\vec{\mathrm{z}}$
   4.2. Substituting 4.1. to 4. leads to:

$$\begin{aligned}\mathrm{Tr}(\nabla_{\vec{\mathrm{h}}}\mathrm{f}) &= \mathrm{Tr}(-\nabla_{\vec{\mathrm{h}}}\mathrm{f}^{\mathrm{T}}\mathrm{B}^{-1}(\partial\mathrm{B})\mathrm{B}^{-1}\vec{\mathrm{z}}) \\ &= \mathrm{Tr}(-\mathrm{B}^{-1}\vec{\mathrm{z}}\nabla_{\vec{\mathrm{h}}}\mathrm{f}^{\mathrm{T}}\mathrm{B}^{-1}\,\partial\mathrm{B}) \\ &= \mathrm{Tr}((-\mathrm{B}^{-\mathrm{T}}\nabla_{\vec{\mathrm{h}}}\mathrm{f}\vec{\mathrm{z}}^{\mathrm{T}}\mathrm{B}^{-\mathrm{T}})^{\mathrm{T}}\,\partial\mathrm{B})\end{aligned}$$

   4.3. $\nabla_{\mathrm{B}}\mathrm{f} = -\mathrm{B}^{-\mathrm{T}}\nabla_{\vec{\mathrm{h}}}\mathrm{f}\vec{\mathrm{z}}^{\mathrm{T}}\mathrm{B}^{-\mathrm{T}}$

We can finally combine the backpropagation equations 2.1., 3.3., and 4.3. to derive the closed-matrix form of
$\nabla_{\mathrm{B}}\mathrm{f} = -\mathrm{B}^{-\mathrm{T}}\mathrm{A}^{\mathrm{T}}\vec{\mathrm{x}} \odot (\sigma \odot (\vec{\mathrm{h}})(1 - \sigma \odot (\vec{\mathrm{h}})))\vec{\mathrm{z}}^{\mathrm{T}}\mathrm{B}^{-\mathrm{T}}$

Guided Map

0. Motivation & Background
1. Basics of Matrix Calculus
2. The Application of Matrix Calculus
3. Backpropagation in Matrix Form
4. Matrix-Form Backpropagation in Deep Models
    4.1. Logistics Regression
    4.2. Fully Connected Neural Networks
    4.3. Convolutional Neural Networks
    4.4. Future Work

Logistics Regression

Example 4-1

Suppose $\vec{h} = W\vec{x}$, $L = -\vec{y}^T \log \odot (\sigma(\vec{h}))$ where $\vec{x} \in \mathbb{R}^N$, $W \in \mathbb{R}^{M\times N}$, $\vec{y} \in \{0,1\}^M : \sum_{i=1}^{M} y_i = 1$, $\sigma(\vec{h}) = \frac{\exp\odot(\vec{h})}{\vec{1}^T\exp\odot(\vec{h})}$ with $\vec{1} = [1, 1, \cdots, 1]^T$ and $\dim(\vec{1}) = M$. Formulate the backpropagation process $L \rightarrow \nabla_{\vec{h}} L \rightarrow \nabla_W L$ in a closed-matrix form.

Solution 4-1-1:

1. $\log \odot (\sigma(\vec{h})) = \log \odot \left(\frac{\exp\odot(\vec{h})}{\vec{1}^T\exp\odot(\vec{h})}\right) = \vec{h} - \log(\vec{1}^T \exp \odot (\vec{h})) \cdot \vec{1}$
2. So, $L = -\vec{y}^T \underbrace{\left(\vec{h} - \log(\vec{1}^T \exp \odot (\vec{h})) \cdot \vec{1}\right)}_{\vec{z}} = -\vec{y}^T \vec{z}$
3. $\partial L = \mathrm{Tr}(-\vec{y}^T\, \partial \vec{z})$

Logistics Regression

Solution 4-1-2:

3. $\partial \mathrm{L} = \mathrm{Tr}(-\vec{\mathrm{y}}^{\mathrm{T}}\, \partial \vec{\mathrm{z}})$

   3.1. Apply laws of matrix differential

$$
\begin{aligned}
\partial(\vec{\mathrm{z}}) &= \partial \vec{\mathrm{h}} - \partial(\log(\vec{1}^{\mathrm{T}} \exp \odot (\vec{\mathrm{h}})) \cdot \vec{1}) \\
&= \partial \vec{\mathrm{h}} - \frac{1}{\vec{1}^{\mathrm{T}} \exp \odot (\vec{\mathrm{h}})} \vec{1}^{\mathrm{T}}\, \partial(\exp \odot (\vec{\mathrm{h}}) \cdot \vec{1}) \\
&= \partial \vec{\mathrm{h}} - \vec{1} \cdot \frac{1}{\vec{1}^{\mathrm{T}} \exp \odot (\vec{\mathrm{h}})} \vec{1}^{\mathrm{T}}\, \partial(\exp \odot (\vec{\mathrm{h}})) \\
&= \partial \vec{\mathrm{h}} - \vec{1} \cdot \frac{1}{\vec{1}^{\mathrm{T}} \exp \odot (\vec{\mathrm{h}})} \vec{1}^{\mathrm{T}} (\exp'(\vec{\mathrm{h}}) \odot (\partial \vec{\mathrm{h}}))
\end{aligned}
$$

Logistics Regression

Solution 4-1-3:

3. $\partial \mathrm{L} = \mathrm{Tr}(-\vec{\mathrm{y}}^{\mathrm{T}}\, \partial \vec{\mathrm{z}})$
   3.1. $\partial(\vec{\mathrm{z}}) = \partial \vec{\mathrm{h}} - \vec{1} \cdot \frac{1}{\vec{1}^{\mathrm{T}} \exp \odot (\vec{\mathrm{h}})} \vec{1}^{\mathrm{T}} (\exp'(\vec{\mathrm{h}}) \odot (\partial \vec{\mathrm{h}}))$
   3.2. Add trace and apply its properties

$$
\begin{aligned}
\mathrm{Tr}(\partial \mathrm{L}) = & -\mathrm{Tr}(\vec{\mathrm{y}}^{\mathrm{T}}\, \partial \vec{\mathrm{h}}) + \\
& \mathrm{Tr}(\underbrace{\vec{\mathrm{y}}^{\mathrm{T}} \vec{1}}_{=1} \cdot \frac{1}{\vec{1}^{\mathrm{T}} \exp \odot (\vec{\mathrm{h}})} \underbrace{\vec{1}^{\mathrm{T}} (\exp'(\vec{\mathrm{h}}) \odot (\partial \vec{\mathrm{h}})))}_{\text{Trace property 5}} \\
& = -\mathrm{Tr}(\vec{\mathrm{y}}^{\mathrm{T}}\, \partial \vec{\mathrm{h}}) + \frac{1}{\vec{1}^{\mathrm{T}} \exp \odot (\vec{\mathrm{h}})} \mathrm{Tr}((\underbrace{\vec{1} \odot \exp'(\vec{\mathrm{h}})}_{=\exp \odot (\vec{\mathrm{h}})})^{\mathrm{T}}\, \partial \vec{\mathrm{h}}) \\
& = -\mathrm{Tr}(\vec{\mathrm{y}}^{\mathrm{T}}\, \partial \vec{\mathrm{h}}) + \mathrm{Tr}(\frac{(\exp \odot (\vec{\mathrm{h}}))^{\mathrm{T}}}{\vec{1}^{\mathrm{T}} \exp \odot (\vec{\mathrm{h}})}\, \partial \vec{\mathrm{h}})
\end{aligned}
$$

Logistics Regression

Solution 4-1-4:

3. $\partial \mathrm{L} = \mathrm{Tr}(-\vec{\mathrm{y}}^{\mathrm{T}}\, \partial \vec{\mathrm{z}})$
   3.2. Add trace and apply its properties

$$\mathrm{Tr}(\partial \mathrm{L}) = -\mathrm{Tr}(\vec{\mathrm{y}}^{\mathrm{T}}\, \partial \vec{\mathrm{h}}) + \mathrm{Tr}(\underbrace{\frac{(\exp \odot (\vec{\mathrm{h}}))^{\mathrm{T}}}{\vec{1}^{\mathrm{T}} \exp \odot (\vec{\mathrm{h}})}}_{=\sigma(\vec{\mathrm{h}})^{\mathrm{T}}}\, \partial \vec{\mathrm{h}})$$

$$= \mathrm{Tr}((\sigma(\vec{\mathrm{h}}) - \vec{\mathrm{y}})^{\mathrm{T}}\, \partial \vec{\mathrm{h}})$$

   3.3. So, $\nabla_{\vec{\mathrm{h}}} \mathrm{L} = \sigma(\vec{\mathrm{h}}) - \vec{\mathrm{y}}$
4. $\partial \mathrm{L} = \mathrm{Tr}(\nabla_{\vec{\mathrm{h}}} \mathrm{L}^{\mathrm{T}}\, \partial \vec{\mathrm{h}})$
   4.1. $\partial \vec{\mathrm{h}} = \partial(\mathrm{W})\vec{\mathrm{x}}$
   4.2. $\partial \mathrm{L} = \mathrm{Tr}(\nabla_{\vec{\mathrm{h}}} \mathrm{L}^{\mathrm{T}}\, \partial(\mathrm{W})\vec{\mathrm{x}}) = \mathrm{Tr}((\nabla_{\vec{\mathrm{h}}} \mathrm{L} \vec{\mathrm{x}}^{\mathrm{T}})^{\mathrm{T}}\, \partial \mathrm{W})$
   4.3. So, $\nabla_{\mathrm{W}} \mathrm{L} = \nabla_{\vec{\mathrm{h}}} \mathrm{L} \vec{\mathrm{x}}^{\mathrm{T}}$

Fully Connected Neural Networks

Example 4-2

1. The loss $L = -\vec{y}^T \log\odot(\sigma_2(\vec{h}_2))$ where $\sigma_2(\vec{h}_2) = \frac{\exp\odot(\vec{h}_2)}{\vec{1}^T \exp\odot(\vec{h}_2)}$, $\vec{1} = [1, \cdots, 1]^T \in \{1\}^{M_2}$, $\vec{y} \in \{0,1\}^{M_2} : \sum_{i=1}^{M_2} y_i = 1$;
2. 2nd layer $\vec{h}_2 = W_2\vec{z}_1 + \vec{b}_2$ where $W_2 \in \mathbb{R}^{M_2 \times M_1}$,$\vec{b}_2 \in \mathbb{R}^{M_2}$, $\vec{z}_1 = \sigma_1 \odot (\vec{h}_1)$ and $\sigma_1 \odot (\cdot)$ is an elementwise Sigmoid;
3. 1st layer $\vec{h}_1 = W_1\vec{x} + \vec{b}_1$ where $\vec{x} \in \mathbb{R}^{M_0}$, $W_1 \in \mathbb{R}^{M_1 \times M_0}$,$\vec{b}_1 \in \mathbb{R}^{M_1}$

Formulate the backpropagation process
$L \to \nabla_{\vec{h}_2} L \to \nabla_{\vec{z}_1} L \to \nabla_{\vec{h}_1} L \to \nabla_{W_1} L$ in a closed-matrix form.

Fully Connected Neural Networks

Example 4-2

Formulate the backpropagation process
$L \to \nabla_{\vec{h}_2} L \to \nabla_{\vec{z}_1} L \to \nabla_{\vec{h}_1} L \to \nabla_{W_1} L$ in a closed-matrix form.

Solution 4-2-1:

1. Example 4-1 indicates that $\nabla_{\vec{h}_2} L = \sigma_2(\vec{h}_2) - \vec{y}$
   1.1. $\partial L = Tr(\nabla_{\vec{h}} L^T \, \partial \vec{h}_2)$
   1.2. $\partial \vec{h}_2 = W_2 \, \partial(\vec{z}_1)$
   1.3. $\partial L = Tr((W_2^T \nabla_{\vec{h}} L)^T \, \partial \vec{z}_1)$
   1.4. So, $\nabla_{\vec{z}_1} L = W_2^T \nabla_{\vec{h}} L$
2. $\partial L = Tr(\nabla_{\vec{z}_1} L^T \, \partial \vec{z}_1)$

Fully Connected Neural Networks

Solution 4-2-2:

1. $\nabla_{\vec{h}_2} L = \sigma_2(\vec{h}_2) - \vec{y}$
2. $\nabla_{\vec{z}_1} L = W_2^T \nabla_{\vec{h}} L$
3. $\partial L = \mathrm{Tr}(\nabla_{\vec{z}_1} L^T \, \partial \vec{z}_1)$
   - 3.1. $\partial \vec{z}_1 = \partial \sigma_1 \odot (\vec{h}_1) = \sigma_1'(\vec{h}_1) \odot \partial \vec{h}_1$ where $\sigma_1'(\vec{h}_1) = \sigma_1 \odot (\vec{h}_1) \odot (1 - \sigma_1 \odot (\vec{h}_1))$
   - 3.2. $\partial L = \mathrm{Tr}(\nabla_{\vec{z}_1} L^T (\sigma_1'(\vec{h}_1) \odot \partial \vec{h}_1)) = \mathrm{Tr}((\nabla_{\vec{z}_1} L \odot \sigma_1'(\vec{h}_1))^T \, \partial \vec{h}_1)$
   - 3.3. So, $\nabla_{\vec{h}_1} L = \nabla_{\vec{z}_1} L \odot \sigma_1'(\vec{h}_1)$
4. $\partial L = \mathrm{Tr}(\nabla_{\vec{h}_1} L^T \, \partial \vec{h}_1)$
   - 4.1. $\partial \vec{h}_1 = (\partial W_1)\vec{x}$
   - 4.2. $\partial L = \mathrm{Tr}(\nabla_{\vec{h}_1} L^T (\partial W_1)\vec{x}) = \mathrm{Tr}((\nabla_{\vec{h}_1} L \vec{x}^T)^T \, \partial W_1)$
   - 4.3. So, $\nabla_{W_1} L = \nabla_{\vec{h}_1} L \vec{x}^T$

Convolutional Neural Networks

Example 4-3

$X \underset{s=1}{\overleftarrow{*}} K = H$ represents that the convolution between $X \in \mathbb{R}^{3\times 3}$ and the kernel $K \in \mathbb{R}^{2\times 2}$ with stride 1 is $H \in \mathbb{R}^{2\times 2}$.

$$\begin{bmatrix} X_{11} & X_{12} & X_{13} \\ X_{21} & X_{22} & X_{23} \\ X_{31} & X_{32} & X_{33} \end{bmatrix} \underset{s=1}{\overleftarrow{*}} \begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{bmatrix} = \begin{bmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{bmatrix}$$

where

$$\begin{aligned} H_{11} &= X_{11}K_{11} + X_{12}K_{12} + X_{21}K_{21} + X_{22}K_{22} \\ H_{12} &= X_{12}K_{11} + X_{13}K_{12} + X_{22}K_{21} + X_{23}K_{22} \\ H_{21} &= X_{21}K_{11} + X_{22}K_{12} + X_{31}K_{21} + X_{32}K_{22} \\ H_{22} &= X_{22}K_{11} + X_{23}K_{12} + X_{32}K_{21} + X_{33}K_{22} \end{aligned}$$

Convolutional Neural Networks

Example 4-3

In the convolution $X \underset{s=1}{\overset{\leftarrow}{*}} K = H$, assume $L(H) \in \mathbb{R}$ and $\nabla_H L$ are given, derive $\nabla_K L$ and $\nabla_X L$ in their closed matrix forms.

Solution 4-3-1:

1. Write $X \underset{s=1}{\overset{\leftarrow}{*}} K = H$ to a matrix multiplication $\hat{K}\vec{x} = \vec{h}$:

$$\begin{bmatrix} K_{11} & K_{12} & 0 & K_{21} & K_{22} & 0 & 0 & 0 & 0 \\ 0 & K_{11} & K_{12} & 0 & K_{21} & K_{22} & 0 & 0 & 0 \\ 0 & 0 & 0 & K_{11} & K_{12} & 0 & K_{21} & K_{22} & 0 \\ 0 & 0 & 0 & 0 & K_{11} & K_{12} & 0 & K_{21} & K_{22} \end{bmatrix} \begin{bmatrix} X_{11} \\ X_{12} \\ X_{13} \\ X_{21} \\ X_{22} \\ X_{23} \\ X_{31} \\ X_{32} \\ X_{33} \end{bmatrix} = \begin{bmatrix} H_{11} \\ H_{12} \\ H_{13} \\ H_{21} \end{bmatrix}$$

Convolutional Neural Networks

Solution 4-3-2:

1. Write $X \underset{s=1}{\overleftarrow{*}} K = H$ as a matrix multiplication $\hat{K}\vec{x} = \vec{h}$.
2. $\nabla_H L \iff \nabla_{\vec{h}} L$.
3. $\partial L = \mathrm{Tr}(\nabla_{\vec{h}} L^T \, \partial\vec{h})$.
   3.1. $\partial\vec{h} = \partial(\hat{K}\vec{x}) = (\partial\hat{K})\vec{x} + \hat{K}(\partial\vec{x})$
   3.2. So

$$\begin{aligned}\partial L &= \mathrm{Tr}(\nabla_{\vec{h}} L^T (\partial\hat{K})\vec{x}) + \mathrm{Tr}(\nabla_{\vec{h}} L^T \hat{K} \, \partial\vec{x}) \\ &= \mathrm{Tr}((\nabla_{\vec{h}} L \vec{x}^T)^T \, \partial\hat{K}) + \mathrm{Tr}((\hat{K}^T \nabla_{\vec{h}} L)^T \, \partial\vec{x})\end{aligned}$$

   3.3. $\nabla_{\hat{K}} L = \nabla_{\vec{h}} L \vec{x}^T$ and $\nabla_{\vec{x}} L = \hat{K}^T \nabla_{\vec{h}} L$

Future Work

1. In CNN, derive new differential laws using ∂K and ∂X to represent $\partial(X \overset{\leftarrow}{\underset{s=1}{*}} K)$ and Trace properties converting $\mathrm{Tr}(H^T(X \overset{\leftarrow}{\underset{s=1}{*}} K))$ to $\mathrm{Tr}(F(H,X,K)^T X)$ and $\mathrm{Tr}(G(H,X,K)^T K)$.
2. Derive the backpropagation process for other typical deep learning models: Generative Adversarial Networks(GANs)and Transformers.
3. Use the closed-matrix form backpropagation to analyze Batch Normalization and Residual Connection.