



✦ Get unlimited access to all of Medium. [Become a member](#)



Finding the Gradient of a Vector Function

Part 3 of Step by Step: The Math Behind Neural Networks



Chi-Feng Wang · [Follow](#)

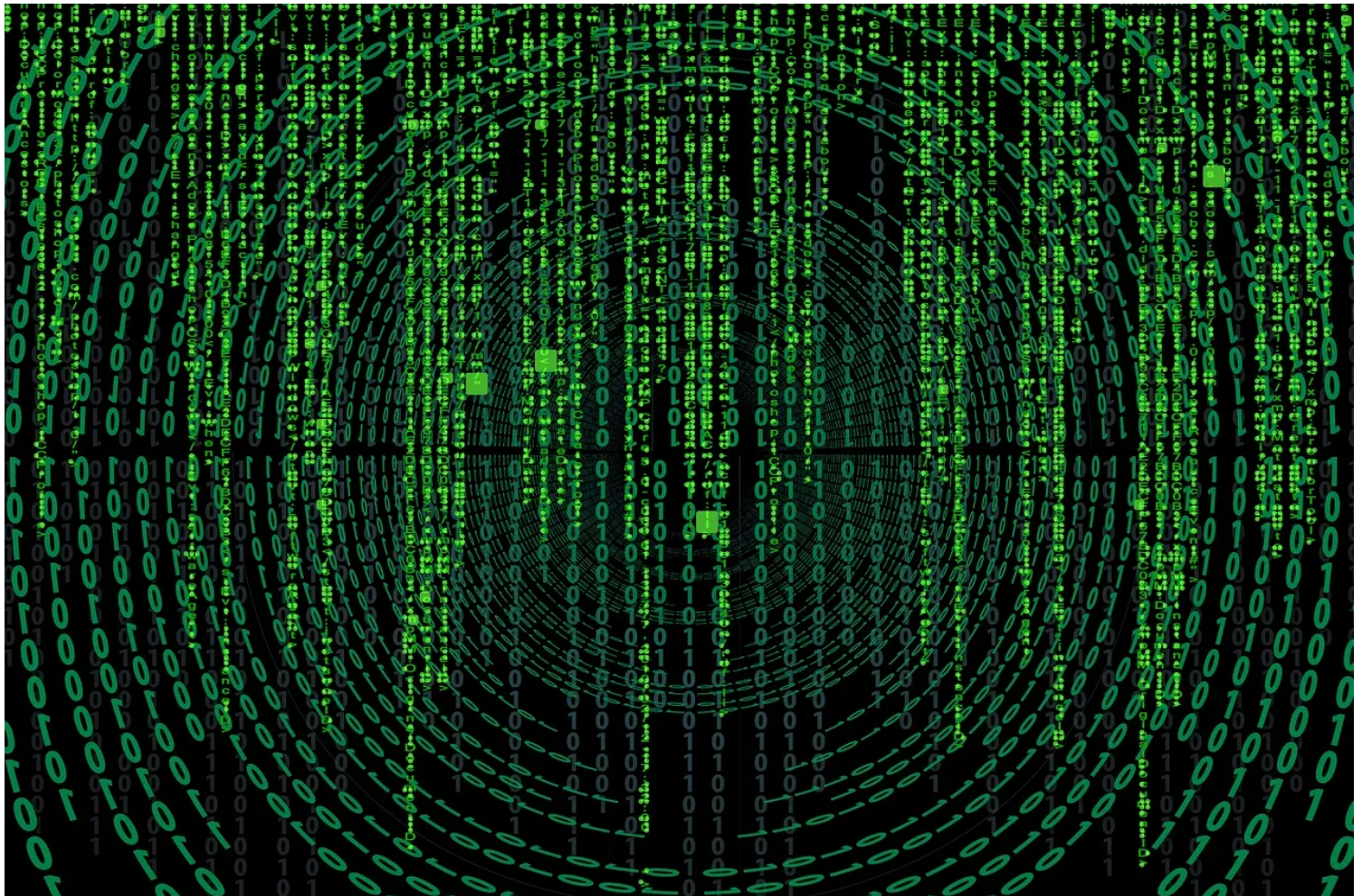
Published in Towards Data Science · 9 min read · Oct 20, 2018



465



1



In Part 1, we have been given a problem: to calculate the gradient of this loss function:

$$C(\mathbf{y}, \mathbf{w}, \mathbf{X}, b) = \frac{1}{N} \sum_{i=1}^N (y_i - \max(0, \mathbf{w} \cdot \mathbf{X}_i + b))^2$$

Image 1: Loss function

To find the gradient, we have to find the derivative the function. In Part 2, we learned to how calculate the partial derivative of function with respect to each variable. However, most of the variables in this loss function are vectors. Being able to find the partial derivative of vector variables is especially important as neural network deals with large quantities of data. Vector and matrix operations are a simple way to represent the operations with so much data. How, exactly, can you find the gradient of a vector function?

Gradient of a Scalar Function

Say that we have a function, $f(x,y) = 3x^2y$. Our partial derivatives are:

$$\frac{\partial f(x, y)}{\partial x} = \frac{\partial}{\partial x} 3x^2 y = 6yx$$

$$\frac{\partial f(x, y)}{\partial y} = \frac{\partial}{\partial y} 3x^2 y = 3x^2$$

Image 2: Partial derivatives

If we organize these partials into a horizontal vector, we get the **gradient** of $f(x,y)$, or $\nabla f(x,y)$:

Image 3: Gradient of $f(x,y)$

$6yx$ is the change in $f(x,y)$ with respect to a change in x , while $3x^2$ is the change in $f(x,y)$ with respect to a change in y .

What happens when we have two functions? Let's add another function, $g(x,y) = 2x+y^8$. The partial derivatives are:

Image 4: Partial derivatives of $g(x,y)$

So the gradient of $g(x,y)$ is:

Image 5: Gradient of $g(x,y)$

Representing Functions

When we have a multiple functions with multiple parameters, it's often useful to represent them in a simpler way. We can combine multiple parameters of functions into a single vector argument, \mathbf{x} , that looks as follows:

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

Image 6: Vector \mathbf{x}

Therefore, $f(x,y,z)$ will become $f(x_1,x_2,x_3)$ which becomes $f(\mathbf{x})$.

We can also combine multiple functions into a vector, like so:

Image 7: Vector \mathbf{y}

Now, $\mathbf{y}=\mathbf{f}(\mathbf{x})$ where $\mathbf{f}(\mathbf{x})$ is a vector of $[f_1(\mathbf{x}), f_2(\mathbf{x}), f_3(\mathbf{x})...f_n(\mathbf{x})]$

For our previous example with two functions, $f(x,y) \Rightarrow f(\mathbf{x})$ and $g(x,y) \Rightarrow g(\mathbf{x})$. Here, vector $\mathbf{x} = [x_1, x_2]$, where $x_1=x$, and $x_2=y$. To simplify it even further, we can combine our functions: $[f(\mathbf{x}),g(\mathbf{x})] = [f_1(\mathbf{x}), f_2(\mathbf{x})] = \mathbf{f}(\mathbf{x}) = \mathbf{y}$.

Image 8: Equations within vector function y

Often, the number of functions and the number of variables will be the same, so that a solution exists for each variable.

Gradient of a Vector Function

Now that we have two functions, how can we find the gradient of both functions? If we organize both of their gradients into a single matrix, we move from vector calculus into matrix calculus. This matrix, and organization of the gradients of multiple functions with multiple variables, is known as the **Jacobian matrix**.

$$J = \begin{bmatrix} \nabla f(x, y) \\ \nabla g(x, y) \end{bmatrix} = \begin{bmatrix} \frac{\partial f(x, y)}{\partial x} & \frac{\partial f(x, y)}{\partial y} \\ \frac{\partial g(x, y)}{\partial x} & \frac{\partial g(x, y)}{\partial y} \end{bmatrix} = \begin{bmatrix} 6yx & 3x^2 \\ 2 & 8y^7 \end{bmatrix}$$

Image 9: The Jacobian

There are multiple ways of representing the Jacobian. This layout, where we stack the gradients vertically, is known as the **numerator layout**, but other papers will use the **denominator layout**, which simply flips it diagonally:

Image 10: Denominator layout of the Jacobian

Gradient of the Identity Function

Let's take the identity function, $\mathbf{y} = \mathbf{f}(\mathbf{x}) = \mathbf{x}$, where $f_i(\mathbf{x}) = x_i$, and find its

gradient:

$$\begin{array}{ccccc} y_1 & = & f_1(\mathbf{x}) & = & x_1 \\ y_2 & = & f_2(\mathbf{x}) & = & x_2 \\ & & \vdots & & \\ y_n & = & f_n(\mathbf{x}) & = & x_n \end{array}$$

Image 11: Identity function

Just as we created our previous Jacobian, we can find the gradients of each scalar function and stack them up vertically to create the Jacobian of the identity function:

Image 12: Jacobian of the identity function

Since it's an identity function, $f_1(\mathbf{x}) = x_1$, $f_2(\mathbf{x}) = x_2$, and so on. Therefore,

Image 13: Jacobian of the identity function

The partial derivative of a function with respect to a variable that's not in the function is zero. For example, the partial derivative of $2x^2$ with respect to y is 0. In other words,

Image 14: The partial derivative of a function with respect to a variable that's not in the function is zero

Therefore, everything not on the diagonal of the Jacobian becomes zero. Meanwhile, the partial derivative of any variable with respect to itself is 1. For example, the partial derivative of x with respect to x is 1. Therefore, the Jacobian becomes:

Image 15: Jacobian of the identity function

Gradient of Element-Wise Vector Function Combinations

Element-wise binary operators are operations (such as addition $w+x$ or $w>x$ which returns a vector of ones and zeros) that applies an operator consecutively, from the first item of both vectors to get the first item of output, then the second item of both vectors to get the second item of output...and so forth.

This paper represents element-wise binary operations with this notation:

$$\mathbf{y} = \mathbf{f}(\mathbf{w}) \bigcirc \mathbf{g}(\mathbf{x})$$

Image 16: Element-wise binary operation with $f(x)$ and $g(x)$

Here, the \bigcirc means any element-wise operator (such as $+$), and not a composition of functions.

So how do you find the gradient of an element-wise operation of two vectors?

Since we have two sets of functions, we need two Jacobians, one representing the gradient with respect to \mathbf{x} and one with respect to \mathbf{w} :

$$J_{\mathbf{w}} = \frac{\partial \mathbf{y}}{\partial \mathbf{w}} = \begin{bmatrix} \frac{\partial}{\partial w_1}(f_1(\mathbf{w}) \circ g_1(\mathbf{x})) & \frac{\partial}{\partial w_2}(f_1(\mathbf{w}) \circ g_1(\mathbf{x})) & \dots & \frac{\partial}{\partial w_n}(f_1(\mathbf{w}) \circ g_1(\mathbf{x})) \\ \frac{\partial}{\partial w_1}(f_2(\mathbf{w}) \circ g_2(\mathbf{x})) & \frac{\partial}{\partial w_2}(f_2(\mathbf{w}) \circ g_2(\mathbf{x})) & \dots & \frac{\partial}{\partial w_n}(f_2(\mathbf{w}) \circ g_2(\mathbf{x})) \\ \dots & \dots & \dots & \dots \\ \frac{\partial}{\partial w_1}(f_n(\mathbf{w}) \circ g_n(\mathbf{x})) & \frac{\partial}{\partial w_2}(f_n(\mathbf{w}) \circ g_n(\mathbf{x})) & \dots & \frac{\partial}{\partial w_n}(f_n(\mathbf{w}) \circ g_n(\mathbf{x})) \end{bmatrix}$$

$$J_{\mathbf{x}} = \frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial}{\partial x_1}(f_1(\mathbf{w}) \circ g_1(\mathbf{x})) & \frac{\partial}{\partial x_2}(f_1(\mathbf{w}) \circ g_1(\mathbf{x})) & \dots & \frac{\partial}{\partial x_n}(f_1(\mathbf{w}) \circ g_1(\mathbf{x})) \\ \frac{\partial}{\partial x_1}(f_2(\mathbf{w}) \circ g_2(\mathbf{x})) & \frac{\partial}{\partial x_2}(f_2(\mathbf{w}) \circ g_2(\mathbf{x})) & \dots & \frac{\partial}{\partial x_n}(f_2(\mathbf{w}) \circ g_2(\mathbf{x})) \\ \dots & \dots & \dots & \dots \\ \frac{\partial}{\partial x_1}(f_n(\mathbf{w}) \circ g_n(\mathbf{x})) & \frac{\partial}{\partial x_2}(f_n(\mathbf{w}) \circ g_n(\mathbf{x})) & \dots & \frac{\partial}{\partial x_n}(f_n(\mathbf{w}) \circ g_n(\mathbf{x})) \end{bmatrix}$$

Image 17: Jacobian with respect to \mathbf{w} and \mathbf{x}

Most arithmetic operations we'll need are simple ones, so $f(w)$ is often simply the vector w . In other words, $f_i(w_i) = w_i$. For example, the operation $w+x$ fits this category as it can be represented as $f(w)+g(x)$ where $f_i(w_i) + g_i(x_i) = w_i + x_i$.

Under this condition, every element in the two Jacobians simplifies to:

$$J_w : \frac{\partial}{\partial w_j} (f_i(w_i) \circ g_i(x_i)) = \frac{\partial}{\partial w_j} (w_i \circ x_i)$$

$$J_x : \frac{\partial}{\partial x_j} (f_i(w_i) \circ g_i(x_i)) = \frac{\partial}{\partial x_j} (w_i \circ x_i)$$

Image 18: Elements in the Jacobian

On the diagonal, $i=j$, so a value exists for the partial derivative. Off the diagonal, however, $i \neq j$, so the partial derivatives become zero:

Image 19: Diagonal jacobian

We can represent this more succinctly as:

$$\frac{\partial \mathbf{y}}{\partial \mathbf{w}} = \text{diag} \left(\frac{\partial}{\partial w_1} (f_1(w_1) \odot g_1(x_1)), \frac{\partial}{\partial w_2} (f_2(w_2) \odot g_2(x_2)), \dots, \frac{\partial}{\partial w_n} (f_n(w_n) \odot g_n(x_n)) \right)$$

$$\frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \text{diag} \left(\frac{\partial}{\partial x_1} (f_1(w_1) \odot g_1(x_1)), \frac{\partial}{\partial x_2} (f_2(w_2) \odot g_2(x_2)), \dots, \frac{\partial}{\partial x_n} (f_n(w_n) \odot g_n(x_n)) \right)$$

Image 20: The Jacobian with respect to \mathbf{w} and \mathbf{x}

Let's try to find the gradient of the function $\mathbf{w} + \mathbf{x}$. We know that everything off the diagonal is 0. The values of the partials on the diagonal with respect to \mathbf{w} and \mathbf{x} are:

$$\frac{\partial}{\partial w_i} (f_i(w_i) + g_i(x_i)) = \frac{\partial}{\partial w_i} (w_i + x_i) = 1 + 0 = 1$$

$$\frac{\partial}{\partial x_i} (f_i(w_i) + g_i(x_i)) = \frac{\partial}{\partial x_i} (w_i + x_i) = 0 + 1 = 1$$

Image 21: Partial derivatives with respect to \mathbf{w} and \mathbf{x}

So both Jacobians have a diagonal of 1. This looks familiar...it's the identity matrix!

Let's try it with multiplication: $\mathbf{w} * \mathbf{x}$. The values of the partials on the diagonal with respect to \mathbf{w} and \mathbf{x} are:

Image 22: Partial derivatives with respect to \mathbf{w} and \mathbf{x}

Therefore, the gradient with respect to \mathbf{w} of $\mathbf{w}^*\mathbf{x}$ is $\text{diag}(\mathbf{x})$, while the gradient with respect to \mathbf{x} of $\mathbf{w}^*\mathbf{x}$ is $\text{diag}(\mathbf{w})$.

Applying the same steps for subtraction and division, we can sum it all up:

Gradient of Vector Sums

One of the most common operations in deep learning is the summation operation. How can we find the gradient of the function $y = \text{sum}(\mathbf{x})$?

$y = \text{sum}(\mathbf{x})$ can also be represented as:

$$y = \sum_i x_i$$

Image 24: $y = \text{sum}(\mathbf{x})$

Therefore, the gradient can be represented as:

Image 25: Gradient of $y = \text{sum}(\mathbf{x})$

And since the partial derivative of a function with respect to a variable that's not in the function is zero, it can be further simplified as:

Image 26: Gradient of $y = \text{sum}(\mathbf{x})$

Note that the result is a horizontal vector.

What about the gradient of $y = \text{sum}(\mathbf{x}z)$? The only difference is that we multiply every partial with a constant, z :

Image 27: Gradient of $y = \text{sum}(\mathbf{x}z)$ with respect to \mathbf{x}

While that is the derivative with respect to \mathbf{x} , the derivative with respect to the scalar z is simply a number:

Image 28: Gradient of $y = \text{sum}(\mathbf{x}z)$ with respect to z

Gradient of Chain Rule Vector Function Combinations

In [Part 2](#), we learned about the multivariable chain rules. However, that only works for scalars. Let's see how we can integrate that into vector calculations!

Let us take a vector function, $y = f(x)$, and find its gradient. Let us define the function as:

$$\begin{bmatrix} y_1(x) \\ y_2(x) \end{bmatrix} = \begin{bmatrix} f_1(x) \\ f_2(x) \end{bmatrix} = \begin{bmatrix} \ln(x^2) \\ \sin(3x) \end{bmatrix}$$

Image 29: $y = f(x)$

Both $f_1(x)$ and $f_2(x)$ are composite functions. Let us introduce intermediate variables for $f_1(x)$ and $f_2(x)$, and rewrite our function:

Image 30: $y = f(g(x))$

Now, we can use our multivariable chain rule to compute the derivative

of vector \mathbf{y} . Simply compute the derivative of $f_1(x)$ and $f_2(x)$, and place them one above another:

Image 31: Gradient of $\mathbf{y} = \mathbf{f}(\mathbf{g}(x))$

Voila! We have our gradient. However, we've come to our solution with scalar rules, merely grouping the numbers together into a vector. Is there a way to represent the multivariable chain rule for vectors?

Right now, our gradient is computed with:

Image 32: Gradient of $\mathbf{y} = \mathbf{f}(\mathbf{g}(x))$

Notice that the first term of the gradients of both $f_1(x)$ and $f_2(x)$ include the partial of g_1 over x , and the second term of the gradients of both $f_1(x)$ and $f_2(x)$ include the partial of g_2 over x . This is just like matrix multiplication! We can therefore represent it as:

Image 33: Vector representation of the gradient of $\mathbf{y} = \mathbf{f}(\mathbf{g}(x))$

Let's test our this new representation of the vector chain rule:

Image 34: Vector chain rule

We get the same answer as the scalar approach! If instead of a single parameter x we have a vector parameter \mathbf{x} , we just have to alter our rule a little to get the complete vector chain rule:

Image 35: Vector chain rule

In other words:

Image 36: Vector chain rule

In our example above, f is purely a function of g ; that is, f_i is a function of g_i but not g_j (each function f matches with exactly 1 function g). In this case, everything off the diagonal becomes zero, and:

Image 37: Special case of vector chain rule

Now we have all the pieces we find the gradient of the neural network we started our series with:

$$C(\mathbf{y}, \mathbf{w}, \mathbf{X}, b) = \frac{1}{N} \sum_{i=1}^N (y_i - \max(0, \mathbf{w} \cdot \mathbf{X}_i + b))^2$$

Image 38: Cost function

Check out [Part 4](#) to find out how to compute its derivative!

If you haven't already, read Parts 1 and 2:

- [Part 1: Introduction](#)
- [Part 2: Partial Derivatives](#)

Read [Part 4](#) for the grand finale!

Download the original paper [here](#).

If you like this article, don't forget to leave some claps! Do leave a comment below if you have any questions or suggestions :)

Artificial Intelligence

Matrix

Calculus

Gradient

Neural Networks



Written by Chi-Feng Wang

1.5K Followers · Writer for Towards Data Science

Student at UC Berkeley; Machine Learning Enthusiast

Follow



More from Chi-Feng Wang and Towards Data Science

 Chi-Feng Wang in Towards Data Science

The Vanishing Gradient Problem

The Problem, Its Causes, Its Significance, and Its Solutions


3 min read · Jan 8, 2019



2.7K

 10



 Jacob Marks, Ph.D. in Towards Data Science

How I Turned My Company's Docs into a Searchable Database with...

And how you can do the same with your docs

15 min read · Apr 25



3.1K

 40



 Leonie Monigatti in Towards Data Science

Getting Started with LangChain: A Beginner's Guide to Building...

A LangChain tutorial to build anything with large language models in Python

★ · 12 min read · Apr 25



2.2K

 18



 Chi-Feng Wang in Towards Data Science

A Basic Introduction to Separable Convolutions

Explaining spatial separable convolutions, depthwise separable convolutions, and th...

8 min read · Aug 14, 2018



7.1K


 44



See all from Chi-Feng Wang

See all from Towards Data Science

Recommended from Medium

 Alexander Nguyen in Level Up Coding

Why I Keep Failing Candidates During Google Interviews...

They don't meet the bar.

★ · 4 min read · Apr 13




4.2K



127



 Peter Kar... in Artificial Intelligence in Plain Eng...

Linear Regression in depth

The directive equation of a straight line, simple linear regression, math, cost...

★ · 6 min read · Jan 27



111



2




Lists

What is ChatGPT?

9 stories · 64 saves

Staff Picks

330 stories · 83 saves

 The PyCoach in Artificial Corner

You're Using ChatGPT Wrong! Here's How to Be Ahead of 99% ...

Master ChatGPT by learning prompt engineering.

★ · 7 min read · Mar 18




21K



364



 Peter Kar... in Artificial Intelligence in Plain Eng...

L1 (Lasso) and L2 (Ridge) regularizations in logistic...

Logistic regression , Lasso and Ridge regularizations, derivations, math

★ · 6 min read · Feb 3



38



1



 Alvin T. in Japonica Publication

Why Translating Emails from English to Japanese Using Only...

Japanese business language proves to be a barrier to AI translation—here's what to d...

★ · 7 min read · Apr 18



681



6



 Luís Roque in Towards Data Science

Maximum Likelihood Estimation from scratch in TensorFlow...

Probabilistic deep learning

★ · 9 min read · Nov 30, 2022



116



1



[See more recommendations](#)