Neural Networks

Kieran Harvie

Copyright ©August 3, 2023. All Rights Reserved.

This document is still under construction.
Changes in content, structure, and readability are expected.

Contents

Introduction

Automatic Differentiation

Automatic Differentiation is a collection of techniques for algorithmically calculating the partial derivative of a function in some general way. The core observation is that if the functions can be expressed in terms of elementary functions with known partial derivatives we should be able to use the chain rule to calculate the partial derivative of the original function. General

2.1 A Tale of Two Derivative

Partial and total. Chain rule. ∇ operator maybe an appendix for vectors?

$$\frac{d\,f}{d\,t} = \sum_i \frac{\partial f}{\partial x_i}\frac{d\,x_i}{d\,t} = \nabla f \cdot \left(\frac{d\,x_0}{d\,t}, \frac{d\,x_1}{d\,t}, \frac{d\,x_2}{d\,t}, \dots\right)$$

2.2 Forward Accumulation

2.2.1 Dual Numbers

Like how we get Complex numbers from Real numbers by adjoining a new number i such that $i^2 = -1$ we get the Dual numbers by adjoining a new number ϵ, called the infinitesimal, such that $\epsilon^2 = 0$.

Some properties aren't that special:

$$(x + x'\epsilon) + (y + y'\epsilon) = (x + y) + (x' + y')\epsilon$$

But consider what happens for multiplication and division:

$$\begin{aligned}(x+x'\epsilon)(y+y'\epsilon) =& xy + (x'y + xy')\epsilon + x'y'\epsilon^2 \\ =& xy + (x'y + xy')\epsilon \\ \frac{x+x'\epsilon}{y+y'\epsilon} =& \frac{(x+x'\epsilon)(y-y'\epsilon)}{(y+y'\epsilon)(y-y'\epsilon)} \\ =& \frac{xy + (x'y - xy')\epsilon - x'y'\epsilon^2}{y^2 - y'^2\epsilon^2} \\ =& \frac{x}{y} + \frac{x'y - xy'}{y^2}\epsilon\end{aligned}$$

Define: $f\left(\langle u, u'\rangle, \langle v, v'\rangle\right) = \langle f(u,v), \nabla f(u,v) \cdot (u', v')\rangle$
Because:

$$\begin{aligned}f\left(\left\langle u, \frac{du}{dt}\right\rangle, \left\langle v, \frac{dv}{dt}\right\rangle\right) &= \left\langle f(u,v), \nabla f(u,v) \cdot \left(\frac{du}{dt}, \frac{dv}{dt}\right)\right\rangle \\ &= \left\langle f(u,v), \frac{d}{d\,t} f(u,v)\right\rangle\end{aligned}$$

(Because of the similar form this argument also works with partials).

Historical Note

As observed with the multiplication and divisions examples, representing infinitesimals with $\epsilon^2 = 0$ means discarding any powers of ϵ larger than 1.

This actually mirrors how calculus was developed. In 1710 Leibniz codified the "Transcendental Law of Homogeneity" which states that when equating sums involving to only include the lowest order infinitesimal terms.

For example, if we have infinitesimals du and dv the Transcendental Law of Homogeneity would mean that:

$$dv^2 + dudv + 2dv = 2dv$$

This can be used to calculate derivative of a function by subtracting the version with with finite variables from infinitesimals ones:

$$(v + dv)(u + du) - uv = vdu + udv + dudv = vdu + udv$$

Which clearly mimics the multiplication example.

A historical note on a historical note, this specific application of the rule is similar to "Adequality" which can be traced back to Pierre de Fermat in a 1636 treatise. Here we find the extrema of a function f at a value x "adequating" $f(x)$ to $f(x+e)$, then dividing by e and discarding any remaining terms involving e.

For a worked example consider $f(x) = x^2 + x + 1$ and represent "adequality" with $\sim$:

$$\begin{aligned} f(x) \sim & f(x+e) \\ x^2 + x + 1 \sim & (x+e)^2 + (x+e)1 \\ \sim & x^2 + 2xe + e^2 + x + e + 1 \\ 0 \sim & 2xe + e^2 + e \quad \text{(Cancellation)} \\ 0 \sim & 2x + e + 1 \quad \text{(Division by } e\text{)} \\ 0 \sim & 2x + 1 \quad \text{(Discarding } e\text{)} \end{aligned}$$

Observe the similarity with the previous arguments and boils down to $\frac{df}{dx} = 0$.

The purpose of this historical tangent is to establish manipulating infinitesimals like this is not some fringe idea related solely to this application.

2.2.2 The Algorithm

Forward Accumulation is the most direct form of automatic differentiation.

Consider the following relation:

$$f(\langle x, 1\rangle, \langle y, 0\rangle) = \left\langle f(x,y), \frac{\partial}{\partial x} f(x,y) \right\rangle$$

Since dual numbers are easy for computers to represent and manipulate we can calculate $\frac{\partial f}{\partial x}$ by substituting in $\langle x, 1\rangle$ and $\langle y, 0\rangle$ and reading off the final infinitesimal.

Consider $f(x,y) = \cos(xy) + y$, cos is an elementary function for which we can easily verify:

$$\cos(\langle x, x'\rangle) = \langle \cos(x), -x' \sin(x)\rangle$$

Hence we can calculate the $\frac{\partial f}{\partial x} = 0$ at $x = 0$ and $y = 1$ as:

$$\begin{aligned} f(\langle 0,1\rangle, \langle 1,0\rangle) &= \cos(\langle 0,1\rangle \times \langle 1,0\rangle) + \langle 1,0\rangle \\ &= \cos(\langle 0,1\rangle) + \langle 1,0\rangle \\ &= \langle 1,0\rangle + \langle 1,0\rangle \\ &= \langle 2,0\rangle \end{aligned}$$

Like wise for $\frac{\partial f}{\partial y} = 1$:

$$\begin{aligned} f(\langle 0,0\rangle, \langle 1,1\rangle) &= \cos(\langle 0,0\rangle \times \langle 1,1\rangle) + \langle 1,1\rangle \\ &= \cos(\langle 0,0\rangle) + \langle 1,1\rangle \\ &= \langle 1,0\rangle + \langle 1,1\rangle \\ &= \langle 2,1\rangle \end{aligned}$$

This method is call *forward* accumulation because we exclusively move from inputs, x and y, to intermediate results, $\cos(x, y)$, to outputs, $f(x, y)$. This has its advantages of letting us reuse how computers call functions to store intermediate results and avoid doing any overhead work. But its main flaw is only calculating one partial at a time.

We can can calculate multiple partials at time with the next method.

2.3 Reverse Accumulation

In reverse accumulation we do a forward pass to calculate and store $f(x, y)$, and intermediates, then do a second pass in the reverse direction to calculate the partials by fixing the final infinitesimal as 1.

To see the intuition as to why this would work consider:

$$\left\langle a(x,y), \frac{\partial f}{\partial a} \right\rangle = a\left(\left\langle x, \frac{\partial f}{\partial x} \right\rangle, \left\langle y, \frac{\partial f}{\partial y} \right\rangle\right)$$

Although this method requires multiple passes and the overhead of storing intermediate results it calculated all partials at the same time. Adjoint:

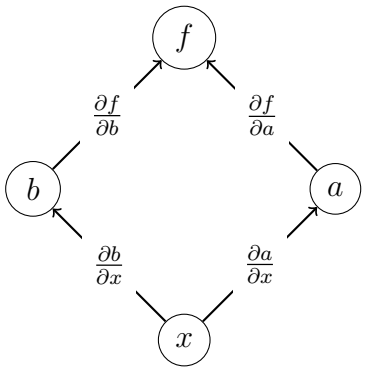
$$\bar{x} = \frac{\partial f}{\partial x}$$

Diamond with a forward value calculation and a reverse adjoint accumulation.

2.3.1 Algorithm Trace

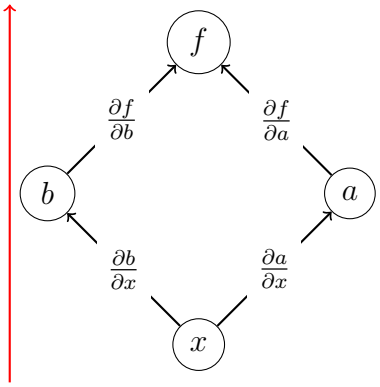
One visual trace of the is worth a thousand lines of analysis. Let $f(x, y) = \frac{1}{2}x + y$, $a(x) = x^2$, $b(x) = x^3$, and $x = 0.5$, we wish to preform reverse accumulation for $f(a(x), b(x))$.

We will represent the state of the algorithm as a weighed directed graph, the nodes are the inputs/outputs/intermediates and an accumulator, the edge direction is downstream (from input to output), the edge weights are the partial derivatives of the nodes.



Node:	x	a	b	f
Value: Accumulator:				
Edge:	$\frac{\partial a}{\partial x}$	$\frac{\partial b}{\partial x}$	$\frac{\partial f}{\partial a}$	$\frac{\partial f}{\partial b}$
Value:				

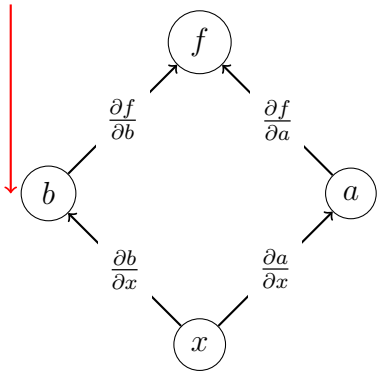
The first pass is forward and we calculate the values of the nodes and edges. This is straight forward, for example, since we have $x = 0.5$ and $a(x) = x^2$ the desired values are $a = 0.25$ and $\frac{\partial a}{\partial x} = 1.0$



Node:	x	a	b	f
Value: Accumulator:	0.5	0.25	0.125	0.25
Edge:	$\frac{\partial a}{\partial x}$	$\frac{\partial b}{\partial x}$	$\frac{\partial f}{\partial a}$	$\frac{\partial f}{\partial b}$
Value:	1.0	0.75	0.5	1.0

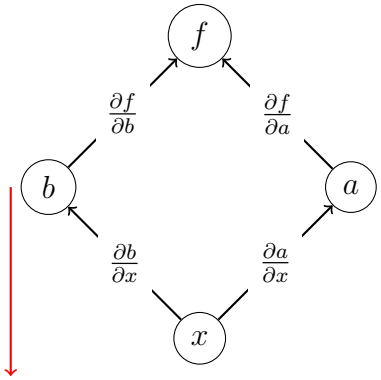
The second pass is backwards where we set the accumulator of the output f to 1 and then recursively add the product of downstream element accumulators by edge weight to their upstream elements accumulators.

We will break this into two parts, first part are the upstreams of the f node. The a accumulator is 0.5×1.0 and b's is 1.0×1.0:



Node:	x	a	b	f
Value:	0.5	0.25	0.125	0.25
Accumulator:		0.5	1.0	1.0
Edge:	$\frac{\partial a}{\partial x}$	$\frac{\partial b}{\partial x}$	$\frac{\partial f}{\partial a}$	$\frac{\partial f}{\partial b}$
Value:	1.0	0.75	0.5	1.0

And then we add continue to add the product of accumulator by edge weight to upstream nodes. The only upstream node left is x and in hence set to $0.5 \times 1.0 + 1.0 \times 0.75$:



Node:	x	a	b	f
Value:	0.5	0.25	0.125	0.25
Accumulator:	1.25	0.5	1.0	1.0
Edge:	$\frac{\partial a}{\partial x}$	$\frac{\partial b}{\partial x}$	$\frac{\partial f}{\partial a}$	$\frac{\partial f}{\partial b}$
Value:	1.0	0.75	0.5	1.0

The algorithm has completed and in the accumulator of each node is the partial derivative of f by that node. For example $\frac{\partial f}{\partial x} = 1.25$, $\frac{\partial f}{\partial a} = 0.5$, $\frac{\partial f}{\partial b} = 1.0$, and $\frac{\partial f}{\partial f} = 1.0$.

2.3.2 Adjoint Formula

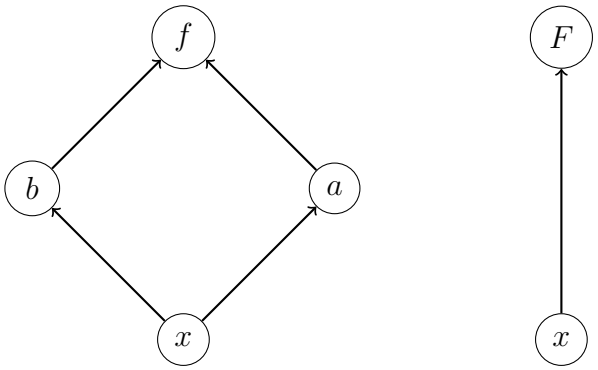
To see why the algorithm works we need a new math result. While it is very similar to the partial chain rule it is in the other direction.

If x is upstream of f and a_n are immediately downstream of x then we can write the partial $\frac{\partial f}{\partial x}$ as:

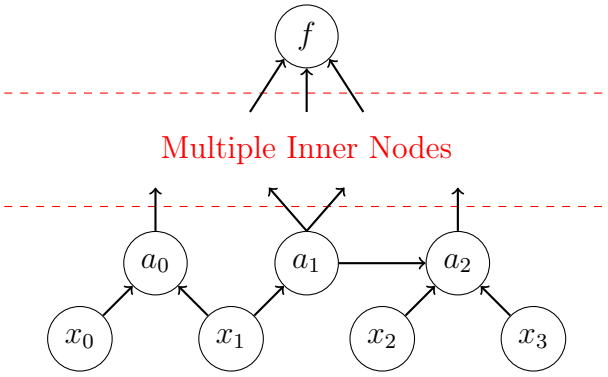
$$\frac{\partial f}{\partial x} = \sum_n \frac{\partial f}{\partial a_n} \frac{\partial a_n}{\partial x}$$

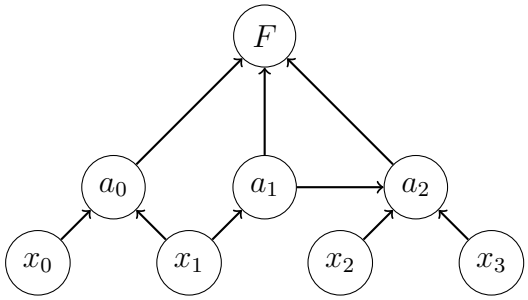
Informally this works because you can combine nodes together. In our previous example we can combine a, b, and f into a single F:

$$F(x) = f(a(x), b(x)) = \frac{1}{2}x^2 + x^3$$



To prove the adjoint relation we consider the more general case:





2.3.3 Evaluation Order

It relates to topologies

Fixed Order

Depth First

Layered

Optimization

3.1 Gradient Descent

3.2 Newton's Method

Like a dynamic step size since we have this info? Only when root finding. (Use duel numbers).

Using error l_2 all the functions are twice differentiable so you can do the Lagrange form of the remainder thing. Also only one root so how does that work with basins

$$x_{n+1} = x_n - \frac{F(x_n)}{|\nabla F(x_n)|^2} \nabla F(x_n)$$

Nodes

Transfer (SUM) Activator (Function (logistic)) Loss/Cost (Error)

4.1 Layered Structure

Back propagate but only keep one row in memory at a time.

Appendix

A.1 Logistic Function

A.2 Table of Elementary Functions of dual numbers