

# Fast.ai Lesson 8b

Joseph Catanzarite  
For TWiML study group  
13 July, 2019

# Overview

- Normalization & initialization
- Forward pass – compute predictions
- Backward pass – compute gradients
- Construct functions needed for backward pass
  - mse\_grad
  - relu\_grad
  - lin\_grad
  - forward\_and\_backward
- Refactoring, using classes

# Backward pass

- In general, **gradients** measure the sensitivity of a function to changes in its parameters
- The **backward pass** computes the gradients that measure the sensitivity of the **loss function** to the neural network parameters (**weights** and **biases**)
- Gradients tell you how the parameters should be **updated** to make the loss function smaller.
- During **training**, you iterate through **forward** and **backward** passes, each time adjusting the weights and biases a bit, in order to **minimize** the loss function.

course

[Go to part 1](#)[Lesson 8](#)[Lesson 9](#)[Lesson 10](#)[Lesson 11](#)[Lesson 12](#)[Lesson 13](#)[Lesson 14](#)

## Lesson 8 (2019) - Deep Learning from the Foundations

Watch later

Share

$$x \rightarrow \text{lin1} \rightarrow \text{relu} \rightarrow \text{lin2} \rightarrow \text{mse} \rightarrow \hat{y}$$

$$\hat{y} = \text{mse}(\text{lin2}(\text{relu}(\text{lin1}(x))), y)$$

$$y = f(u) \quad \frac{dy}{dx} = \frac{dy}{du} \cdot \frac{du}{dx}$$

$$u = f(x)$$

MORE VIDEOS

1:51:16 / 2:06:40

CC BY-NC-SA

## Lesson 8: Matrix multiplication; forward and backward passes

In this course, we will learn to implement many of the capabilities of Fastai and PyTorch that we could use to build our own deep learning libraries. Along the way, we will learn to implement research papers, which is an important skill to master when making state of the art models.

### Lesson resources

- [Course notebooks](#)
- [Slides](#)
- [Excel spreadsheets](#) (today's is called [broadcasting.xlsx](#)). There's also a [Google Sheet version](#) thanks to @Moody

### Lesson overview

In today's lesson we'll discuss the purpose of this course, which is, in some ways, the opposite of part 1. This time, we're not learning practical things that we will use right away, but foundations that we can build on. This is particularly important nowadays because this field is moving so fast. We'll also talk about why the last two lessons of this course are about Swift, not Python (Chris Lattner, the original creator of Swift, and now lead of Swift for TensorFlow, will be joining Jeremy to co-teach these lessons).

## Shai

## Brought to

(We teach in University of San Francisco's [MS in Data Science program](#) and have other nefarious projects underway. You might know Terence as the creator of the [ANTLR parser generator](#). For more material, see Jeremy's [fast.ai courses](#) and University of San Francisco's Data Institute [in-person version of the deep learning course](#).)

[Printable version](#) (This HTML was generated from markup using [bookish](#))

## Abstract

This paper is an attempt to explain all the matrix calculus you need in order to understand the training of deep neural networks. We assume no math knowledge beyond what you learned in calculus 1, and provide links to help you refresh the necessary math where needed. Note that you do **not** need to understand this material before you start learning to train and use deep learning in practice; rather, this material is for those who are already familiar with the basics of neural networks, and wish to deepen their understanding of the underlying math. Don't worry if you get stuck at some point along the way-- just go back and reread the

More videos

More videos: questions in the [Theory](#) category at [forums.fast.ai](#). Note: There is a [reference section](#) at the end of the paper summarizing all the key matrix calculus rules and terminology discussed here.

$x \rightarrow \ln 3 = \ln e \ln 3 = \ln 2 \rightarrow msc \rightarrow G$   
 $G = msc(\ln 2(\ln 2(\ln 2(x)))y)$   
 $y = f(x)$   
 $u = f(x)$



Figure 1 displays several mathematical structures and diagrams:

- Top Left:** A 10x10 grid of numbers 1 through 10, arranged in a repeating pattern.
- Top Middle:** A table with 4 rows and 4 columns of numbers.
- Top Right:** A table with 4 rows and 4 columns of numbers.
- Bottom Left:** A vertical sequence of numbers 1 through 10.
- Bottom Middle:** A diagram showing 10 nodes on the left connected to 4 nodes on the right by multiple colored lines.



**Why not Python?**

Python is nice... but:

- **Slow:** forces things into external C libraries - see lesson 8!
- **Conspicuous:** CIL forces more into external C libraries
- **Accidental:** forces more into CUDA, etc.

We'll wrap up our Python adventures with a deeper dive in to u-nets, plus some more vision applications

|                    |                   |                               |              |
|--------------------|-------------------|-------------------------------|--------------|
| Build your own DNN | Tensor2           | U-net Deep Dive               | Fast shuffle |
| Self attention     | Variable Sharding | Generational video models     | DeiTSL       |
|                    | CycleGAN          | Object detection (DeiT-u-net) |              |

▶ 🔊 1:48:48 / 2:06:40

CC



7

In this course, we will learn to implement many of the capabilities of Fastai and PyTorch that we could use to build our own deep learning libraries. Along the way, we will learn to implement research papers, which is an important skill to master when making state of the art models.

## Lesson resources

- Course notebooks
- Slides
- Excel spreadsheets (today's is called broadcasting.xlsx). There's also a Google Sheet version thanks to @Moody

## Lesson overview

In today's lesson we'll discuss the purpose of this course, which is, in some ways, the opposite of part 1. This time, we're not learning practical things that we will use right away, but foundations that we can build on. This is particularly important nowadays because this field is moving so fast. We'll also talk about why the last two lessons of this course are about Swift, not Python (Chris Lattner, the original creator of Swift, and now lead of Swift for TensorFlow, will be joining Jeremy to co-teach these lessons).

# Next up

- Lesson 9: build resnet