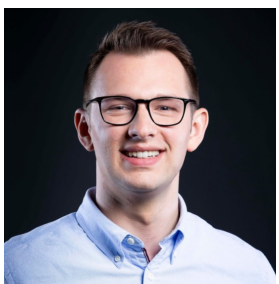
Introduction & Machine Learning Review

course overview, review of fully connected networks, backpropagation, optimization

CSC420
David Lindell
University of Toronto
cs.toronto.edu/~lindell/teaching/420
Slide credit: Babak Taati ←Ahmed Ashraf ←Sanja Fidler

TCIG
Toronto Computational Imaging Group

Who are we?



David Lindell

Javad Rajabi

Michael Neumayr

Andrew Guo

Who are you?

Outline

- motivation
- syllabus
- course overview
- fully-connected networks & neural fields
- backpropagation

Outline

- motivation
- syllabus
- course overview
- fully-connected networks & neural fields
- backpropagation

Hu et al. ‘25

Hu et al. ‘25

Segment Anything v2 (Meta)


Why is this funny?

Does GPT 5 “get it”?

Why is this funny?

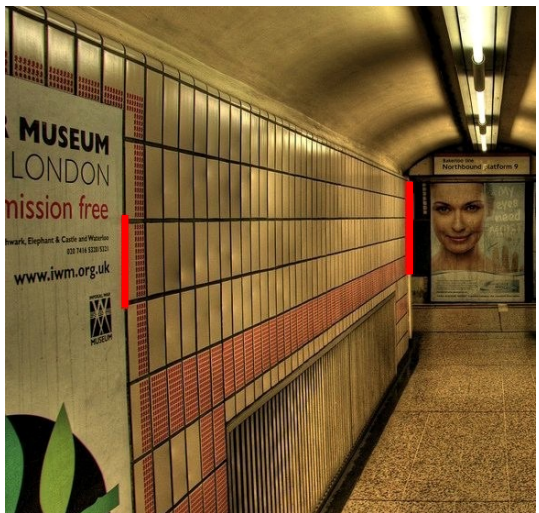
This image is funny because it captures a playful, unexpected moment:

- The man in the suit on the scale is presumably getting weighed in a serious or routine setting.
- Meanwhile, another man (smiling broadly) is playfully stepping on the scale from behind, adding extra weight without the person on the scale noticing immediately.
- The humor comes from the subversion of expectations—scales are typically used seriously, but here, the reading is being mischievously manipulated.
- The reactions of the surrounding people—smiling, laughing, or clearly in on the joke—add to the comedic tone.

It’s essentially a lighthearted prank in what would otherwise be a formal environment, which makes it even funnier.

How do we understand images?

Which red line is longer?



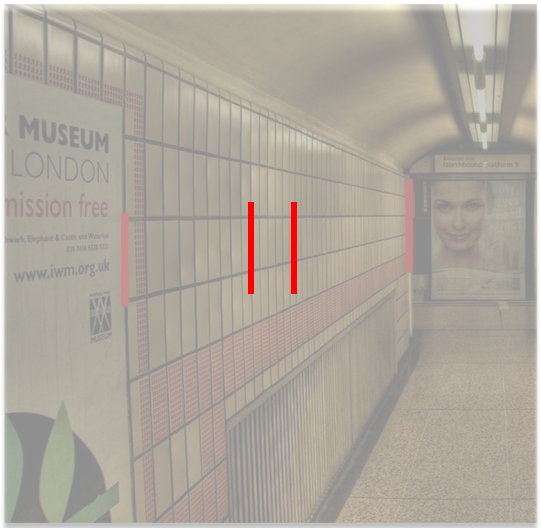
[Walt Anthony 2006]

[Slide: A. Torralba]

How do we understand images?

Which red line is longer?

They are the same…



[Walt Anthony 2006]

[Slide: A. Torralba]

How do we understand images?

Can you describe what this is?

[Torralba et al.]

How do we understand images?

Can you describe what this is?

[Torralba et al.]

Humans can tell a lot from a little information...
we have prior knowledge that can (usually) fill in the right information

How do we understand images?

Can you describe what this is?

[Torralba et al.]

How can we create computer vision models that have the same capabilities?

Outline

- motivation
- syllabus
- course overview
- fully-connected networks & neural fields
- backpropagation

Course Info

- Class time: Wednesdays 1-3pm (MP137) or 3-5pm (GB221)
- Tutorials: Fridays 1-2pm (GB221) or 3-4pm (MP137)
- Class Website: https://www.cs.toronto.edu/~lindell/teaching/420/
- Quercus: https://q.utoronto.ca/
- Course material (lecture notes, reading material, assignments, announcements, etc.) will be posted on Quercus
- Forum: Piazza (link on Quercus)
- Your grade will not depend on your participation on discussions. It's just a good way for asking questions, discussing with your instructor, TAs and your peers

Textbooks

- Computer Vision: Algorithms and Applications (Szeliski) — available online
- Foundations of Computer Vision (Torralba, Isola, Freeman) — available online
- 3D Computer Graphics (Gortler) — on Quercus
- Multiple View Geometry in Computer Vision (Hartley, Zisserman) — on Quercus

Links to other material (papers, code, etc.) will be posted on the class webpage

Course Prerequisites

- Machine learning (CSC311)
- Fundamentals of computer vision/image processing (CSC320)
- Data structures
- Linear Algebra
- Vector calculus

Knowing these is a plus:

- Python/PyTorch
- Jupyter Notebooks
- (Solving assignments sooner rather than later)

Grading

- Assignments (29%)
  - Assignment 1: 8%
  - Assignment 2: 7%
  - Assignment 3: 7%
  - Assignment 4: 7%

- Exams (70%)
  - Midterm Exam 30%
  - Final Exam: 40%

- Ethics Module: 1% (2 surveys, 0.5 each)

Assignments

- Download from Files section on Quercus, Submitted via MarkUs

Assignments

- Download from Files section on Quercus, Submitted via MarkUs

- Assignments: They will consist of problem sets and programming problems with the goal of deepening your understanding of the material covered in class.
  - problem sets will not be graded — your knowledge of these concepts will be tested on the midterm and final
  - programming problems will be graded

Assignments

- Download from Files section on Quercus, Submitted via MarkUs

- Assignments: They will consist of problem sets and programming problems with the goal of deepening your understanding of the material covered in class.
  - problem sets will not be graded — your knowledge of these concepts will be tested on the midterm and final
  - programming problems will be graded

- Assignment 1 is out now, Thurs Sep 25 at 12:59 pm (in the afternoon)

Assignments

Deadline

- The solutions to the assignments / project should be submitted by 12:59 pm on the date they are due.

Assignments

Deadline

- The solutions to the assignments / project should be submitted by 12:59 pm on the date they are due.

Lateness

- We provide one hour grace period after the deadline to account for any technical glitches
- If you require an extension fill out the special consideration request form (link on the course webpage). These requests will be automatically granted up to an extension of 7 days.
- Extensions beyond 7 days will not be granted!

Assignments

Collaboration policy

- ***Graded portions of assignments must be implemented individually***. However, you may discuss general strategies and discuss the concepts of the course with other students, TAs (e.g., on Piazza), or AI. You must not share code or derivations.

- **AI Assistants Policy:** You are welcome to use AI assistants in a responsible fashion.
  - You should not ask an AI assistant to complete your assignments for you; this is just as inappropriate as asking another student.
  - However, you may ask an AI assistant questions about lecture material, clarifications about the assignments, tips on how to approach a problem, etc.
  - **If you decide to use an AI assistant, you must include a sentence or two describing how you used it at the very top of your submitted assignment.**

Midterm

- A timed one-hour exam held during the tutorial hour.
- The exam will be a combination of multiple choice and short-answer/long-answer.
- **There are no make-ups for the midterm.** If you miss the midterm, we will add a 30% weight to your final exam (so that the final will be worth 70% of your total grade).

All info is on the course website!

Schedule and Syllabus

Week	Date	Description	Material	Readings	Event	Deadline
Week 1	Wed Sep 3	**Lecture 1: Introduction & Machine Learning Review** Course overview, review of fully connected networks, optimization, backpropagation	[slides]	FCV 9-14 What is a neural network? Gradient descent Backpropagation Backpropagation calculus (optional) Notes on backpropagation (optional) Least squares review (optional) SVD review	Assignment 1 out on MarkUs	
	Fri Sep 5	**Tutorial 1** PyTorch tutorial, Jupyter Notebook tutorial, lecture 1 practice problems, A1 Q&A				
Week 2	Wed Sep 10	**Lecture 2: Recognition** Convolutional neural networks and sequence models for segmentation, object detection, and classification	[slides]	FCV 24 FCV 25 FCV 50 (optional) Hubel & Wiesel 1979		
	Fri Sep 12	**Tutorial 2** PyTorch Tutorial (CNNs), fine-tuning, logging tools, lecture 2 practice problems, A1 Q&A				
Week 3	Wed Sep 17	**Lecture 3: Vision Transformers** attention, transformers, vision transformers, CLIP, autoregressive text-to-image	[slides]	FCV 26 FCV 51		
	Thu Sep 18					Ethics Survey 1 due
	Fri Sep	**Tutorial 3** understanding & visualizing				

https://www.cs.toronto.edu/~lindell/teaching/420/

Outline

- motivation
- syllabus
- course overview
- fully-connected networks & neural fields
- backpropagation

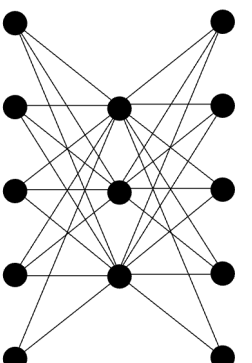
Course Overview

Course Overview

Machine Learning Review (today)

How do we train a neural network for classification or signal representation?

Images



Class

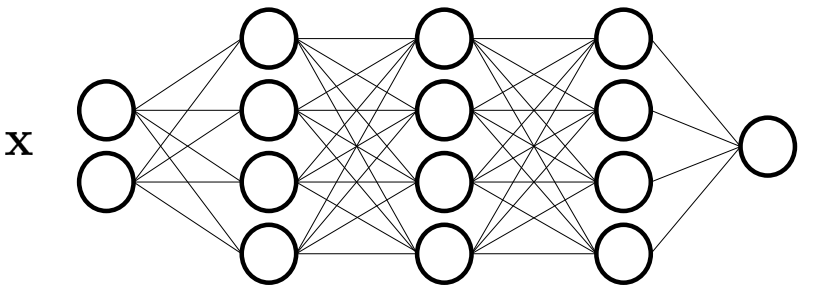
"zero"
"one"

...

"nine"

Machine Learning Review (today)

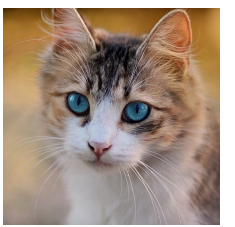
How do we train a neural network for classification or signal representation?



$$\Phi(\mathbf{x}) : \mathbb{R}^2 \rightarrow \mathbb{R}^3$$

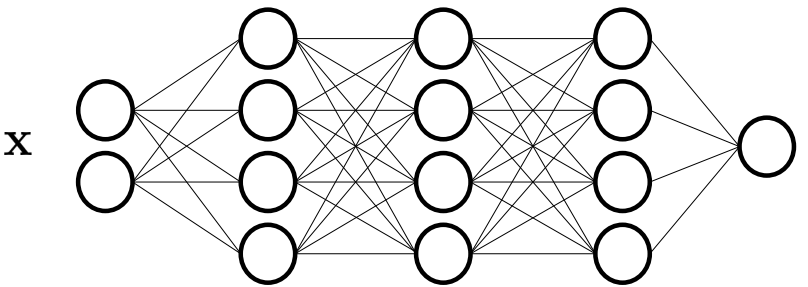
pixel coordinates

pixel values



Machine Learning Review (today)

How do we train a neural network for classification or signal representation?

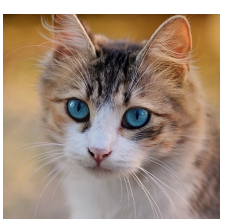


$$\Phi(\mathbf{x}) : \mathbb{R}^2 \rightarrow \mathbb{R}$$

pixel coordinates

pixel values

Why would we want to use this vs. a pixel array representation?



Recognition

Visual information is complicated and nuanced...

These are all dogs!

[Slide: R. Urtasun]

Recognition

Recognition

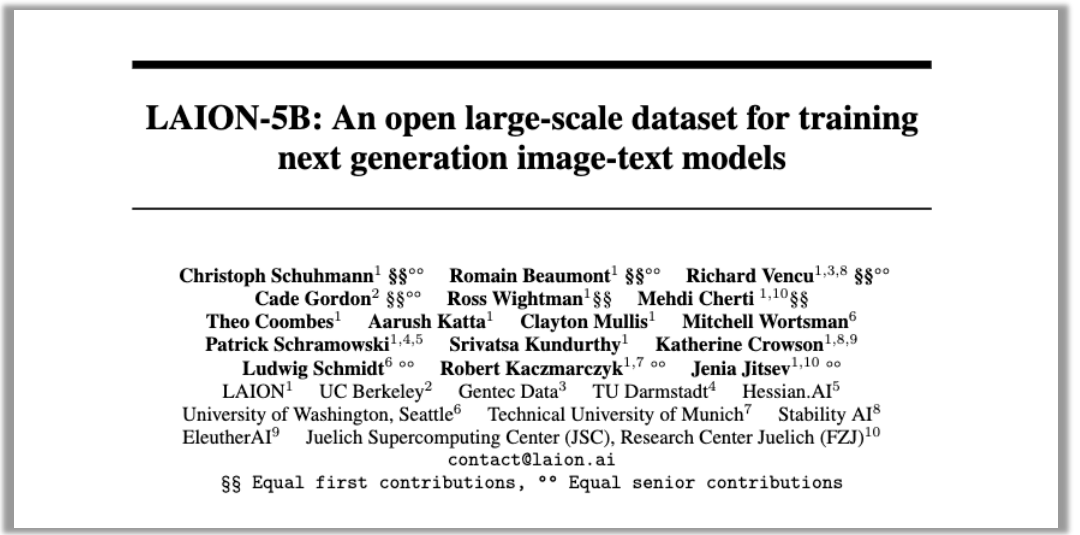
Recognition

Biederman, 1987

[Slide: R. Urtasun]

Vision Transformers

How do we make models that can scale to billions of input images?

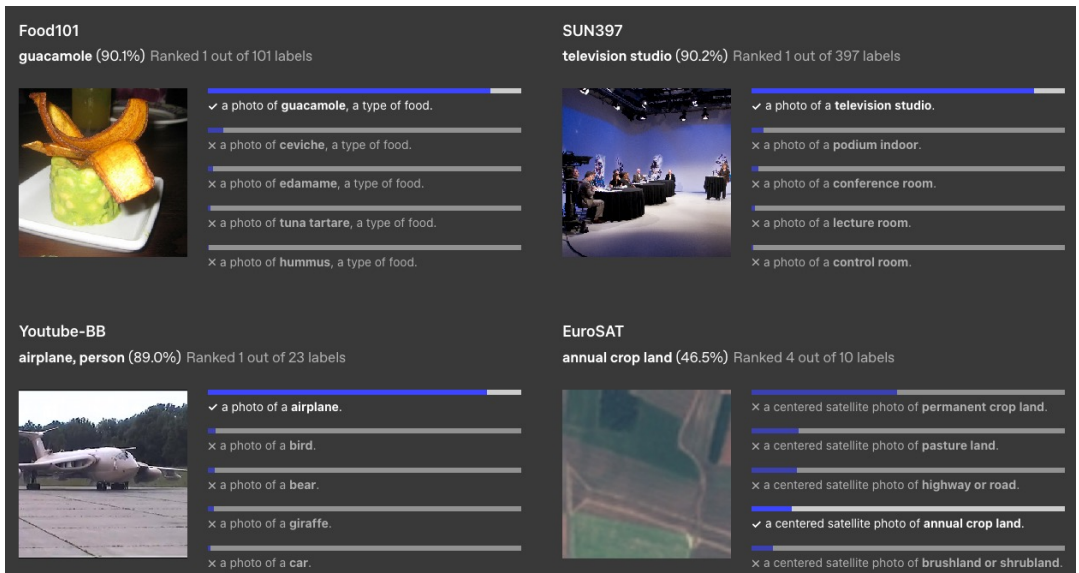


LAION-5B: An open large-scale dataset for training next generation image-text models

Christoph Schuhmann[1] §§°° **Romain Beaumont**[1] §§°° **Richard Vencu**[1,3,8] §§°°
Cade Gordon[2] §§°° **Ross Wightman**[1] §§ **Mehdi Cherti** [1,10] §§
Theo Coombes[1] **Aarush Katta**[1] **Clayton Mullis**[1] **Mitchell Wortsman**[6]
Patrick Schramowski[1,4,5] **Srivatsa Kundurthy**[1] **Katherine Crowson**[1,8,9]
Ludwig Schmidt[6] °° **Robert Kaczmarczyk**[1,7] °° **Jenia Jitsev**[1,10] °°
LAION[1] UC Berkeley[2] Gentec Data[3] TU Darmstadt[4] Hessian.AI[5]
University of Washington, Seattle[6] Technical University of Munich[7] Stability AI[8]
EleutherAI[9] Juelich Supercomputing Center (JSC), Research Center Juelich (FZJ)[10]
contact@laion.ai
§§ Equal first contributions, °° Equal senior contributions

Vision Transformers

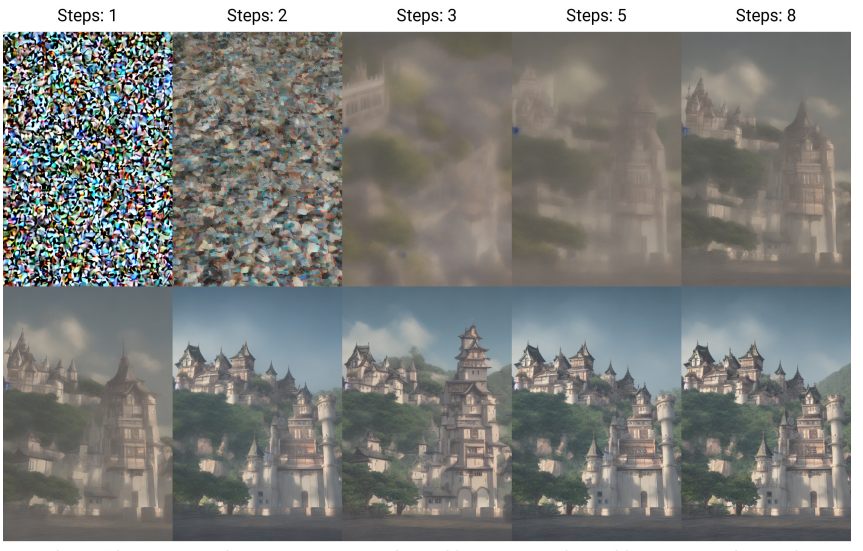
How do we make multi-modal models (e.g., text + visual information)?



https://openai.com/index/clip/

Generative Models

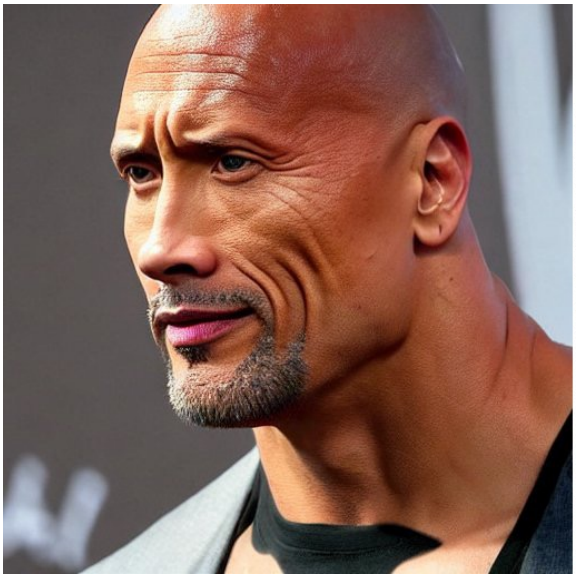
How can we generate photo-real images and what are the implications of doing so?



Benlisquare (Wikipedia)

Generative Models

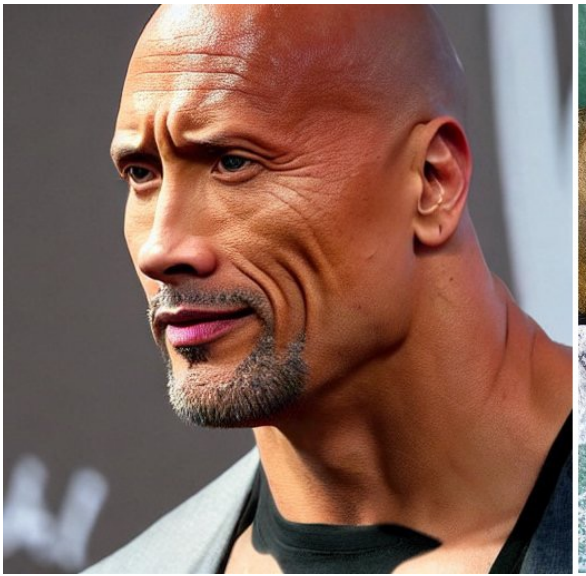
Generate an image from a caption (stable diffusion)



"Dwayne Johnson side view"

Generative Models

Generate an image from a caption (stable diffusion)



"Dwayne Johnson side view"

"Dwayne Johnson top view"


Course Overview
Part II
Dynamic Vision
Machine Learning
Digital Photography
6. radiometry & photometry
5. digital photoraphy
2. CNNs
3D Vision

Digital Photography

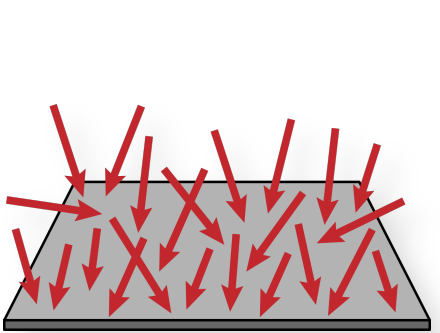
How does a digital camera capture an image?

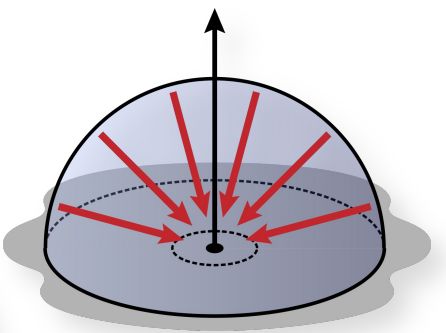
optics
aperture
depth of field
field of view
exposure
noise
color filter arrays

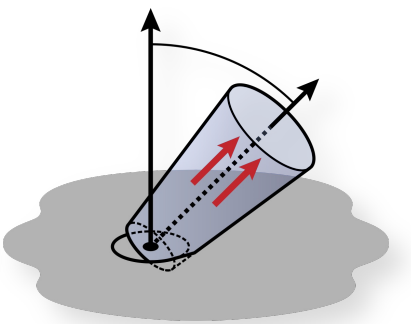
Course Overview

Radiometry & Photometric Cues

How do we model how light interacts with an environment?

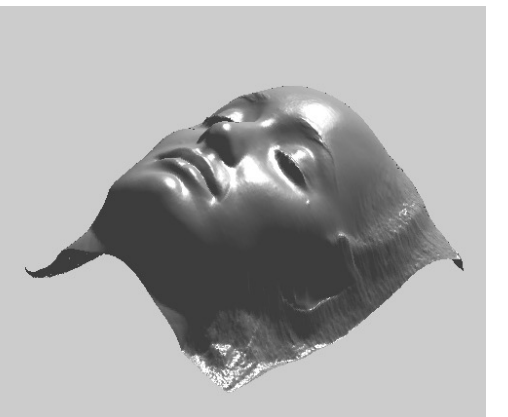






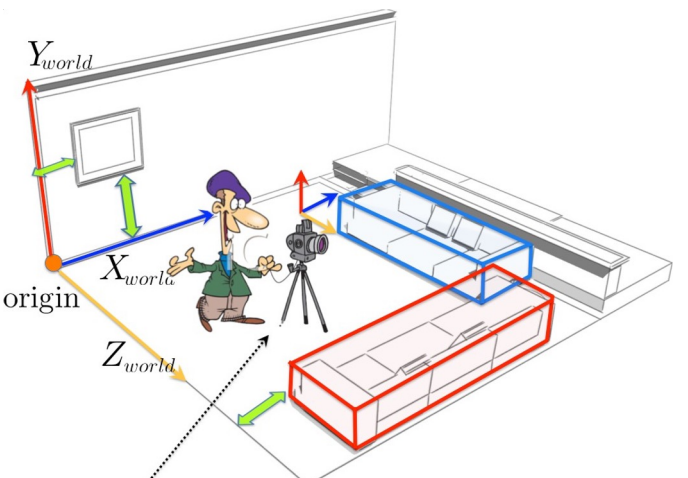
Radiometry & Photometric Cues

...and how can we use such models to recover 3D shape?



3D Vision & Stereo I

How do we model how 3D points project onto a captured image?



But to project my room to camera, I need to have the room in the camera coordinate system!

Epipolar Geometry & Stereo II

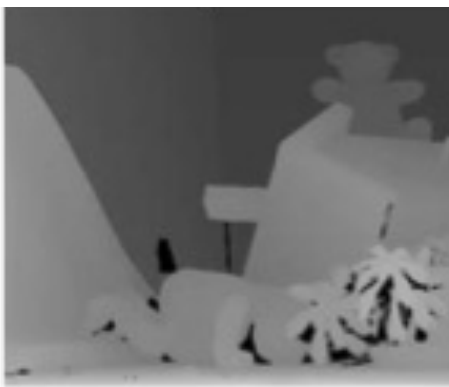
What can I infer about a scene if I have stereo images?

Epipolar Geometry & Stereo II

What can I infer about a scene if I have stereo images?

Epipolar Geometry & Stereo II

What can I infer about a scene if I have stereo images?



Structure from Motion & Radiance Fields

What about many images?

Building Rome in a Day
Agarwal et al. 2009

Structure from Motion & Radiance Fields

What about many images?

Zip-NeRF
Barron et al. ‘23


Course Overview
Part IV
11. modeling humans
Dynamic Vision
10. optical flow & tracking
Machine Learning
Digital Photography
3D Vision

Optical Flow & Tracking

What does the bottom video tell us about scene motion?

Optical Flow & Tracking

How is that different from this video?

Harley et al. ‘25

Modeling Humans

What techniques can we use to recover 3D geometry of humans from a single image?

Taubner et al. ‘25

Course Overview

- Part I: Machine learning for vision
  - fully-connected networks
  - CNNs
  - vision transformers
  - generative models
- Part II: Digital photography
  - camera ISPs
  - ray optics
- Part III: 3D vision
  - radiometry & photometric cues
  - epipolar geometry & depth from stereo
  - structure from motion & radiance fields
- Part IV: Dynamic vision
  - optical flow & tracking
  - modeling humans

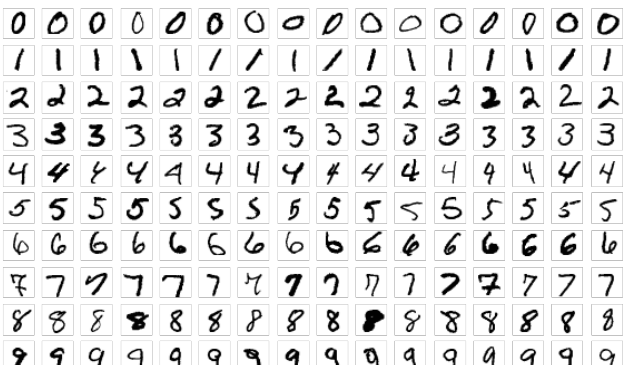
Outline

- motivation
- syllabus
- course overview
- fully-connected networks & neural fields
- backpropagation

Image Classification

- Image classification example

Images

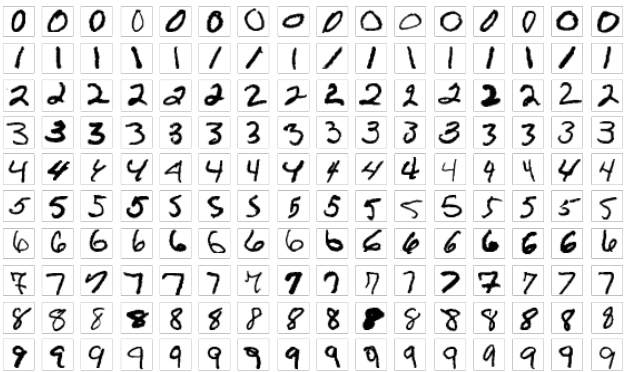


MNIST Dataset

Image Classification

- Image classification example

Images



Class

“zero”
“one”

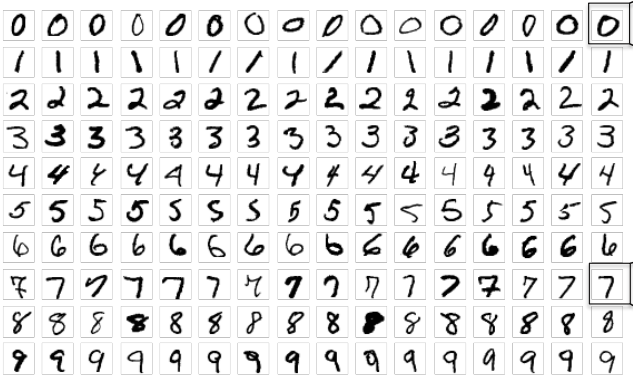
…

“nine”

Image Classification

- Image classification example

Images



What the computer “sees”

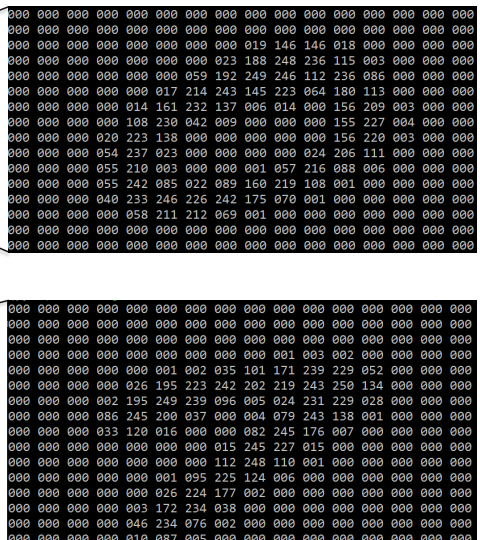
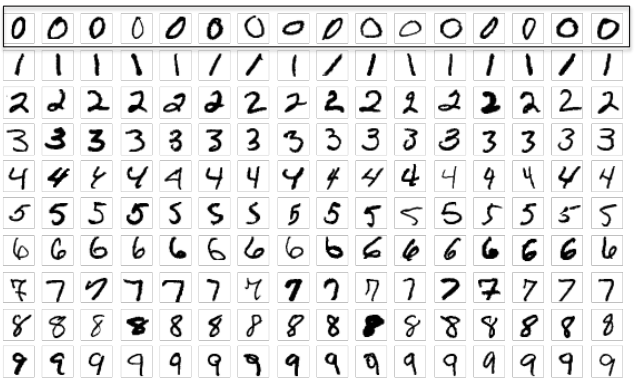


Image Classification

- Image classification example

Images



Challenges

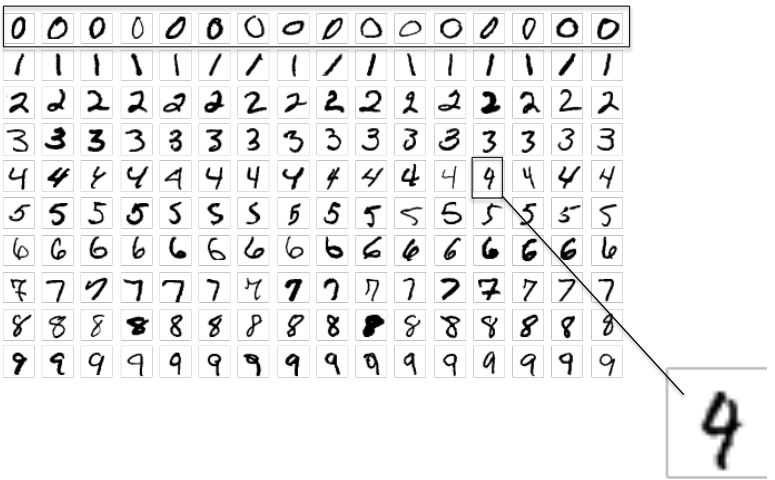
Intra-class variation

- stroke widths
- alignment
- writing styles

Image Classification

- Image classification example

Images



Challenges

Intra-class variation

- stroke widths
- alignment
- writing styles

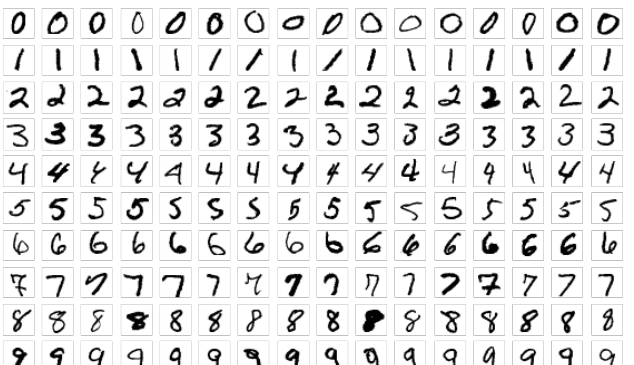
Inter-class similarities

- “four” or “nine”?

Image Classification

- Image classification example

Images



Implementation?

```
def classify_digit(image):
    # ???
    return image_class
```

Can’t hardcode solution!

Image Classification

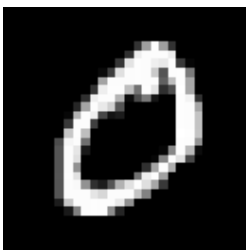
- Data-driven approach
  - Collect training images and labels
  - Train a classifier using machine learning
  - Evaluate the classifier on unseen images

Implementation?

```
1 def train(images, labels):
2     # machine learning model
3     return image_class
4
5 def evaluate(model, test_images):
6     # machine learning model
7     return test_labels
8
```

Image Classification

- Linear Model $f(x, W) = Wx$



vectorize

x

Image Classification

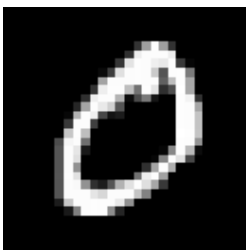
- Linear Model $f(x, W) = Wx$

vectorize

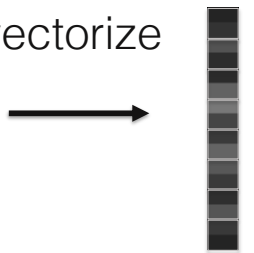
x

Image Classification

- Linear Model $f(x, W) = Wx$



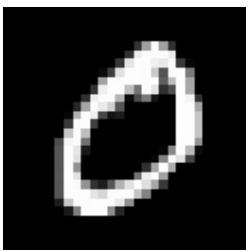
vectorize



x

Image Classification

- Linear Model $f(x, W) = Wx$



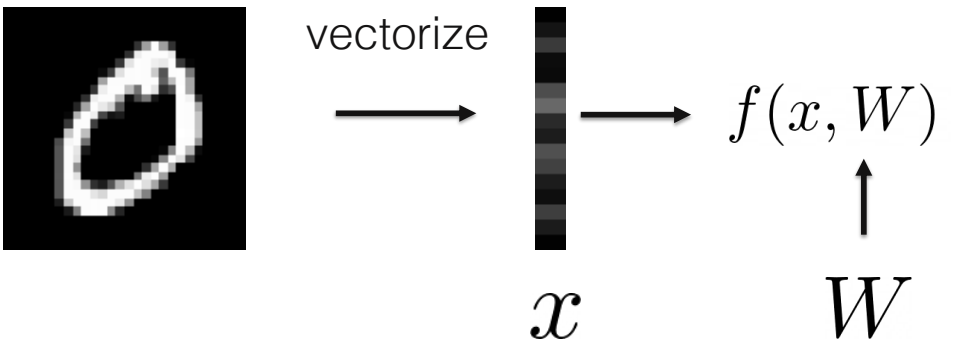
vectorize

x

Length of this vector is the “dimensionality” of our problem!

Image Classification

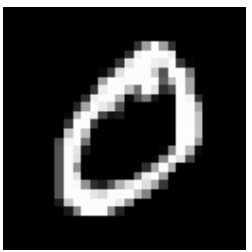
- Linear Model $f(x, W) = Wx$



In general: $Wx + b$

Image Classification

- Linear Model $f(x, W) = Wx$



vectorize

x

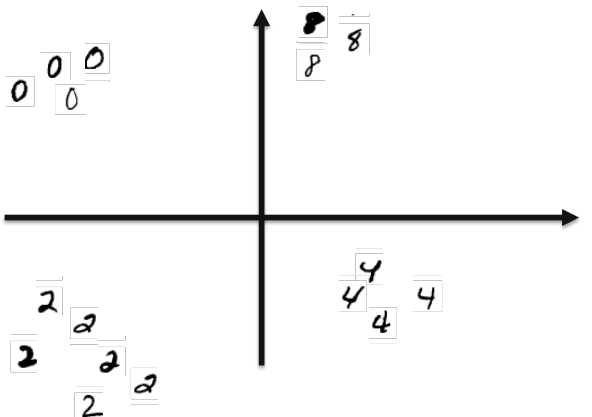
$f(x, W)$

W

10 numbers with class scores

Image Classification

- Linear model: geometric intrepretation

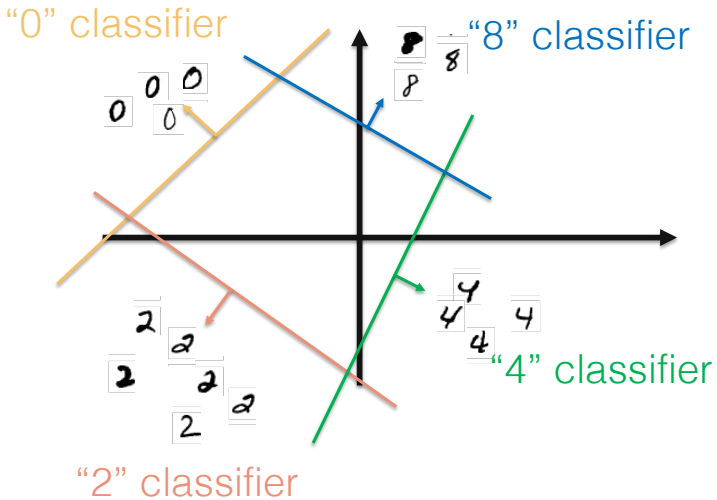


Each image is a point in an N-dimensional space

- N is the number of pixels

Image Classification

- Linear model: geometric interpretation



$$f(x, W) = Wx$$

Computes inner product between rows of W and x!

- Each row of W is a hyperplane
- Sign of inner product tells you which side of the hyperplane
- "separates" the digits

Image Classification

- Linear model (visual interpretation)

Learned filters (rows of W)

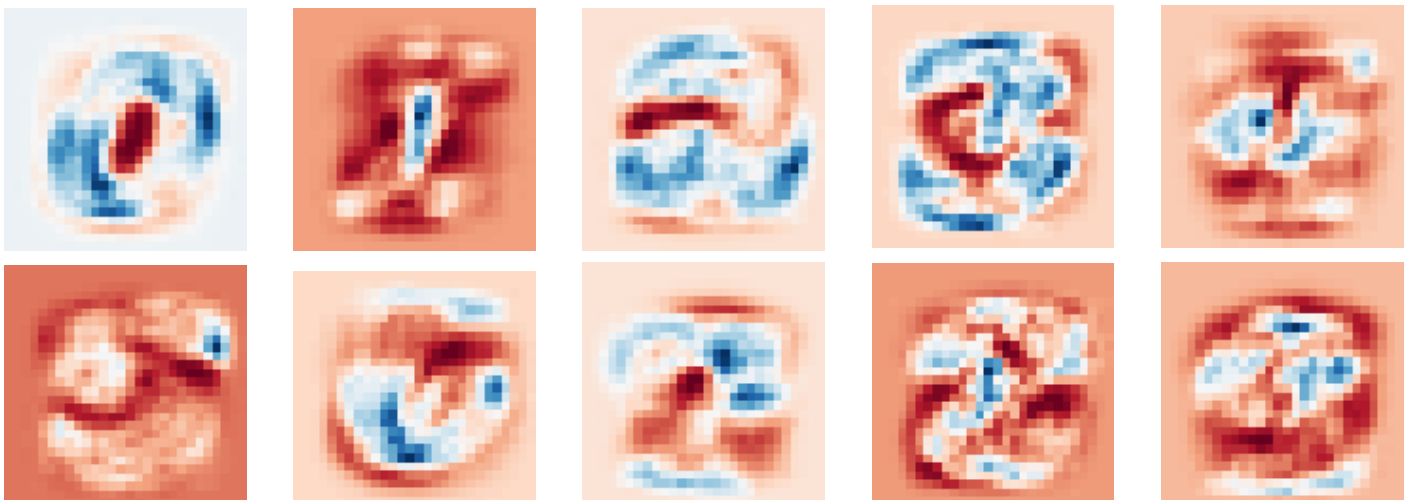
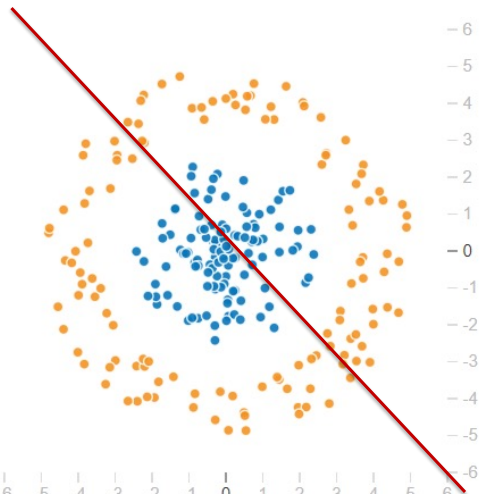


Image Classification

- Limits of linear classifiers

Linear classifiers learn linear decision planes

What if dataset is not linearly separable?



Multilayer Perceptrons (MLPs)

- Linear Model $f = Wx$
- 2-layer MLP $f = W_2 \max(0, W_1 x)$

Multilayer Perceptrons (MLPs)

- Linear Model $f = Wx$
- 2-layer MLP $f = W_2 \max(0, W_1 x)$
- 3-layer MLP $f = W_3 \max(0, W_2 \max(0, W_1 x))$

Multilayer Perceptrons (MLPs)

- Linear Model $f = Wx$
- 2-layer MLP $f = W_2 \max(0, W_1 x)$
- 3-layer MLP $f = W_3 \max(0, W_2 \max(0, W_1 x))$

Non-linearity/activation function between linear layers

Multilayer Perceptrons (MLPs)

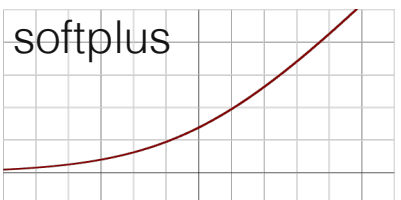
- Linear Model $f = Wx$
- 2-layer MLP $f = W_2 \max(0, W_1 x)$
- 3-layer MLP $f = W_3 \max(0, W_2 \max(0, W_1 x))$

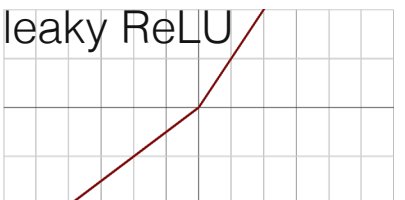
Otherwise we have:

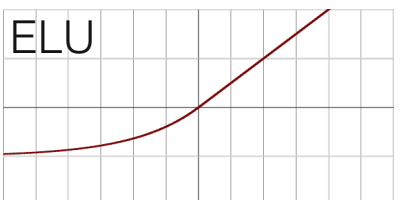
$$f = W_3 W_2 W_1 x$$

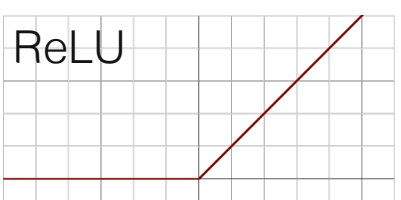
Activation Functions

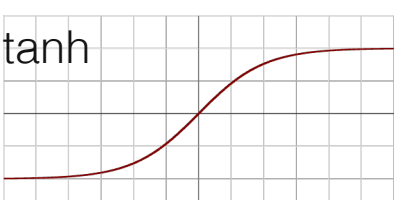
…many to choose from

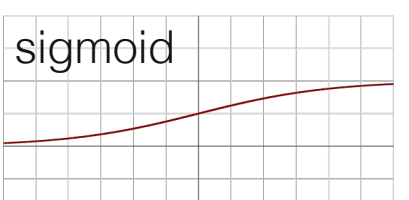










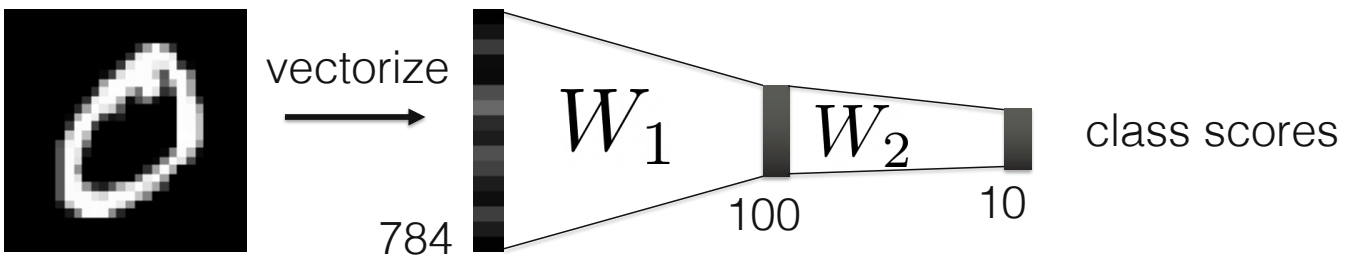


… ReLU is a good general-purpose choice: ReLU(x) = max(0, x)

Multilayer Perceptrons (MLPs)

- Linear Model $f = Wx$
- 2-layer MLP $f = W_2 \max(0, W_1 x)$

Back to our classification example…

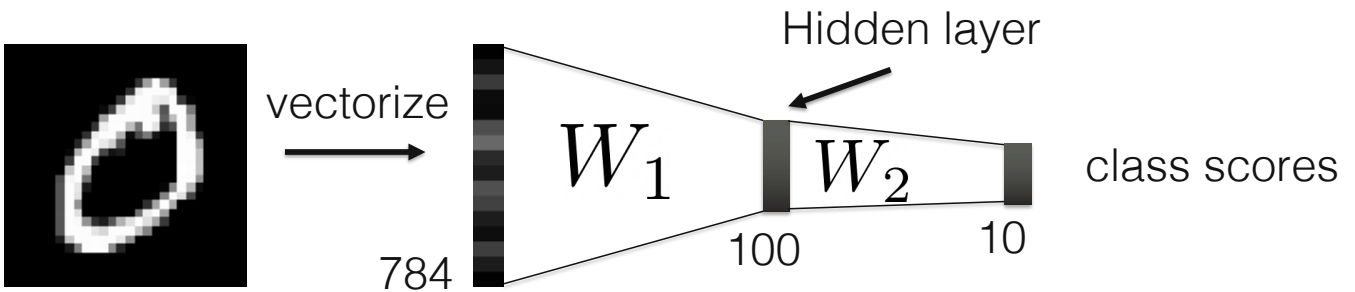


$x \in \mathbb{R}^D$, $W_1 \in \mathbb{R}^{H \times D}$, $W_2 \in \mathbb{R}^{C \times H}$

Multilayer Perceptrons (MLPs)

- Linear Model $f = Wx$
- 2-layer MLP $f = W_2 \max(0, W_1 x)$

Back to our classification example…

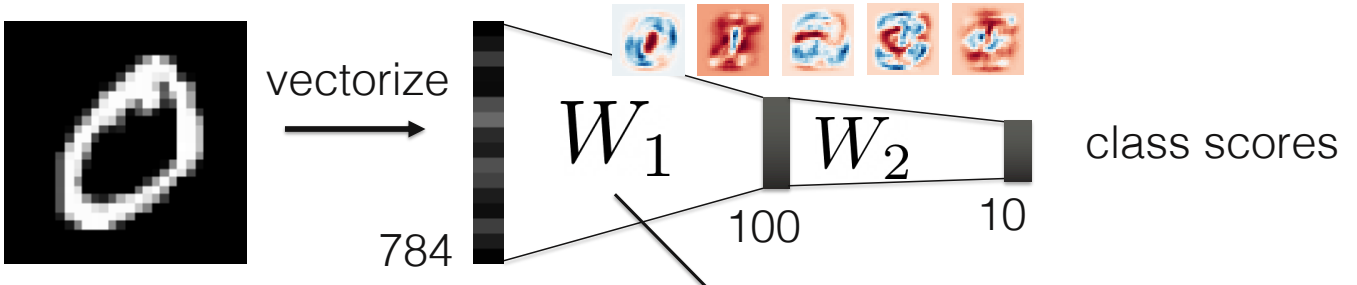


$x \in \mathbb{R}^D$, $W_1 \in \mathbb{R}^{H \times D}$, $W_2 \in \mathbb{R}^{C \times H}$

Multilayer Perceptrons (MLPs)

- Linear Model $f = Wx$
- 2-layer MLP $f = W_2 \max(0, W_1 x)$

Back to our classification example…

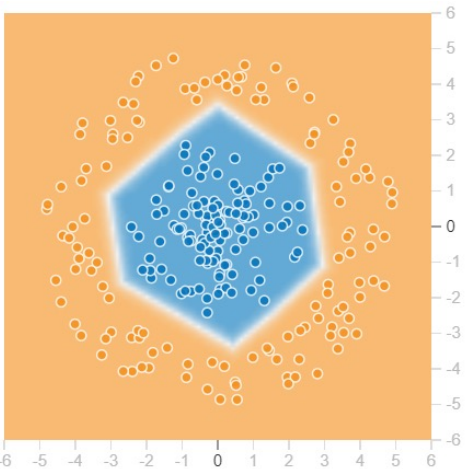


Now we have 100 shape templates, shared between classes

Multilayer Perceptrons (MLPs)

- Overcomes limits of linear classifiers

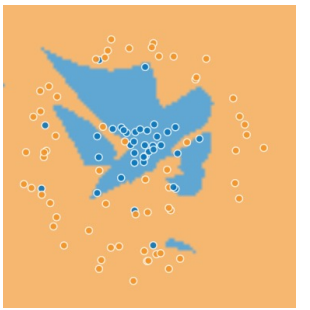
  - Can learn non-linear decision boundaries
  - Complexity scales with the number of neurons/hidden layers



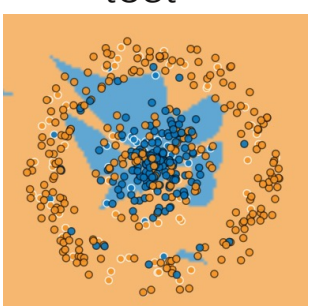
Multilayer Perceptrons (MLPs)

- More parameters is not always better!
  - Can lead to overfitting the training data
  - Performance on test data is worse

train

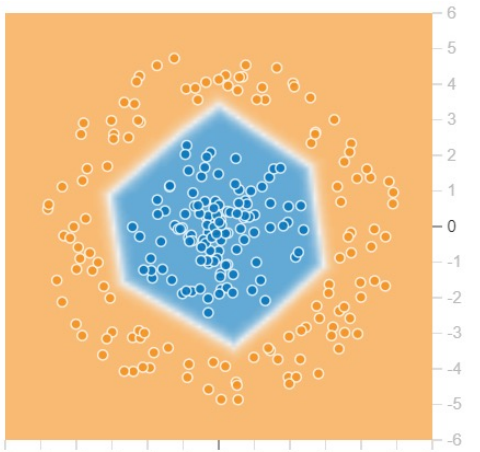


test

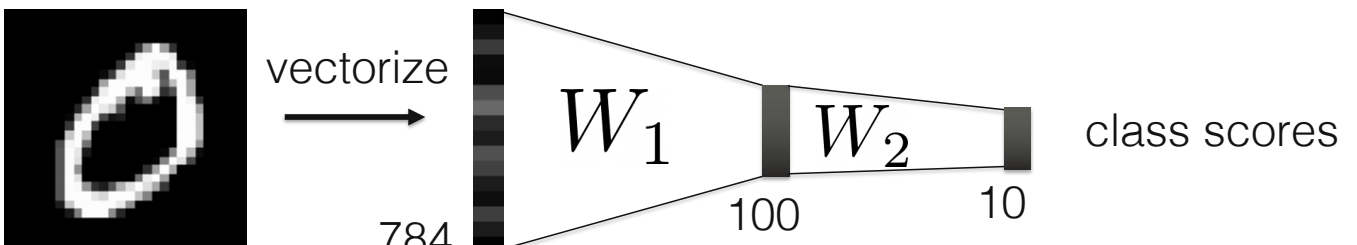


Multilayer Perceptrons (MLPs)

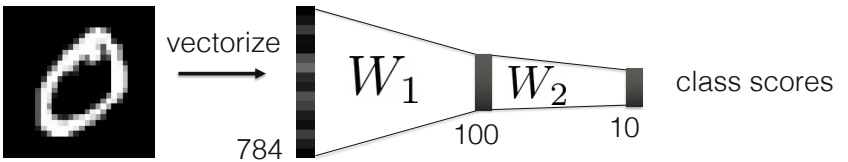
- More on classification…
  - https://cs231n.github.io/linear-classify/
  - https://amfarahmand.github.io/NN-Winter2024/

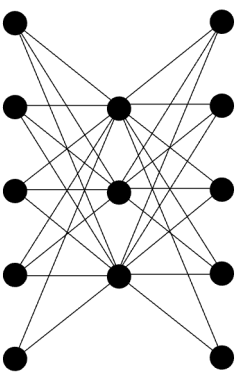


Why “neural” network?

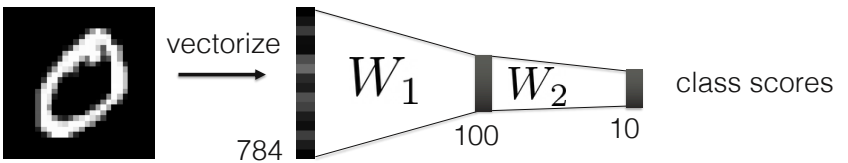


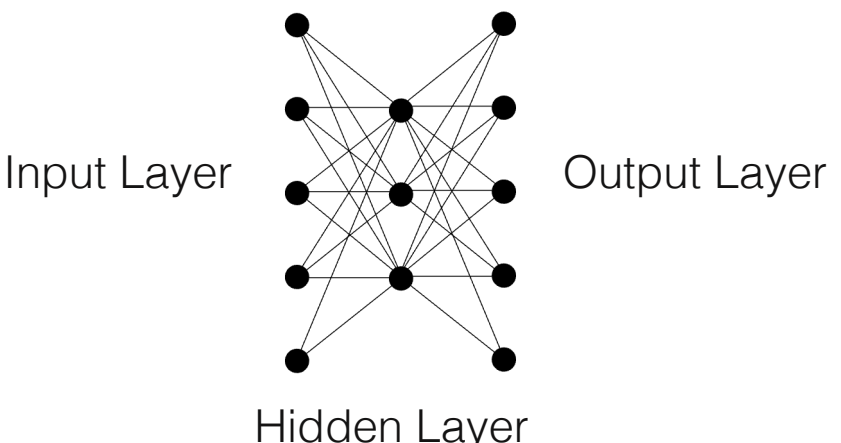
Why “neural” network?



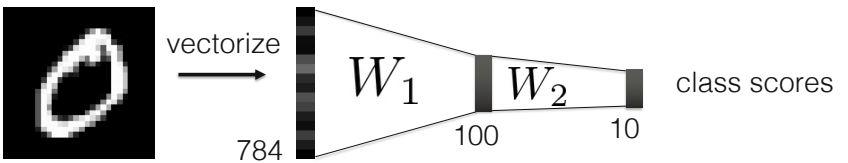


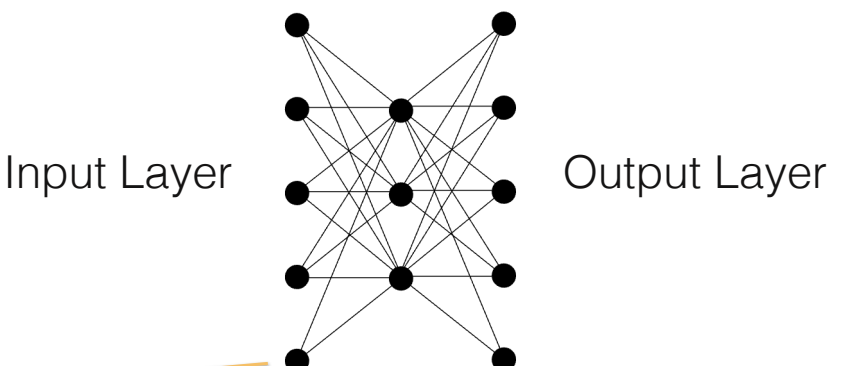
Why “neural” network?



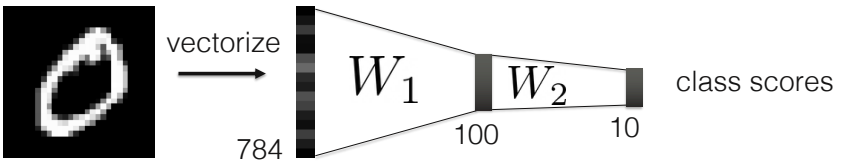


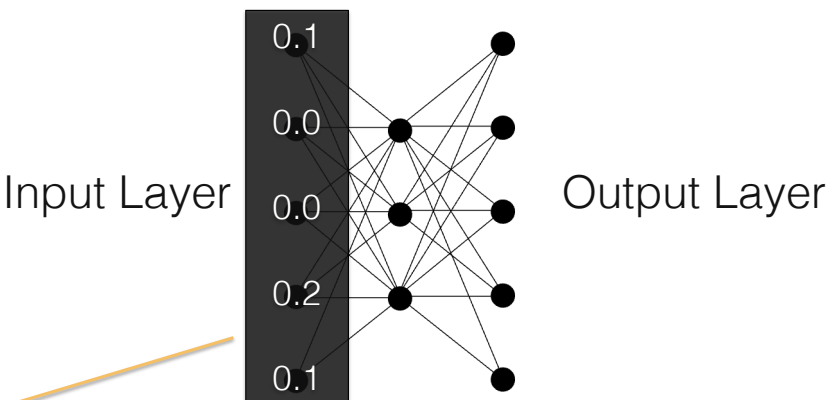
Why “neural” network?



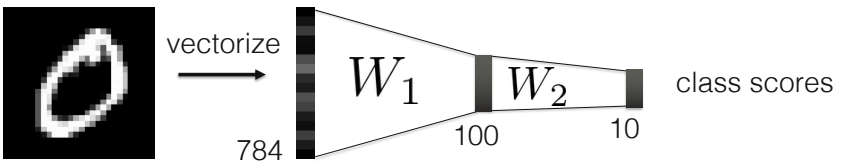


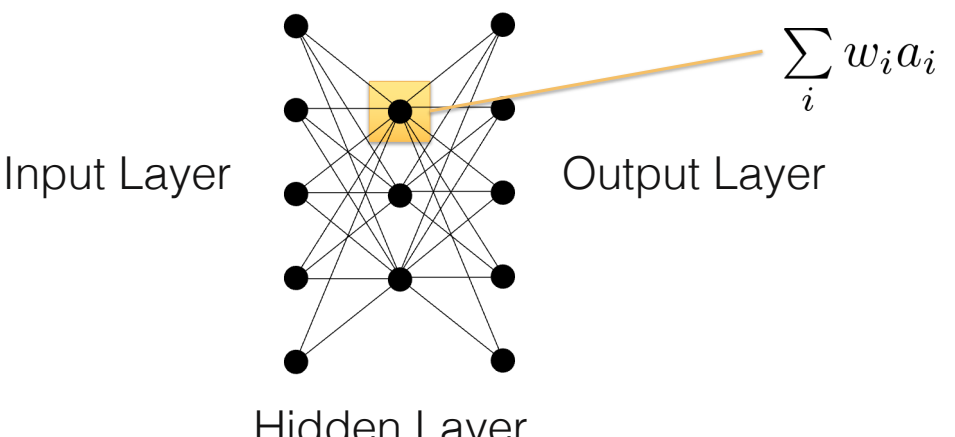
Why “neural” network?

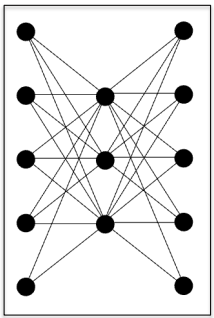


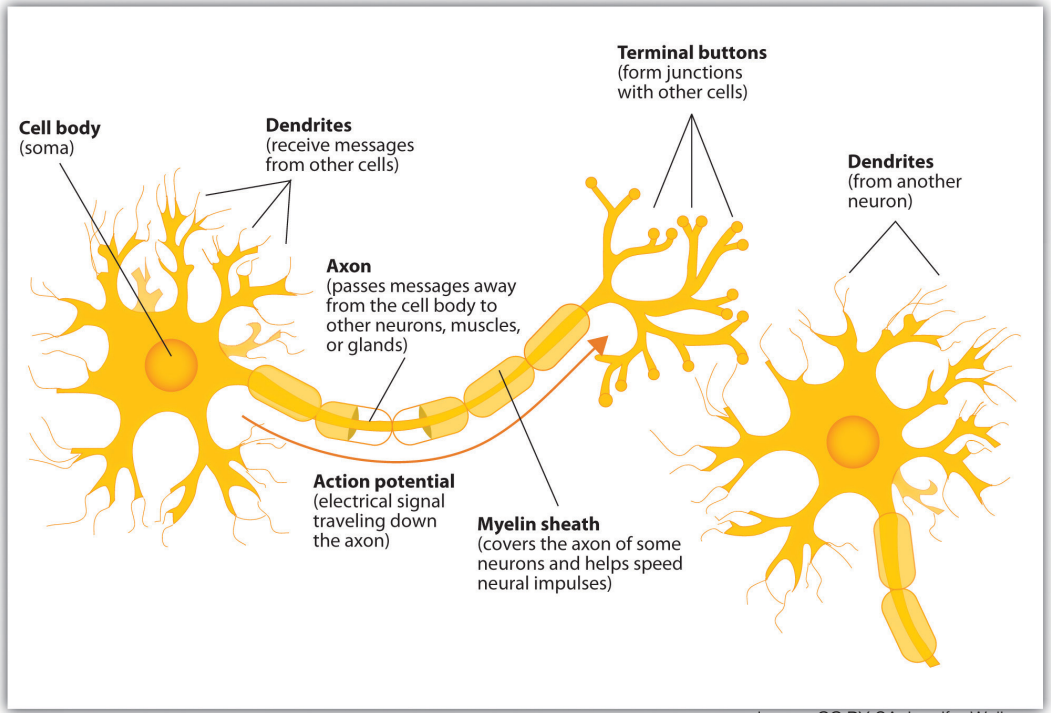


Why “neural” network?



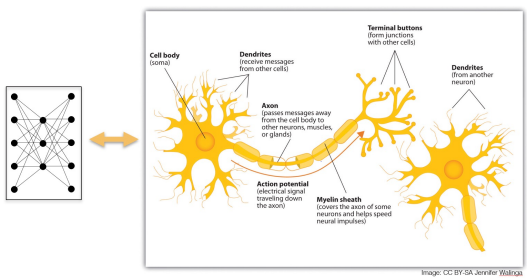




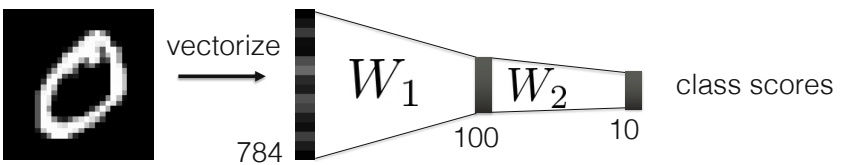


Loose analogy!

- Neurons have activation potentials, all-or-none firing behavior
- Interconnectivity between actual neurons is dense and complicated
- Connection between neurons is complex non-linear dynamical system

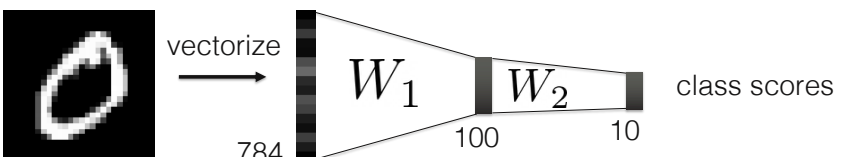


Drawbacks of fully-connected networks



- spatial structure is destroyed
- fully-connected weights do not scale

Drawbacks of fully-connected networks



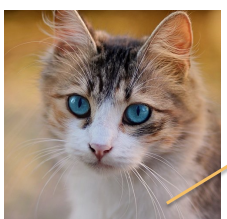
- spatial structure is destroyed

- fully-connected weights do not scale
- how do we fix this? — will talk about this more next lecture

Outline

- motivation
- syllabus
- course overview
- fully-connected networks & neural fields
- backpropagation

Neural field representations



pixel coordinate

(x, y)

2

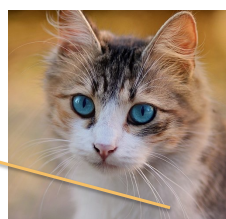
W_1

100

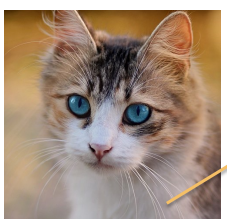
W_2

3

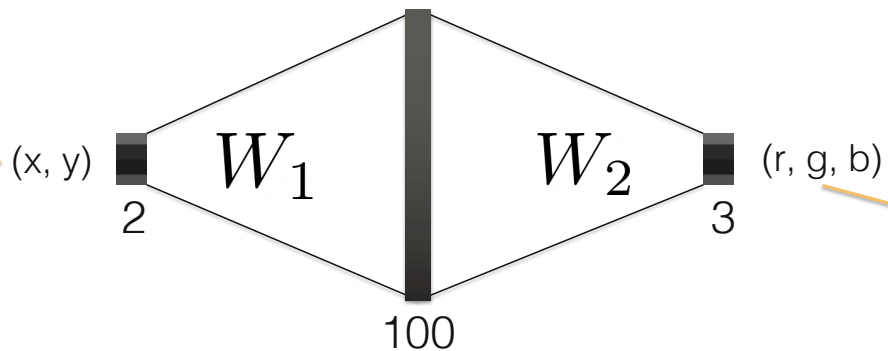
(r, g, b)

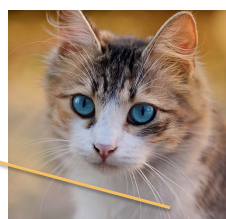


Neural field representations



pixel coordinate





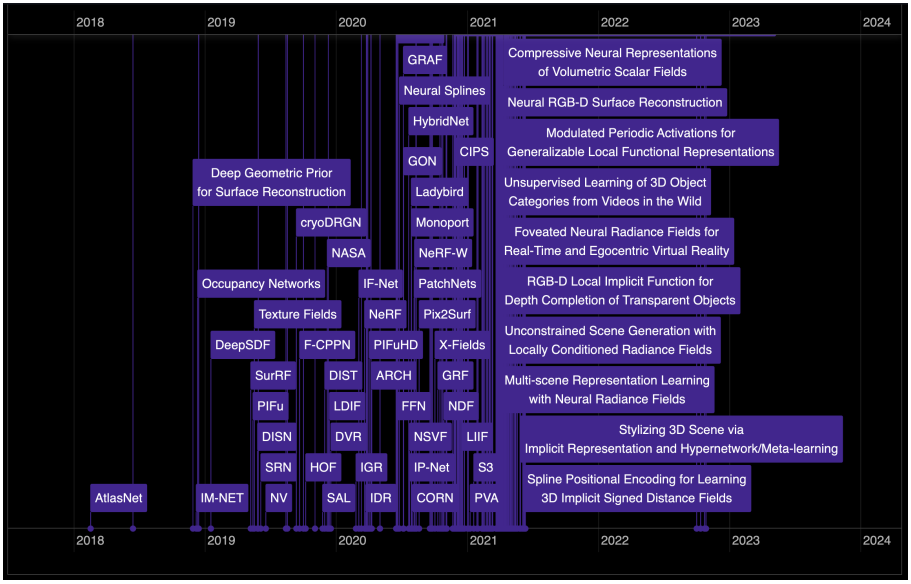
Field: a set of numbers along with mathematical operations defined on that set.

CVPR 2022 workshop: "a scalar or vector-valued quantity defined across an input domain"

- Neural field: "a field that is parameterized partly or fully by a neural network"

Neural field explosion!

https://neuralfields.cs.brown.edu/



Began around 2019 and accelerated since 2020

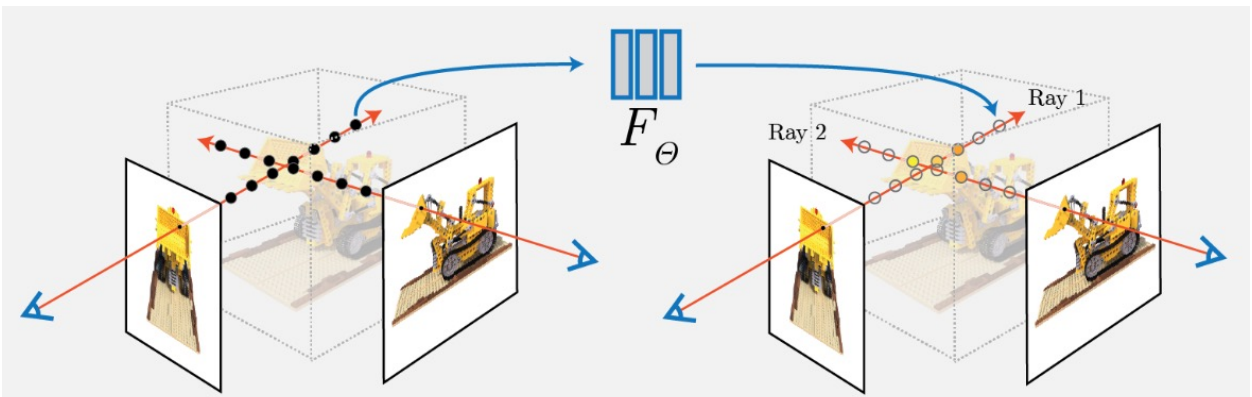
Neural field representations

- compact representations
- differentiable & easy to optimize
- multi-modal and easy to scale to high dimensions

Neural field representations

This scene is stored in about 30 MB of trained network weights
(optimized from ~2 GB of image pixels)

Neural field representations



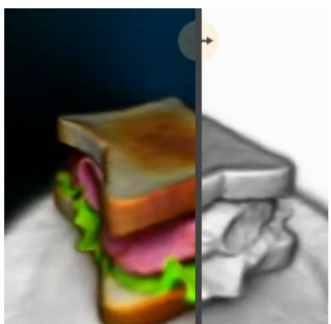
Becomes trivial to reconstruct 3D appearance and geometry from multiview imagery

Neural field representations

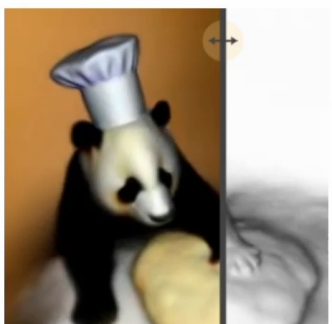
Can learn priors/generative models over a space of signals

[Chan et al. ‘22]

Neural field representations



[...] an overstuffed pastrami sandwich



[...] a panda wearing a chef's hat and kneading bread dough on a countertop

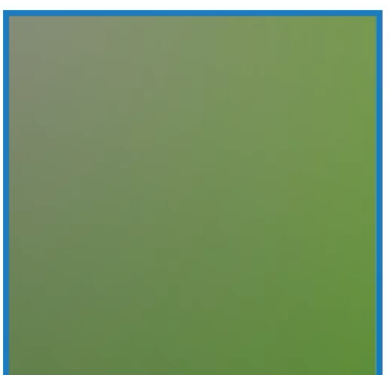
Luminescent wild horses

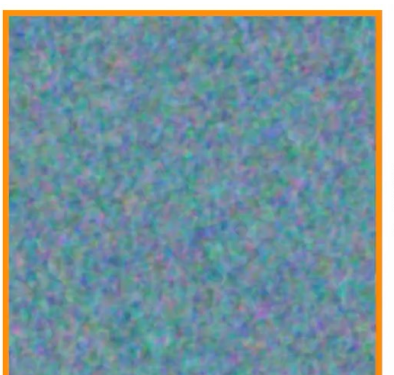
A model for text to 3D shape and appearance based on neural fields

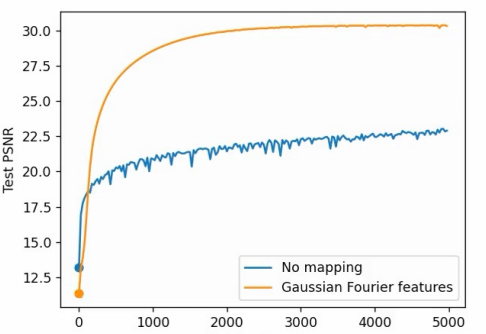
[Poole et al. ‘22]

Positional encoding

Toy example of image fitting







Without Pos. Enc.

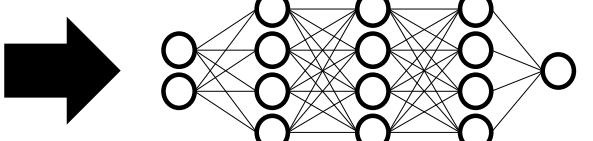
With Pos. Enc

[Tancik et al. '20]

Positional encoding

- Simple trick!
  - Instead of passing in v =(x, y) into the network we pass

$$\begin{bmatrix} \cos(\pi v) & \sin(\pi v) \\ \cos(2\pi v) & \sin(2\pi v) \\ \cos(4\pi v) & \sin(4\pi v) \\ \vdots & \\ \cos(2^{L-1}\pi v) & \sin(2^{L-1}\pi v) \end{bmatrix}$$



Positional encoding

- Why does this work?

- Explained with theory from Neural Tangent Kernel
  - Training a network is similar to kernel regression (becomes closer as network layers become wider)
  - https://arxiv.org/abs/1806.07572

Positional encoding

- Kernel Regression

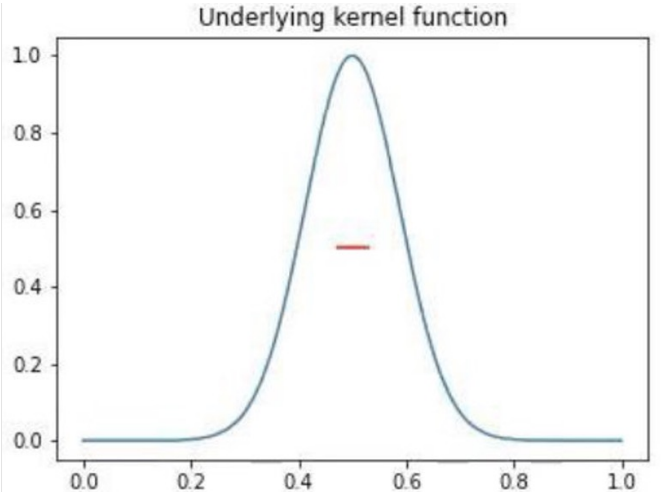
weighting & kernel function computes similarity between input and training points

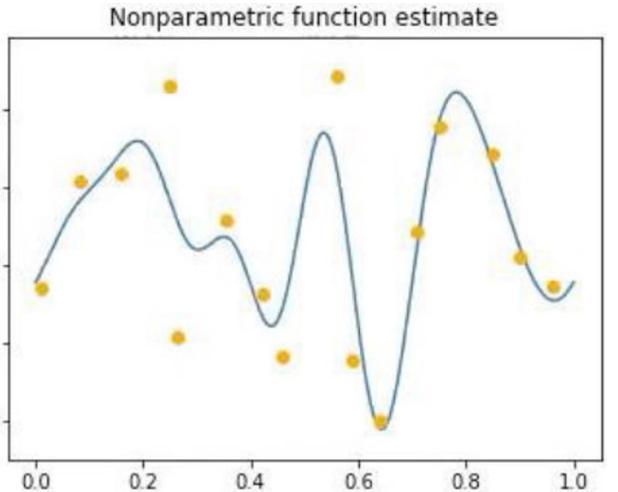
$$f(x) = \sum_{i=1}^{N} w_i k(x - x_i)$$

Sum over training points

e.g., if the kernel is a Gaussian, this puts a bump at every training data point

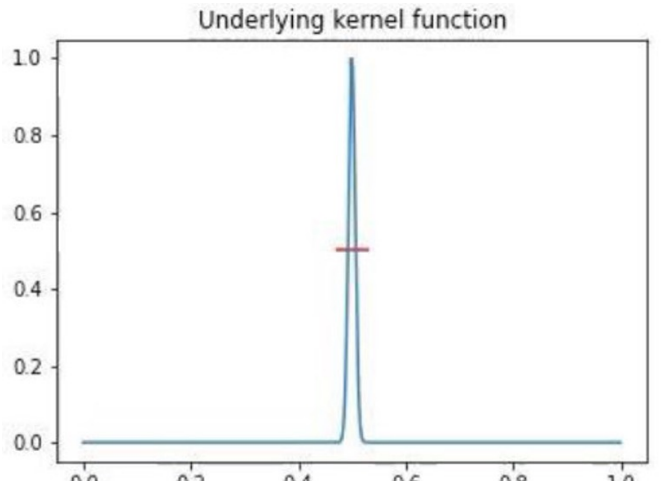
Positional encoding

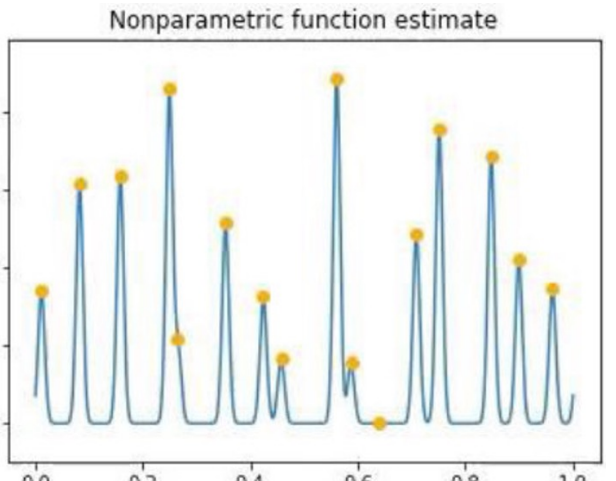




Width of kernel is important to trade off interpolation/overfitting to data!

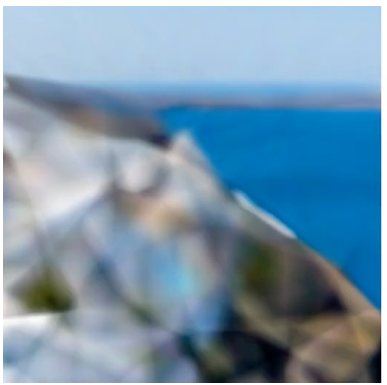
Positional encoding

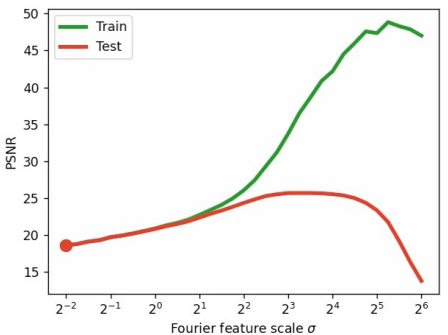


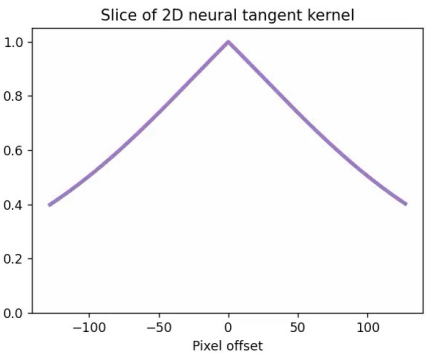


Width of kernel is important to trade off interpolation/overfitting to data!

Neural tangent kernel visualization





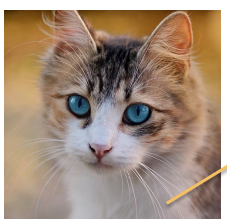


Width of kernel is important to trade off interpolation/overfitting to data!

You'll explore this more in your homework!

[Tancik et al. '20]

Neural field representations



pixel coordinate

(x, y)

2

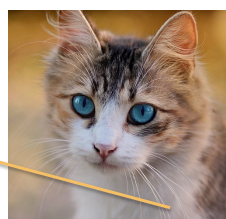
W_1

100

W_2

3

(r, g, b)

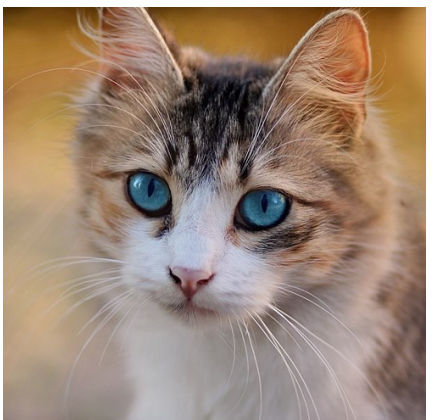


Training the neural field

Training dataset:

- pixel coordinates and rgb values
  - (x, y) -> (r, g, b) pairs

- train network to predict all pixel values!



Training the MLP

Train the network to minimize the loss function

$$\mathcal{L}_\theta = \frac{1}{2}\|y - \hat{y}\|_2^2$$

network parameters

$\theta = \{W_1, W_2\}$

GT pixel value

pred. pixel value

Training the MLP

How do we figure out θ ?

$$\mathcal{L}_\theta = \frac{1}{2}\|y - \hat{y}\|_2^2$$

network parameters

$\theta = \{W_1, W_2\}$

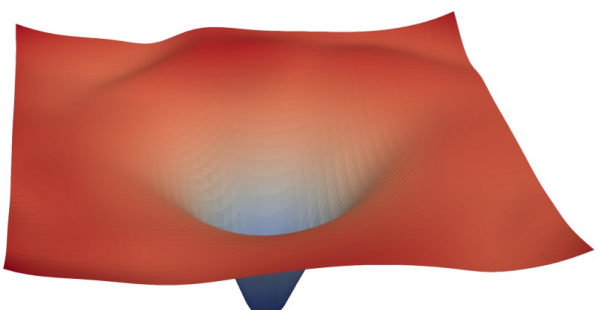
GT pixel value

pred. pixel value

Training the MLP

Gradient-based optimization

$$\theta^{(k+1)} = \theta^{(k)} - \nabla_\theta \mathcal{L}_\theta$$



Loss Landscape

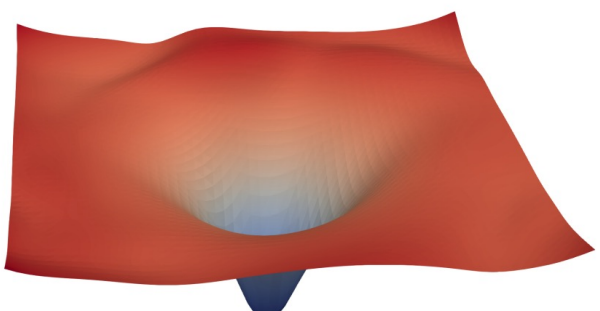
[Li et al. ‘18]

Training the MLP

Gradient-based optimization

$$\theta^{(k+1)} = \theta^{(k)} - \nabla_\theta \mathcal{L}_\theta$$

Need to calculate the partial derivative with respect to each parameter



Loss Landscape

[Li et al. ‘18]

Outline

- motivation
- syllabus
- course overview
- fully-connected networks & neural fields
- backpropagation

Training the MLP

Generally there are 3 options

1. Numerical differentiation
2. Symbolic differentiation
3. “Automatic” differentiation

Numerical Differentiation

$$\frac{\partial f(x)}{\partial x} \approx \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

Not very accurate, computationally expensive

Easy to implement! Can be used to check your analytical answers..

Symbolic Differentiation

$$\frac{\partial \mathcal{L}_\theta}{\partial W_1} = \frac{\partial}{\partial W_1} \frac{1}{2} \|y - \hat{y}\|_2^2$$

$$= \frac{\partial}{\partial W_1} \frac{1}{2} \left(W_2 \sigma(W_1 x)\right)^T \left(W_2 \sigma(W_1 x)\right)$$

$$= \frac{\partial}{\partial W_1} \frac{1}{2} \sigma(W_1 x)^T W_2^T W_2 \sigma(W_1 x)$$

$= \ldots$ chain rule, product rule…

Accurate, but must be manually calculated for each term
Tedious!

Automatic Differentiation

Think about the problem as a "computational graph"

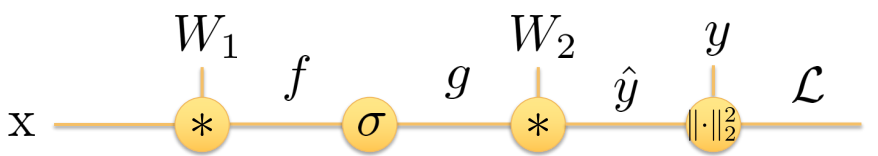
Divide and conquer using the chain rule

Enables "backpropagation" – an efficient way to take derivatives of all parameters in a computational graph

Automatic Differentiation

Think about the problem as a “computational graph”

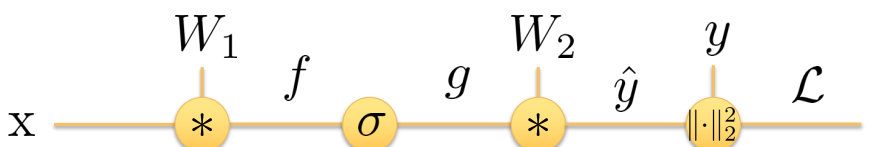
Divide and conquer using the chain rule



Automatic Differentiation

Think about the problem as a "computational graph"

Divide and conquer using the chain rule

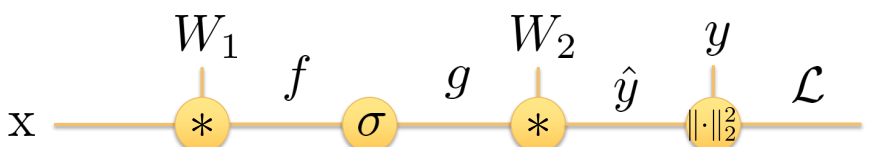


$$\frac{\partial \mathcal{L}}{\partial W_2} = \frac{\partial \hat{y}}{\partial W_2} \frac{\partial \mathcal{L}}{\partial \hat{y}}$$

Automatic Differentiation

Think about the problem as a "computational graph"

Divide and conquer using the chain rule

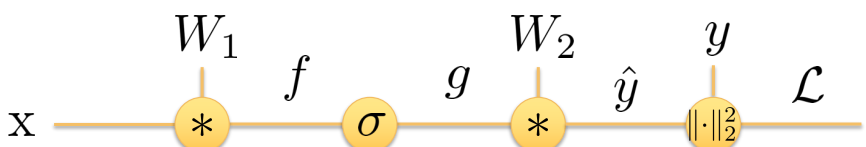


$$\frac{\partial \mathcal{L}}{\partial W_1} = \frac{\partial f}{\partial W_1} \frac{\partial g}{\partial f} \frac{\partial \hat{y}}{\partial g} \frac{\partial \mathcal{L}}{\partial \hat{y}}$$

Automatic Differentiation

Think about the problem as a "computational graph"

Divide and conquer using the chain rule

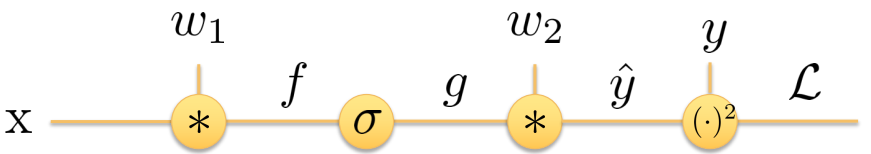


$$\frac{\partial \mathcal{L}}{\partial W_1} = \frac{\partial f}{\partial W_1} \frac{\partial g}{\partial f} \frac{\partial \hat{y}}{\partial g} \frac{\partial \mathcal{L}}{\partial \hat{y}}$$

We can calculate analytical expressions for each of these terms and then plug in our values

Autodiff Example

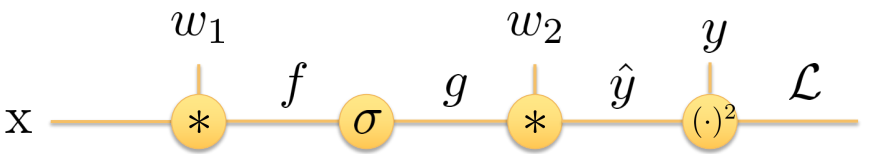
(assume scalar values for now)



$$\frac{\partial \mathcal{L}}{\partial w_1} = \frac{\partial f}{\partial w_1} \frac{\partial g}{\partial f} \frac{\partial \hat{y}}{\partial g} \boxed{\frac{\partial \mathcal{L}}{\partial \hat{y}}}$$

Autodiff Example

(assume scalar values for now)

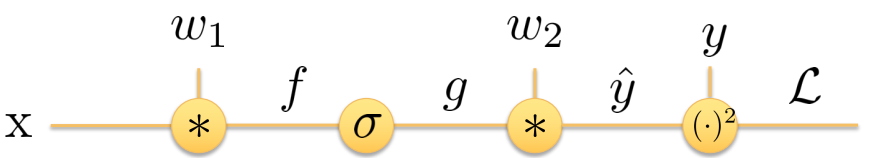


$$\frac{\partial \mathcal{L}}{\partial w_1} = \frac{\partial f}{\partial w_1}\frac{\partial g}{\partial f}\frac{\partial \hat{y}}{\partial g}\frac{\partial \mathcal{L}}{\partial \hat{y}}$$

$$\frac{\partial \mathcal{L}}{\partial \hat{y}} = \frac{\partial}{\partial \hat{y}}\frac{1}{2}(\hat{y} - y)^2 = \hat{y} - y$$

Autodiff Example

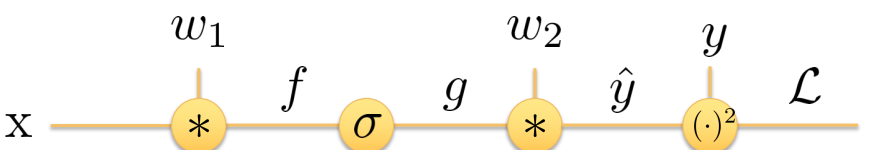
(assume scalar values for now)



$$\frac{\partial \mathcal{L}}{\partial w_1} = \frac{\partial f}{\partial w_1} \frac{\partial g}{\partial f} \frac{\partial \hat{y}}{\partial g} \frac{\partial \mathcal{L}}{\partial \hat{y}}$$

Autodiff Example

(assume scalar values for now)

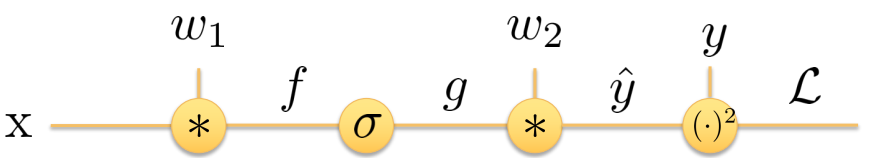


$$\frac{\partial \mathcal{L}}{\partial w_1} = \frac{\partial f}{\partial w_1}\frac{\partial g}{\partial f}\frac{\partial \hat{y}}{\partial g}\frac{\partial \mathcal{L}}{\partial \hat{y}}$$

$$\frac{\partial \hat{y}}{\partial g} = \frac{\partial}{\partial g} w_2 \cdot g = w_2$$

Autodiff Example

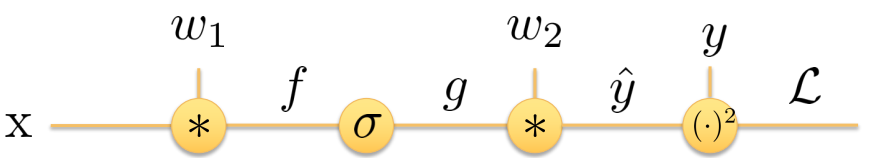
(assume scalar values for now)



$$\frac{\partial \mathcal{L}}{\partial w_1} = \frac{\partial f}{\partial w_1}\frac{\partial g}{\partial f}\frac{\partial \hat{y}}{\partial g}\frac{\partial \mathcal{L}}{\partial \hat{y}}$$

Autodiff Example

(assume scalar values for now)

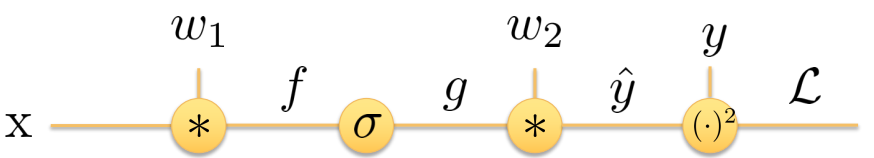


$$\frac{\partial \mathcal{L}}{\partial w_1} = \frac{\partial f}{\partial w_1}\frac{\partial g}{\partial f}\frac{\partial \hat{y}}{\partial g}\frac{\partial \mathcal{L}}{\partial \hat{y}}$$

$$\frac{\partial g}{\partial f} = \frac{\partial}{\partial f}\sigma(f) = \frac{\partial}{\partial f}\max(0, f) = \begin{cases} 0, & f < 0 \\ 1 & \text{else} \end{cases}$$

Autodiff Example

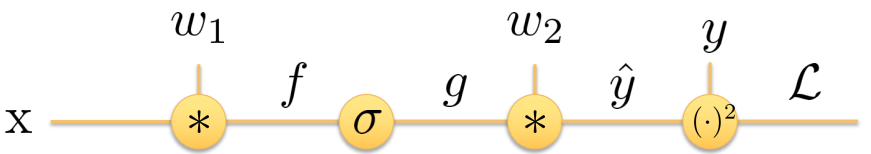
(assume scalar values for now)



$$\frac{\partial \mathcal{L}}{\partial w_1} = \frac{\partial f}{\partial w_1} \frac{\partial g}{\partial f} \frac{\partial \hat{y}}{\partial g} \frac{\partial \mathcal{L}}{\partial \hat{y}}$$

Autodiff Example

(assume scalar values for now)

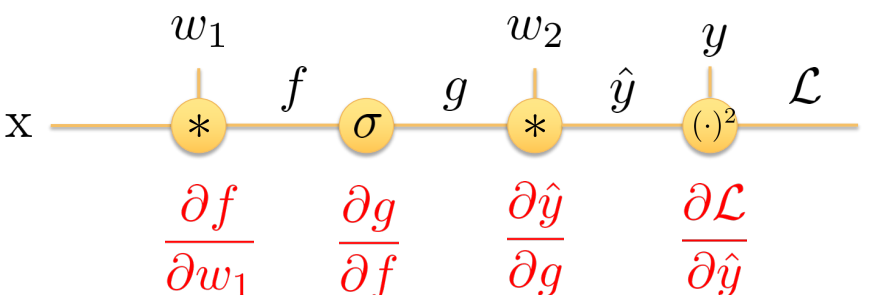


$$\frac{\partial \mathcal{L}}{\partial w_1} = \frac{\partial f}{\partial w_1} \frac{\partial g}{\partial f} \frac{\partial \hat{y}}{\partial g} \frac{\partial \mathcal{L}}{\partial \hat{y}}$$

$$\frac{\partial f}{\partial w_1} = \frac{\partial}{\partial w_1} w_1 \cdot x = x$$

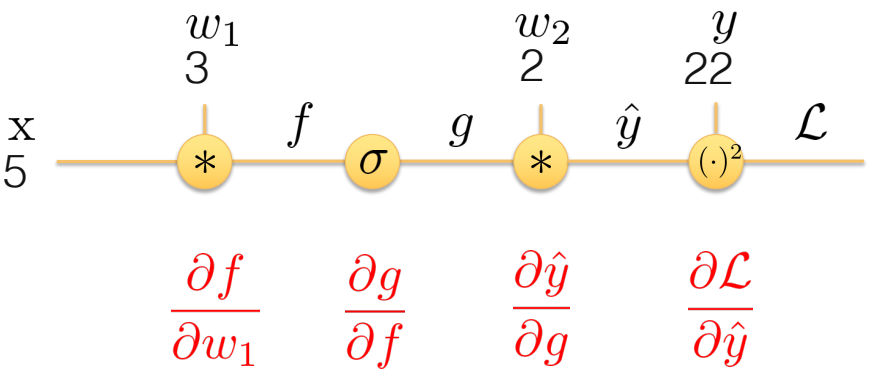
Autodiff Example

(assume scalar values for now)



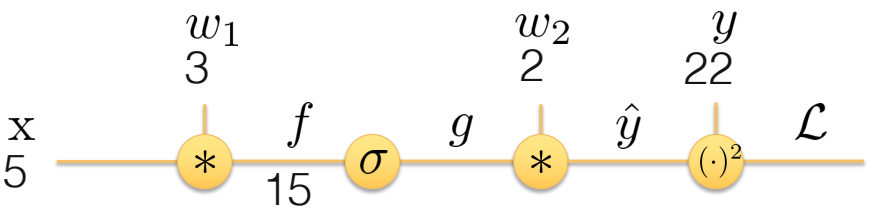
Autodiff Example

Let's plug in the values now…



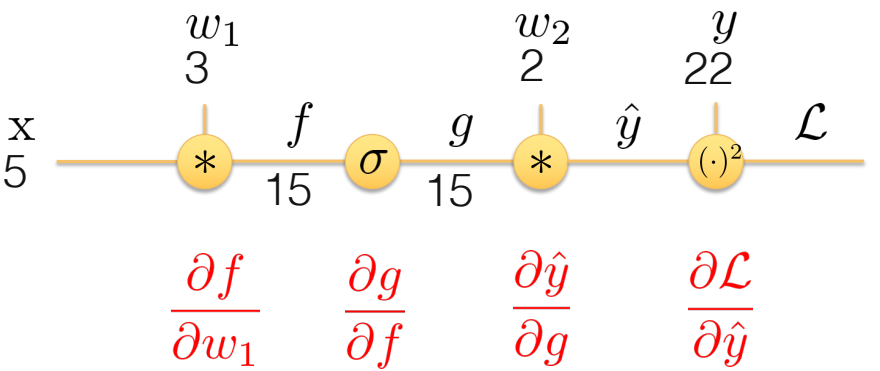
Autodiff Example

Let's plug in the values now…



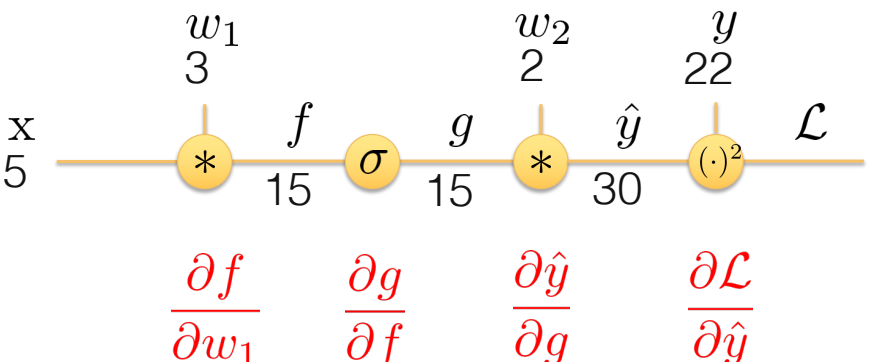
Autodiff Example

Let's plug in the values now…



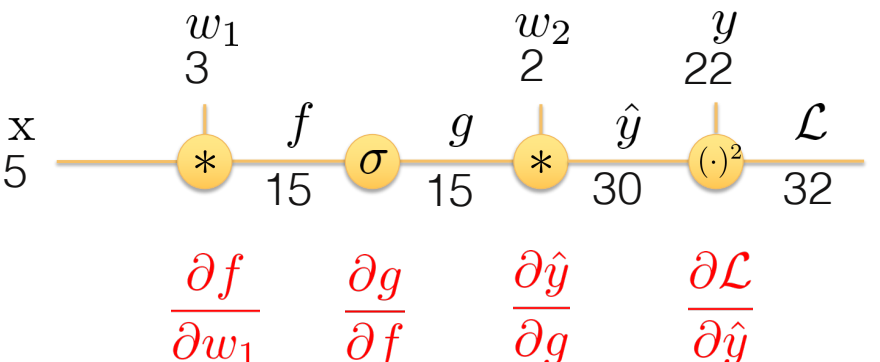
Autodiff Example

Let's plug in the values now…



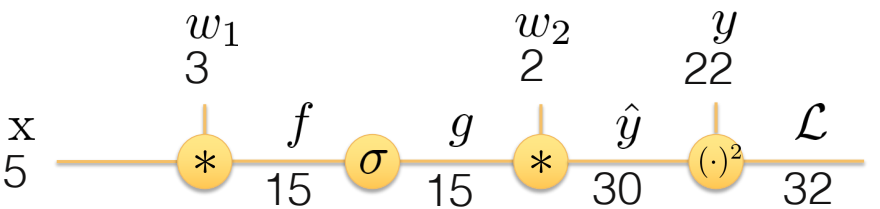
Autodiff Example

Let's plug in the values now…



Autodiff Example

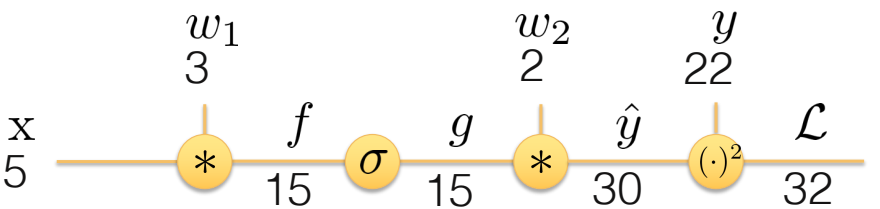
Let's plug in the values now…



$$\frac{\partial f}{\partial w_1} \quad \frac{\partial g}{\partial f} \quad \frac{\partial \hat{y}}{\partial g} \quad \frac{\partial \mathcal{L}}{\partial \hat{y}} = \hat{y} - y$$

Autodiff Example

Let's plug in the values now…

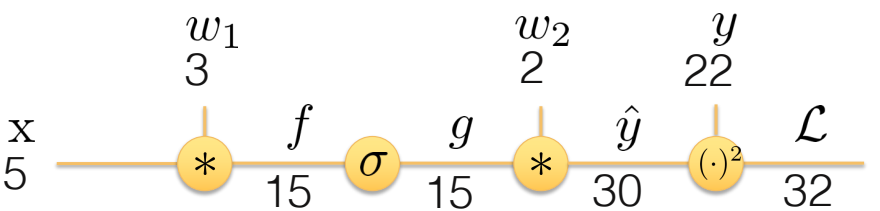


$$\frac{\partial f}{\partial w_1} \quad \frac{\partial g}{\partial f} \quad \frac{\partial \hat{y}}{\partial g} \quad \frac{\partial \mathcal{L}}{\partial \hat{y}} = \hat{y} - y$$

8

Autodiff Example

Let's plug in the values now…

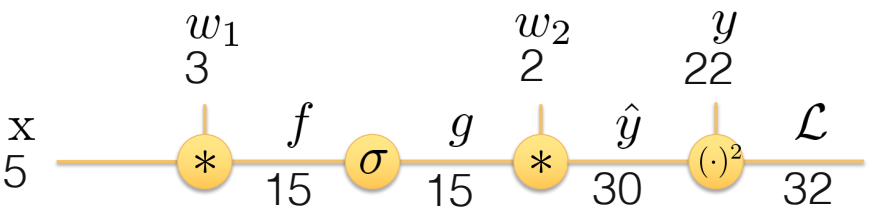


$$\frac{\partial f}{\partial w_1} \qquad \frac{\partial g}{\partial f} \qquad \frac{\partial \hat{y}}{\partial g} \qquad \frac{\partial \mathcal{L}}{\partial \hat{y}} = \hat{y} - y$$

2 8

Autodiff Example

Let's plug in the values now…

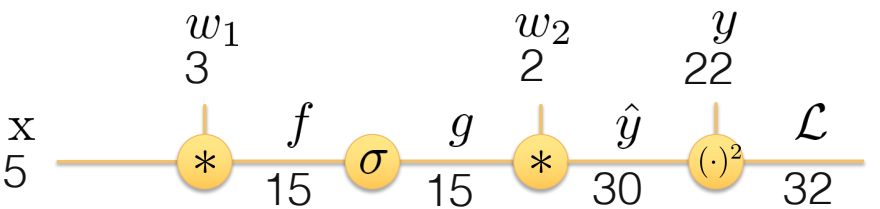


$$\frac{\partial f}{\partial w_1} \qquad \frac{\partial g}{\partial f} \qquad \frac{\partial \hat{y}}{\partial g} \qquad \frac{\partial \mathcal{L}}{\partial \hat{y}} = \hat{y} - y$$

1 2 8

Autodiff Example

Let's plug in the values now…

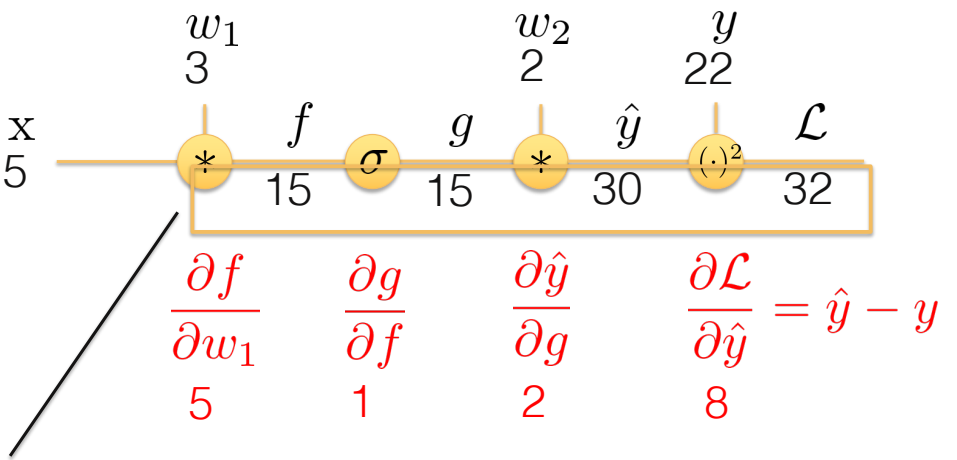


$$\frac{\partial f}{\partial w_1} \quad \frac{\partial g}{\partial f} \quad \frac{\partial \hat{y}}{\partial g} \quad \frac{\partial \mathcal{L}}{\partial \hat{y}} = \hat{y} - y$$

5 1 2 8

Autodiff Example

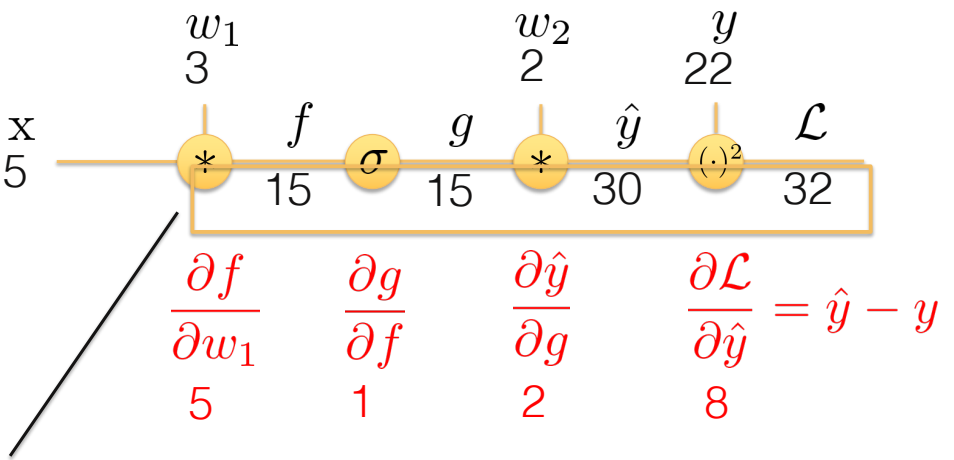
What is backpropagation?



Save these intermediate values during forward computation

Autodiff Example

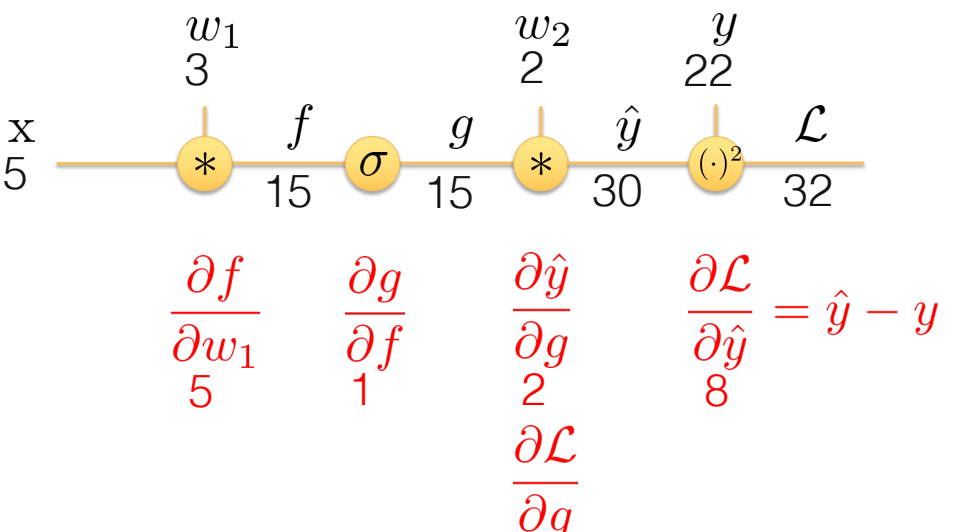
What is backpropagation?



Then we perform a “backward pass”

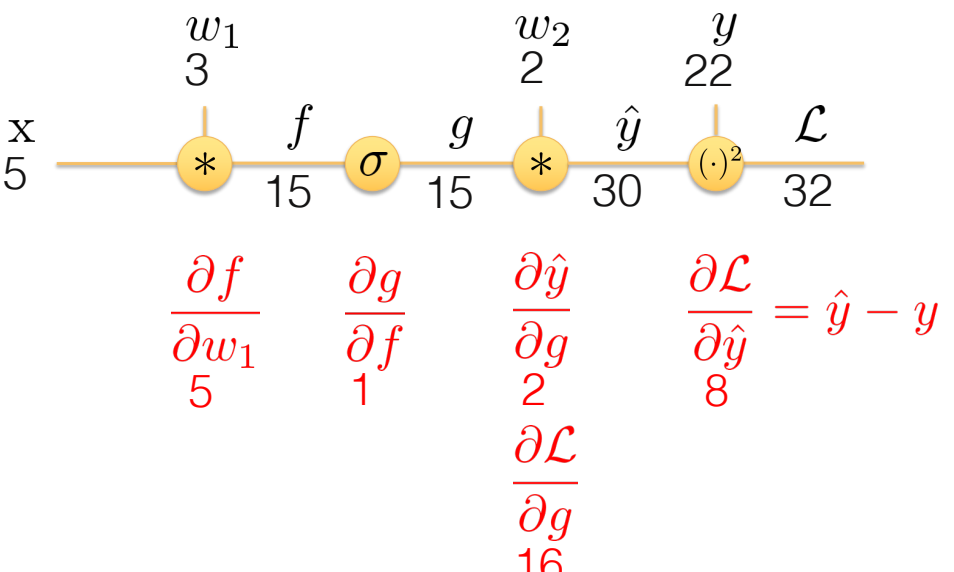
Autodiff Example

What is backpropagation?



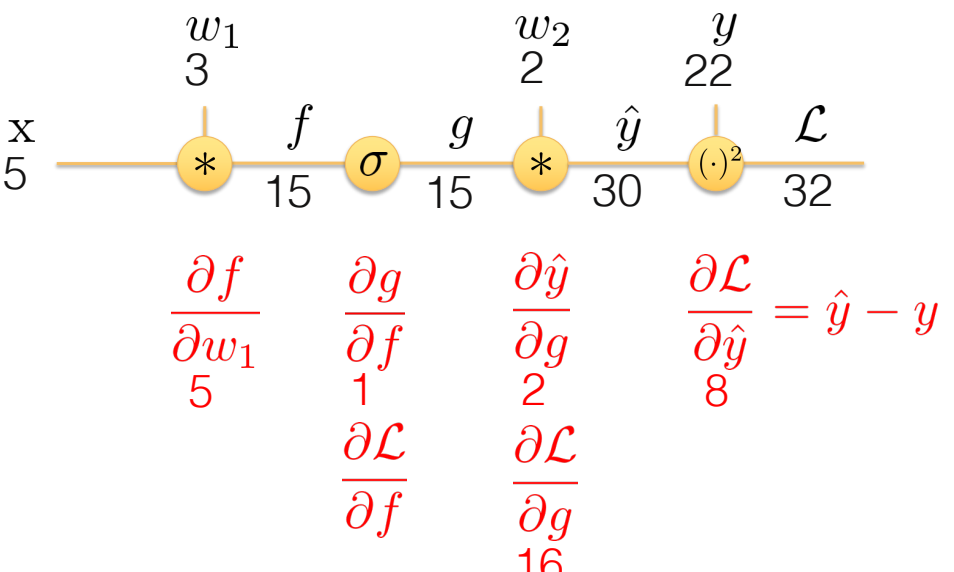
Autodiff Example

What is backpropagation?



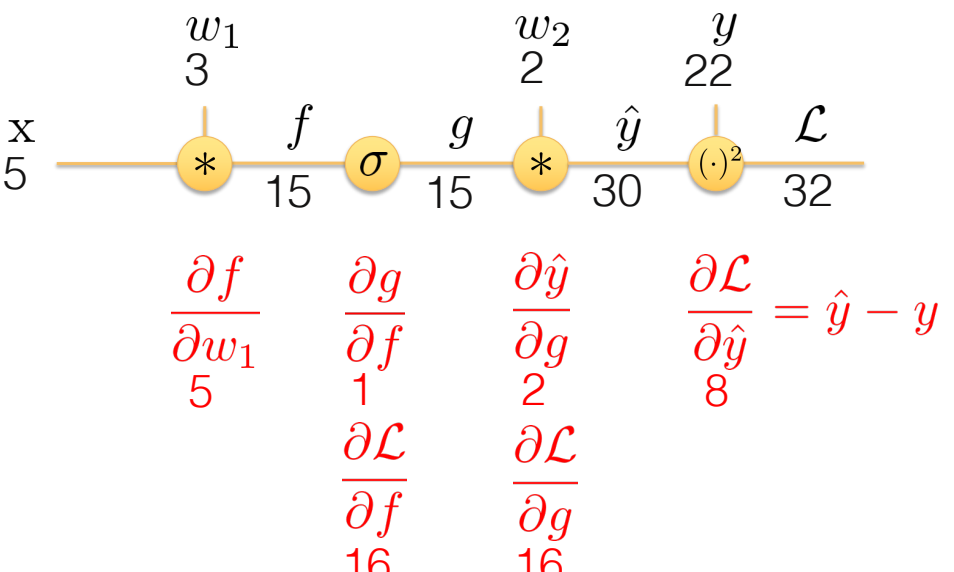
Autodiff Example

What is backpropagation?



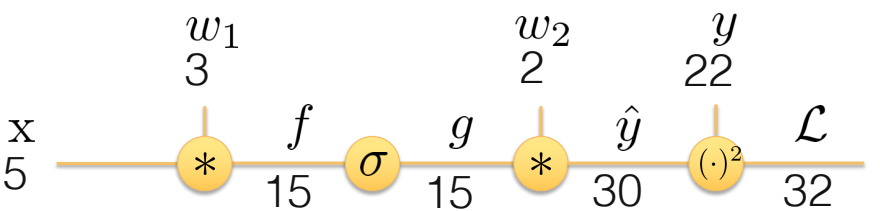
Autodiff Example

What is backpropagation?



Autodiff Example

What is backpropagation?



$$\frac{\partial f}{\partial w_1} \quad \frac{\partial g}{\partial f} \quad \frac{\partial \hat{y}}{\partial g} \quad \frac{\partial \mathcal{L}}{\partial \hat{y}} = \hat{y} - y$$

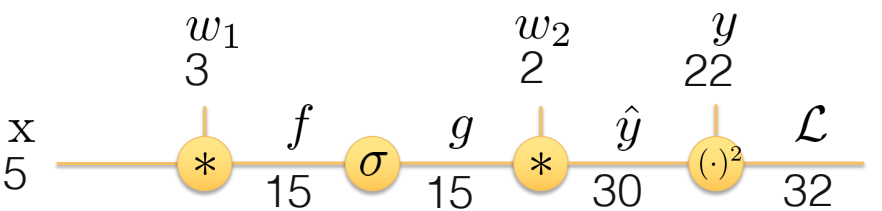
5 1 2 8

$$\frac{\partial \mathcal{L}}{\partial w_1} \quad \frac{\partial \mathcal{L}}{\partial f} \quad \frac{\partial \mathcal{L}}{\partial g}$$

16 16

Autodiff Example

What is backpropagation?



$$\frac{\partial f}{\partial w_1} \quad \frac{\partial g}{\partial f} \quad \frac{\partial \hat{y}}{\partial g} \quad \frac{\partial \mathcal{L}}{\partial \hat{y}} = \hat{y} - y$$

5 1 2 8

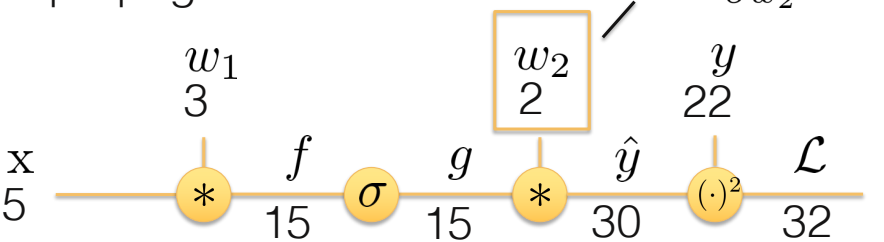
$$\frac{\partial \mathcal{L}}{\partial w_1} \quad \frac{\partial \mathcal{L}}{\partial f} \quad \frac{\partial \mathcal{L}}{\partial g}$$

80 16 16

Autodiff Example

What is backpropagation?

What about $\frac{\partial \mathcal{L}}{\partial w_2}$?



$$\frac{\partial f}{\partial w_1} \quad \frac{\partial g}{\partial f} \quad \frac{\partial \hat{y}}{\partial g} \quad \frac{\partial \mathcal{L}}{\partial \hat{y}} = \hat{y} - y$$

5 1 2 8

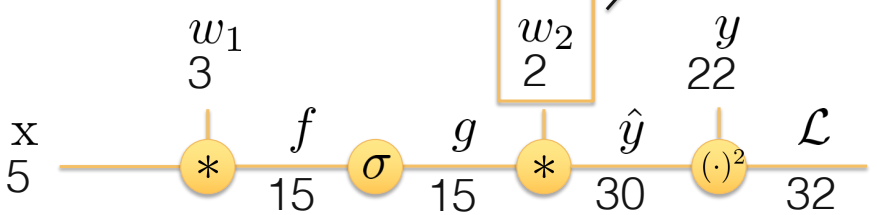
$$\frac{\partial \mathcal{L}}{\partial w_1} \quad \frac{\partial \mathcal{L}}{\partial f} \quad \frac{\partial \mathcal{L}}{\partial g}$$

80 16 16

Autodiff Example

What is backpropagation?

What about $\frac{\partial \mathcal{L}}{\partial w_2}$?



$$\frac{\partial f}{\partial w_1} \quad \frac{\partial g}{\partial f} \quad \frac{\partial \hat{y}}{\partial g} \quad \frac{\partial \mathcal{L}}{\partial \hat{y}} = \hat{y} - y$$

5 1 2

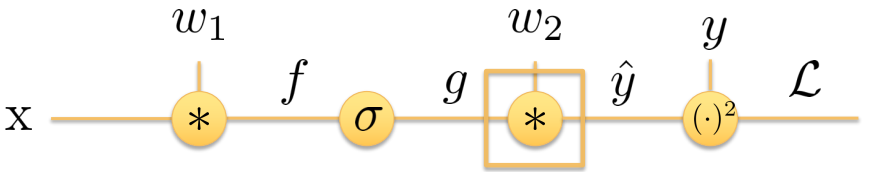
$$\frac{\partial \mathcal{L}}{\partial w_1} \quad \frac{\partial \mathcal{L}}{\partial f} \quad \frac{\partial \mathcal{L}}{\partial g}$$

80 16 16

We can re-use computation!

$$\frac{\partial \mathcal{L}}{\partial w_2} = \frac{\partial \hat{y}}{\partial w_2} \frac{\partial \mathcal{L}}{\partial \hat{y}}$$

Autodiff Example



PyTorch Code:

```
class Multiply(torch.autograd.Function):
  @staticmethod
  def forward(ctx, x, y):
    ctx.save_for_backward(x, y)
    z = x * y
    return z
  @staticmethod
  def backward(ctx, grad_z):
    x, y = ctx.saved_tensors
    grad_x = y * grad_z   # dz/dx * dL/dz
    grad_y = x * grad_z   # dz/dy * dL/dz
    return grad_x, grad_y
```

Need to stash some values for use in backward

Upstream gradient

Multiply upstream and local gradients

Neural field training loop

1. Sample batch of pixel coordinates and pixel values

Neural field training loop

1. Sample batch of pixel coordinates and pixel values

2. ?

Neural field training loop

1. Sample batch of pixel coordinates and pixel values

2. Run forward pass to calculate pixel output for each coordinate

Neural field training loop

1. Sample batch of pixel coordinates and pixel values

2. Run forward pass to calculate pixel output for each coordinate

3. ?

Neural field training loop

1. Sample batch of pixel coordinates and pixel values

2. Run forward pass to calculate pixel output for each coordinate

3. Run backward pass to calculate gradients with backpropagation

Neural field training loop

1. Sample batch of pixel coordinates and pixel values

2. Run forward pass to calculate pixel output for each coordinate

3. Run backward pass to calculate gradients with backpropagation

4. ?

Neural field training loop

1. Sample batch of pixel coordinates and pixel values

2. Run forward pass to calculate pixel output for each coordinate

3. Run backward pass to calculate gradients with backpropagation

4. Update parameters with stochastic gradient descent

4. Update parameters with stochastic gradient descent

$$\mathcal{L}_\theta = \|y - \hat{y}\|_2^2$$

$$W_2^{(k+1)} = W_2^{(k)} - \alpha \frac{\partial \mathcal{L}}{\partial W_2}$$

$$W_1^{(k+1)} = W_1^{(k)} - \alpha \frac{\partial \mathcal{L}}{\partial W_1}$$

4. Update parameters with stochastic gradient descent

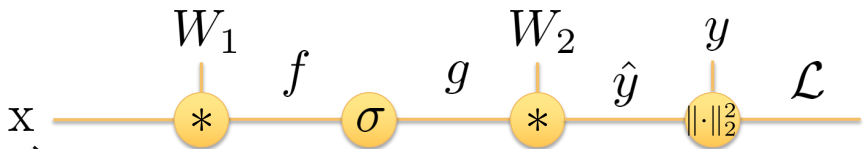
$$\mathcal{L}_\theta = \|y - \hat{y}\|_2^2$$

$$W_2^{(k+1)} = W_2^{(k)} - \alpha \frac{\partial \mathcal{L}}{\partial W_2}$$

$$W_1^{(k+1)} = W_1^{(k)} - \alpha \frac{\partial \mathcal{L}}{\partial W_1}$$

why is this a negative sign and not a positive sign?

Vector Differentiation



But wait, aren't these vectors?
(see supplemental slides for vector differentiation)

Takeaways

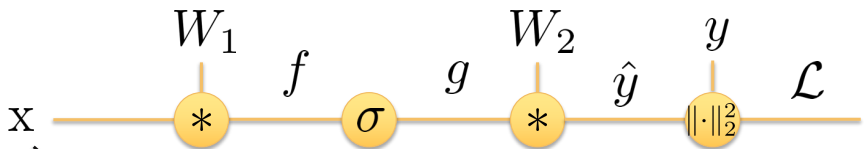
- What is a neural network? (can you write the equation for an MLP?)
- What is a neural field and what is positional encoding?
- Backpropagation, automatic differentiation, and gradient descent

Next Time

- Recognition!
  - CNNs
    - classification
    - object detection
    - segmentation

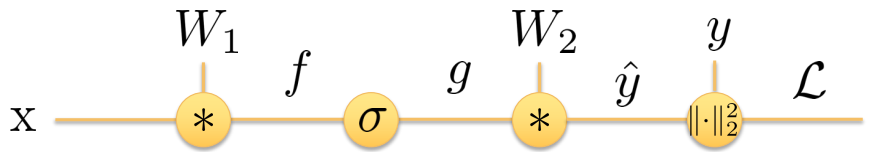
Supplemental Slides

Vector Differentiation



But wait, aren't these vectors?
(see supplemental slides for vector differentiation)

Vector Differentiation



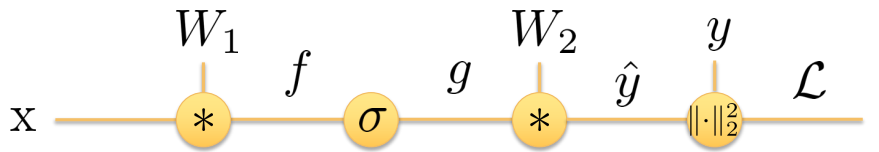
Recap: vector differentiation

Scalar by Scalar

$$x, y \in \mathbb{R}$$

$$\frac{\partial y}{\partial x} \in \mathbb{R}$$

Vector Differentiation



Recap: vector differentiation

Scalar by Scalar

$$x, y \in \mathbb{R}$$

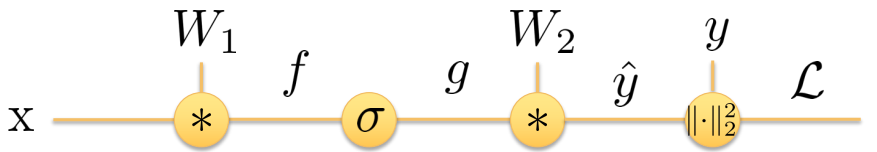
$$\frac{\partial y}{\partial x} \in \mathbb{R}$$

Scalar by Vector

$$x \in \mathbb{R}^N, y \in \mathbb{R}$$

$$\frac{\partial y}{\partial x} \in \mathbb{R}^N$$

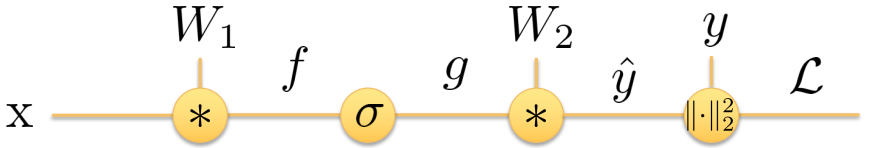
Vector Differentiation



Recap: vector differentiation

Scalar by Scalar	Scalar by Vector	Vector by Vector
$x, y \in \mathbb{R}$	$x \in \mathbb{R}^N, y \in \mathbb{R}$	$x \in \mathbb{R}^N, y \in \mathbb{R}^M$
$\frac{\partial y}{\partial x} \in \mathbb{R}$	$\frac{\partial y}{\partial x} \in \mathbb{R}^N$	$\frac{\partial y}{\partial x} \in \mathbb{R}^{N \times M}$

Recap: vector differentiation



Example 1: matrix multiply

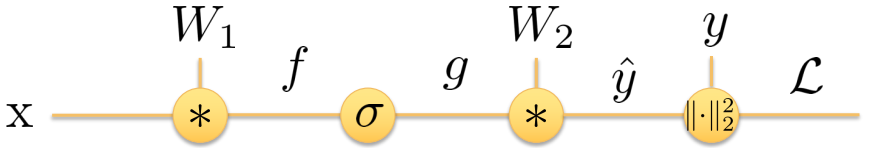
$$\frac{\partial \hat{y}}{\partial g} = \frac{\partial}{\partial g} W_2 g$$

$$W_2 \in \mathbb{R}^{M \times N}$$

$$g \in \mathbb{R}^{N}$$

$$\frac{\partial \hat{y}}{\partial g} \in \mathbb{R}^{N \times M}$$

Recap: vector differentiation



Example 1: matrix multiply

$$\frac{\partial \hat{y}}{\partial g} = \frac{\partial}{\partial g} W_2 g$$

$$W_2 \in \mathbb{R}^{M \times N}$$

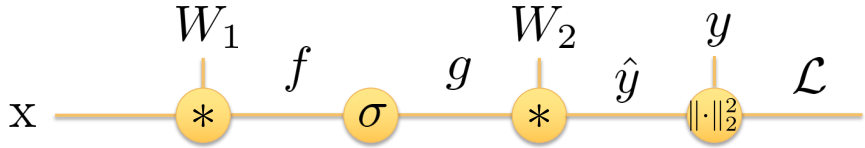
$$g \in \mathbb{R}^{N}$$

$$\frac{\partial \hat{y}}{\partial g} \in \mathbb{R}^{N \times M}$$

$$\frac{\partial}{\partial g} W_2 g$$

$$= \frac{\partial}{\partial g} \begin{bmatrix} w_{11} g_1 + \cdots + w_{1n} g_n \\ \vdots \quad \ddots \quad \vdots \\ w_{m1} g_1 + \cdots + w_{mn} g_n \end{bmatrix}$$

Recap: vector differentiation



$$\frac{\partial \hat{y}}{\partial g} = \begin{bmatrix} \frac{\partial \hat{y}_1}{\partial g_1} & \cdots & \frac{\partial \hat{y}_m}{\partial g_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial \hat{y}_1}{\partial g_n} & \cdots & \frac{\partial \hat{y}_m}{\partial g_n} \end{bmatrix}$$

Example 1: matrix multiply

$$\frac{\partial \hat{y}}{\partial g} = \frac{\partial}{\partial g} W_2 g$$

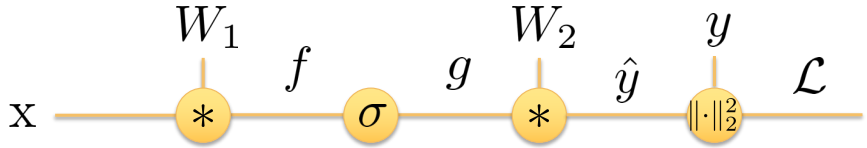
$$W_2 \in \mathbb{R}^{M \times N}$$

$$g \in \mathbb{R}^{N}$$

$$\frac{\partial \hat{y}}{\partial g} \in \mathbb{R}^{N \times M}$$

$$\frac{\partial}{\partial g} W_2 g = \frac{\partial}{\partial g} \begin{bmatrix} w_{11} g_1 + \cdots + w_{1n} g_n \\ \vdots \quad \ddots \quad \vdots \\ w_{m1} g_1 + \cdots + w_{mn} g_n \end{bmatrix} = \begin{bmatrix} w_{11} & \cdots & w_{m1} \\ \vdots & \ddots & \vdots \\ w_{1n} & \cdots & w_{mn} \end{bmatrix}$$

Recap: vector differentiation



$$\frac{\partial \hat{y}}{\partial g} = \begin{bmatrix} \frac{\partial \hat{y}_1}{\partial g_1} & \cdots & \frac{\partial \hat{y}_n}{\partial g_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial \hat{y}_1}{\partial g_n} & \cdots & \frac{\partial \hat{y}_n}{\partial g_n} \end{bmatrix}$$

Example 1: matrix multiply

$$\frac{\partial \hat{y}}{\partial g} = \frac{\partial}{\partial g} W_2 g$$

$$W_2 \in \mathbb{R}^{M \times N}$$

$$g \in \mathbb{R}^{N}$$

$$\frac{\partial \hat{y}}{\partial g} \in \mathbb{R}^{N \times M}$$

$$\frac{\partial}{\partial g} W_2 g$$

$$= \frac{\partial}{\partial g} \begin{bmatrix} w_{11} g_1 + \cdots + w_{1n} g_n \\ \vdots \quad \ddots \quad \vdots \\ w_{n1} g_1 + \cdots + w_{nn} g_n \end{bmatrix} = \begin{bmatrix} w_{11} & \cdots & w_{n1} \\ \vdots & \ddots & \vdots \\ w_{1n} & \cdots & w_{nn} \end{bmatrix}$$

$$= W_2^T$$

Recap: vector differentiation

Example 2: elementwise functions

$$h = f \odot g$$

$$f \in \mathbb{R}^N$$

$$g \in \mathbb{R}^N$$

$$\frac{\partial h}{\partial f} \in \mathbb{R}^{N \times N}$$

Recap: vector differentiation

Example 2: elementwise functions

$$h = f \odot g$$

$$f \in \mathbb{R}^N$$

$$g \in \mathbb{R}^N$$

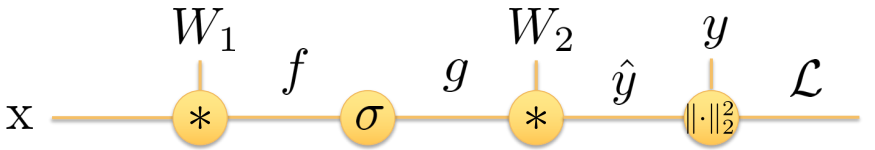
$$\frac{\partial h}{\partial f} \in \mathbb{R}^{N \times N}$$

$$\frac{\partial h}{\partial f} = \begin{bmatrix} g_1 & & 0 \\ & \ddots & \\ 0 & & g_n \end{bmatrix} = \text{diag}(g)$$

$$\frac{\partial h}{\partial f} = \begin{bmatrix} \frac{\partial h_1}{\partial f_1} & \cdots & \frac{\partial h_n}{\partial f_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial h_1}{\partial f_n} & \cdots & \frac{\partial h_n}{\partial f_n} \end{bmatrix}$$

Recap: vector differentiation

Final hint: dimensions should always match up!



$$\frac{\partial \mathcal{L}}{\partial W_1} = \frac{\partial f}{\partial W_1} \frac{\partial g}{\partial f} \frac{\partial \hat{y}}{\partial g} \frac{\partial \mathcal{L}}{\partial \hat{y}}$$

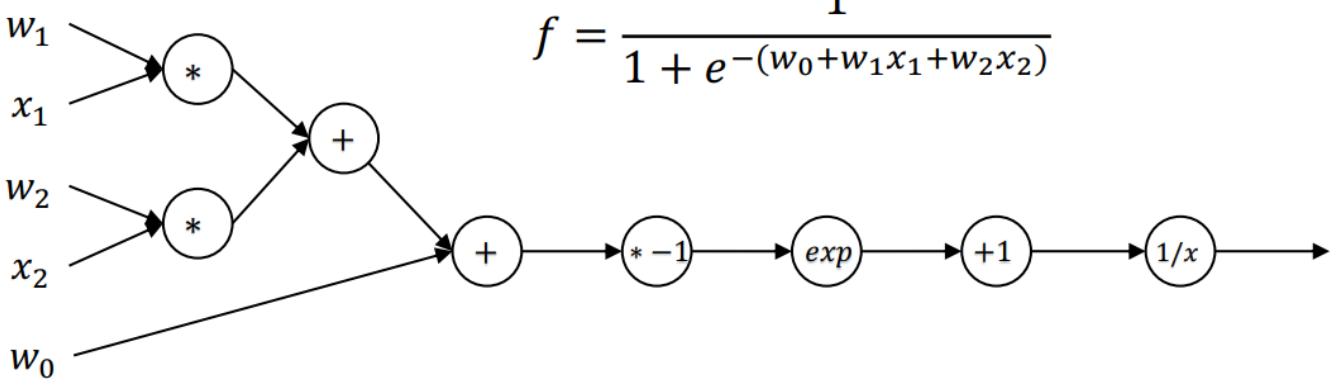
You should be able to calculate derivatives of each of these terms and then perform matrix multiplications without issues

Extra backpropagation example (adapted from Stanford CS231n)

$$f = \frac{1}{1 + e^{-(w_0 + w_1 x_1 + w_2 x_2)}}$$

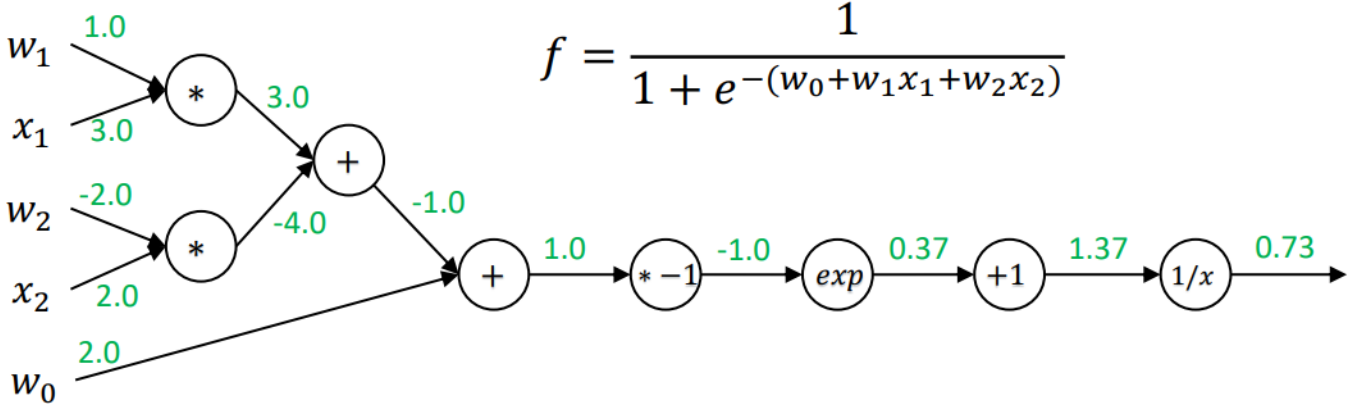
Extra backpropagation example

$$f = \frac{1}{1 + e^{-(w_0 + w_1 x_1 + w_2 x_2)}}$$

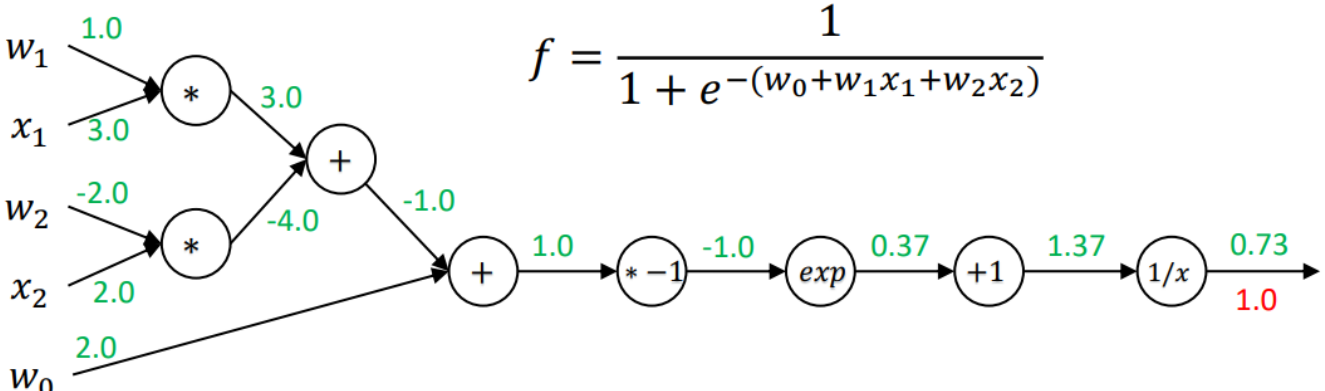


Extra backpropagation example

$$f = \frac{1}{1 + e^{-(w_0 + w_1 x_1 + w_2 x_2)}}$$

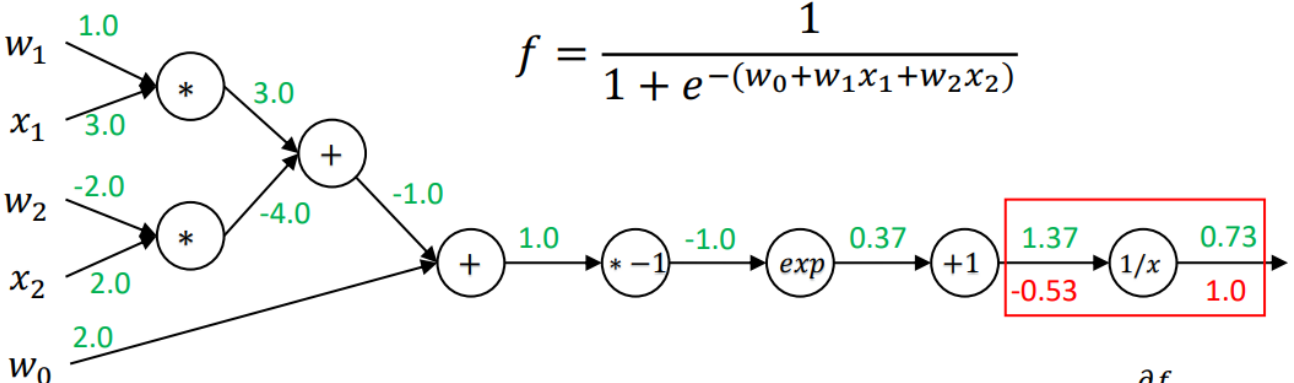


Extra backpropagation example



$$f = \frac{1}{1 + e^{-(w_0 + w_1 x_1 + w_2 x_2)}}$$

Extra backpropagation example

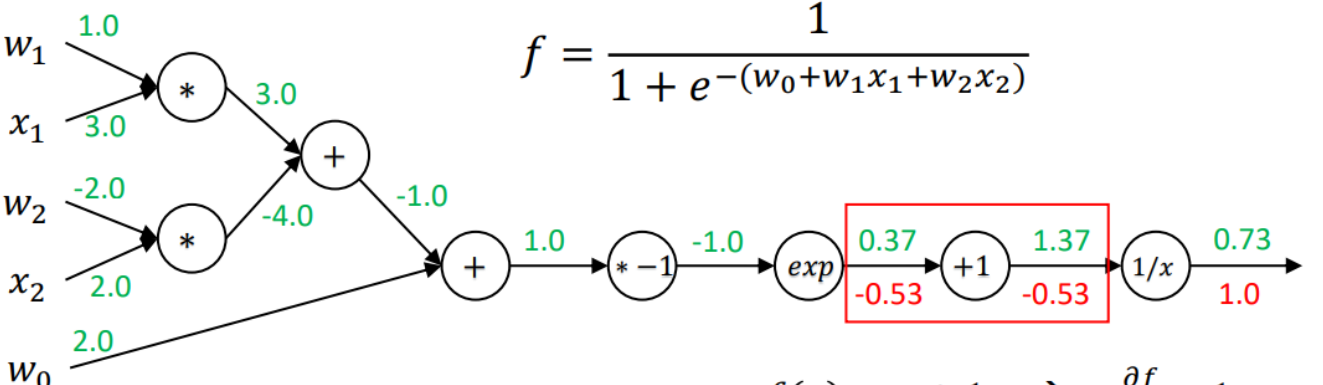


$$f = \frac{1}{1 + e^{-(w_0 + w_1 x_1 + w_2 x_2)}}$$

$$f(x) = 1/x \quad \rightarrow \quad \frac{\partial f}{\partial x} = -1/x^2$$

$$\frac{\partial J}{\partial x} = \frac{\partial J}{\partial f}\frac{\partial f}{\partial x} = -1/x^2$$

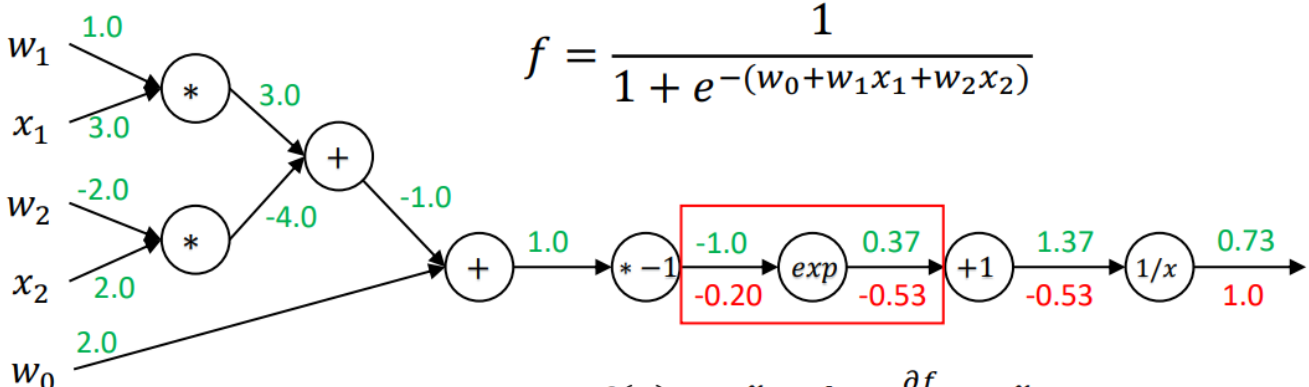
Extra backpropagation example



$$f = \frac{1}{1 + e^{-(w_0 + w_1 x_1 + w_2 x_2)}}$$

$$f(x) = x + 1 \quad \rightarrow \quad \frac{\partial f}{\partial x} = 1$$

Extra backpropagation example

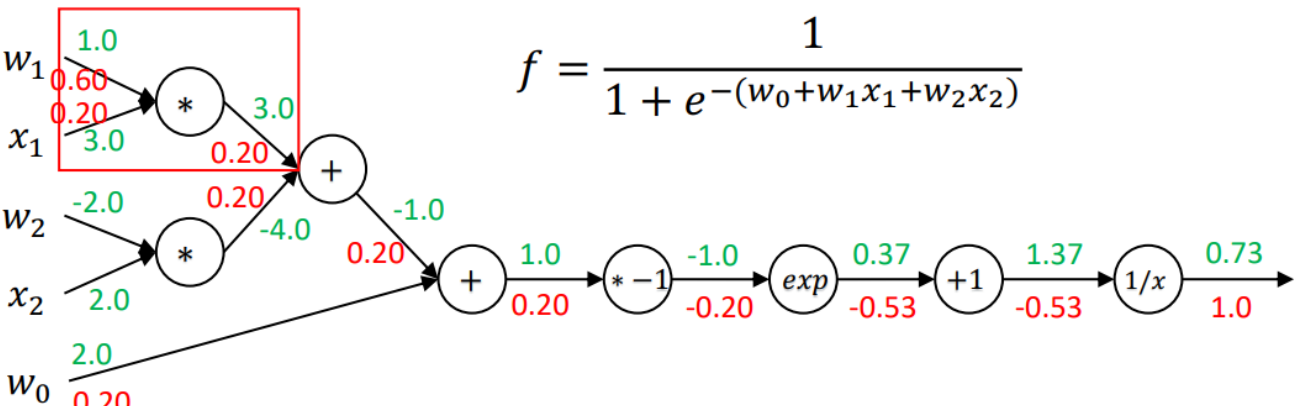


$$f = \frac{1}{1 + e^{-(w_0 + w_1 x_1 + w_2 x_2)}}$$

$$f(x) = e^x \quad \rightarrow \quad \frac{\partial f}{\partial x} = e^x$$

$$\frac{\partial J}{\partial x} = \frac{\partial J}{\partial f}\frac{\partial f}{\partial x} = \frac{\partial J}{\partial f} \cdot e^x$$

Extra backpropagation example



$$f = \frac{1}{1 + e^{-(w_0 + w_1 x_1 + w_2 x_2)}}$$

$$f(x, w) = xw \quad \rightarrow \quad \frac{\partial f}{\partial x} = w, \ \frac{\partial f}{\partial w} = x$$

Extra backpropagation example

$$f = \frac{1}{1 + e^{-(w_0 + w_1 x_1 + w_2 x_2)}}$$