SUSTech
Southern University of Science and Technology

神经计算与控制实验室
Neural Computing & Control Lab

# Brain Intelligence and Artificial Intelligence

# 人脑智能与机器智能

## Lecture 5 – A brief intro to AI & Hands-on

**Quanying Liu （刘泉影）**

SUSTech, BME department

Email: liuqy@sustech.edu.cn

# Lecture 5 – Intro to AI & hands on

- **A general introduction to AI**
  - 3 key components
  - The network **Architecture**
- **AI learning: Gradient Descent (GD)** to minimize a *loss function*
  - What is Gradient Descent?
  - Gradient Descent to train deep NNs → Error Backpropagation
- **Hands-on** (pytorch), thanks to three TAs
- **Error Back-propagation (BP)** in fully-connected NN
  - Backpropagation – forward pass
  - Backpropagation – backward pass
  - BP in the brain

# 3 Key Components

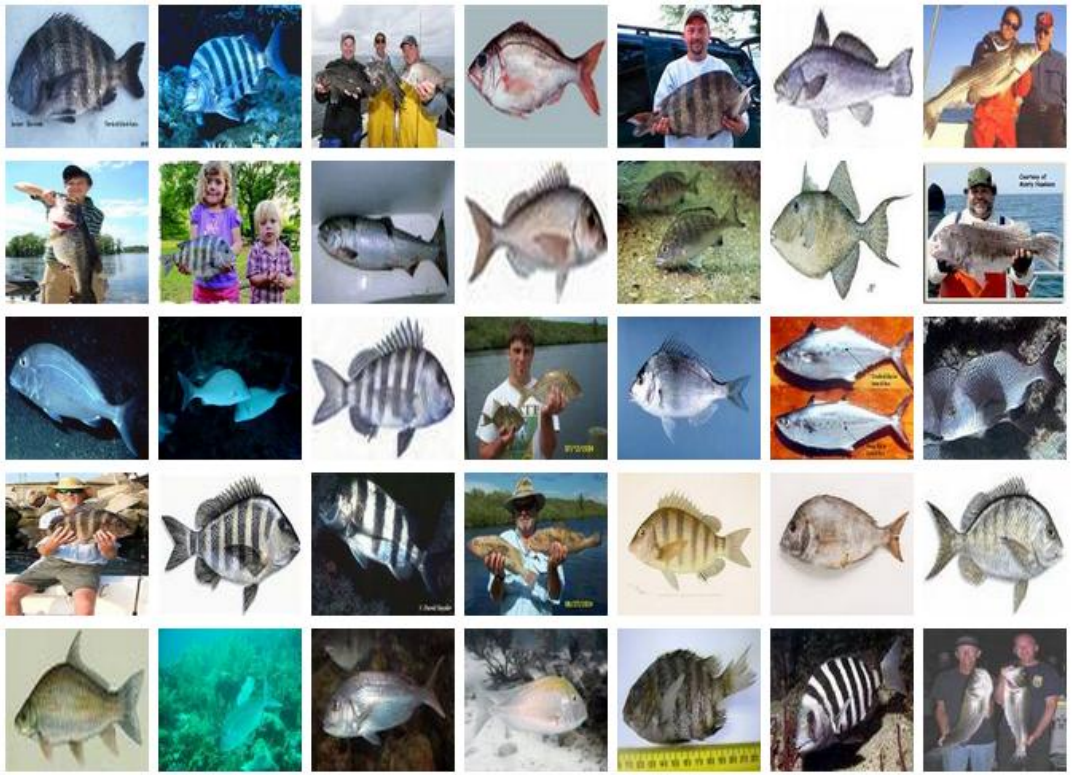
1. **Data**
   - Without labels: a data point (x), a dataset (X)
   - With labels: a pair of data (x, y), a dataset (X, Y)

2. **Model**
   - Network: the high-level concept, a function (input, output)
   - Loss function (objective function) : the goal of machine learning
   - Parameters: the variables to be learned/estimated

3. **Optimization algorithm**
   - The algorithm to minimize the loss function (or maximize the objective function)

# Benchmark datasets in computer vision (机器视觉)


- 14 million images and 1000 categories.
- Largest database of labeled images.



- Images in Fish category.
- Captures variations of fish.

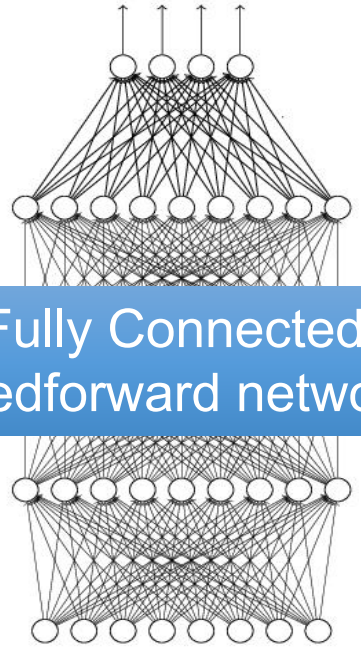
Picture credits: Image-net.org, ZDnet.com

# The network architecture

- MLP (fully-connected)
- CNN (convolution)
- GNN (graph)
- Transformer (attention)
- ...

# The architecture of CNN

cat / dog ……



Fully Connected Feedforward network

Convolution

Activation function

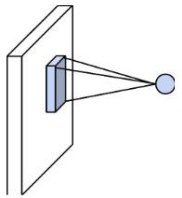
Pooling

… …

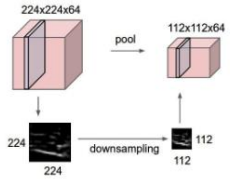
Flatten

A block involves some core *components*.
These blocks can repeat many times.

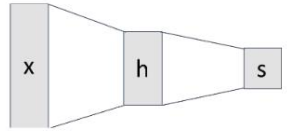
Convolution Layers



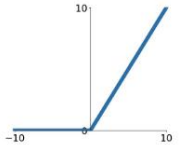
Pooling Layers




Fully-Connected Layers




Activation Function

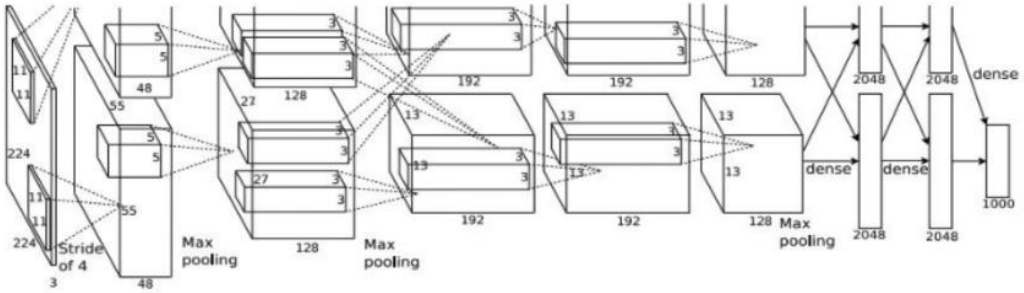



Normalization

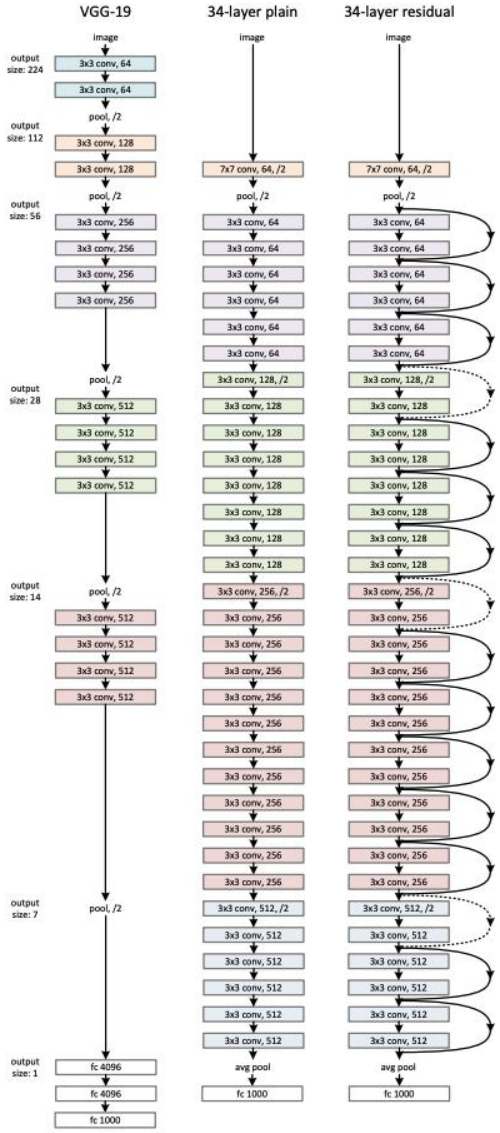
$$\hat{x}_{i,j} = \frac{x_{i,j} - \mu_j}{\sqrt{\sigma_j^2 + \varepsilon}}$$

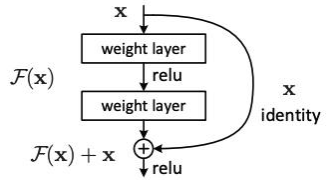
# The architecture of CNN

## AlexNet

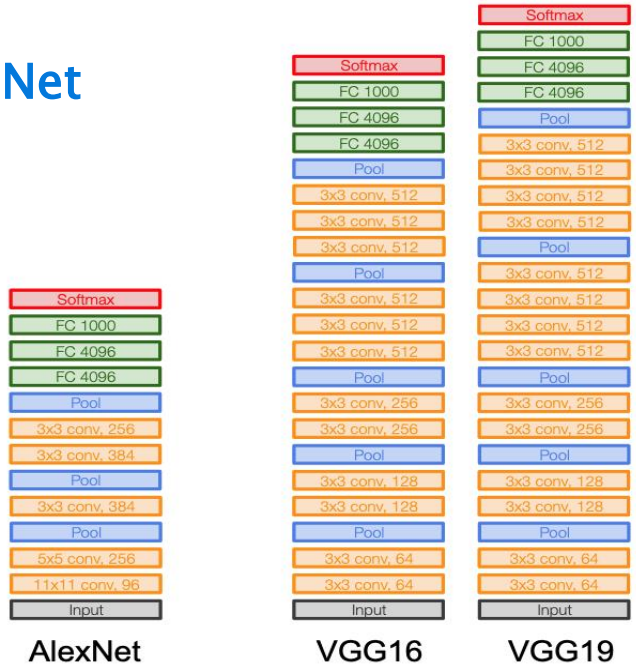



## ResNet

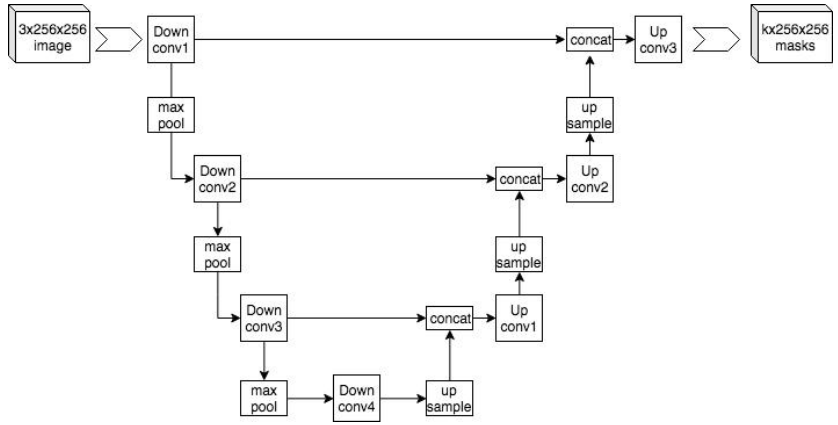


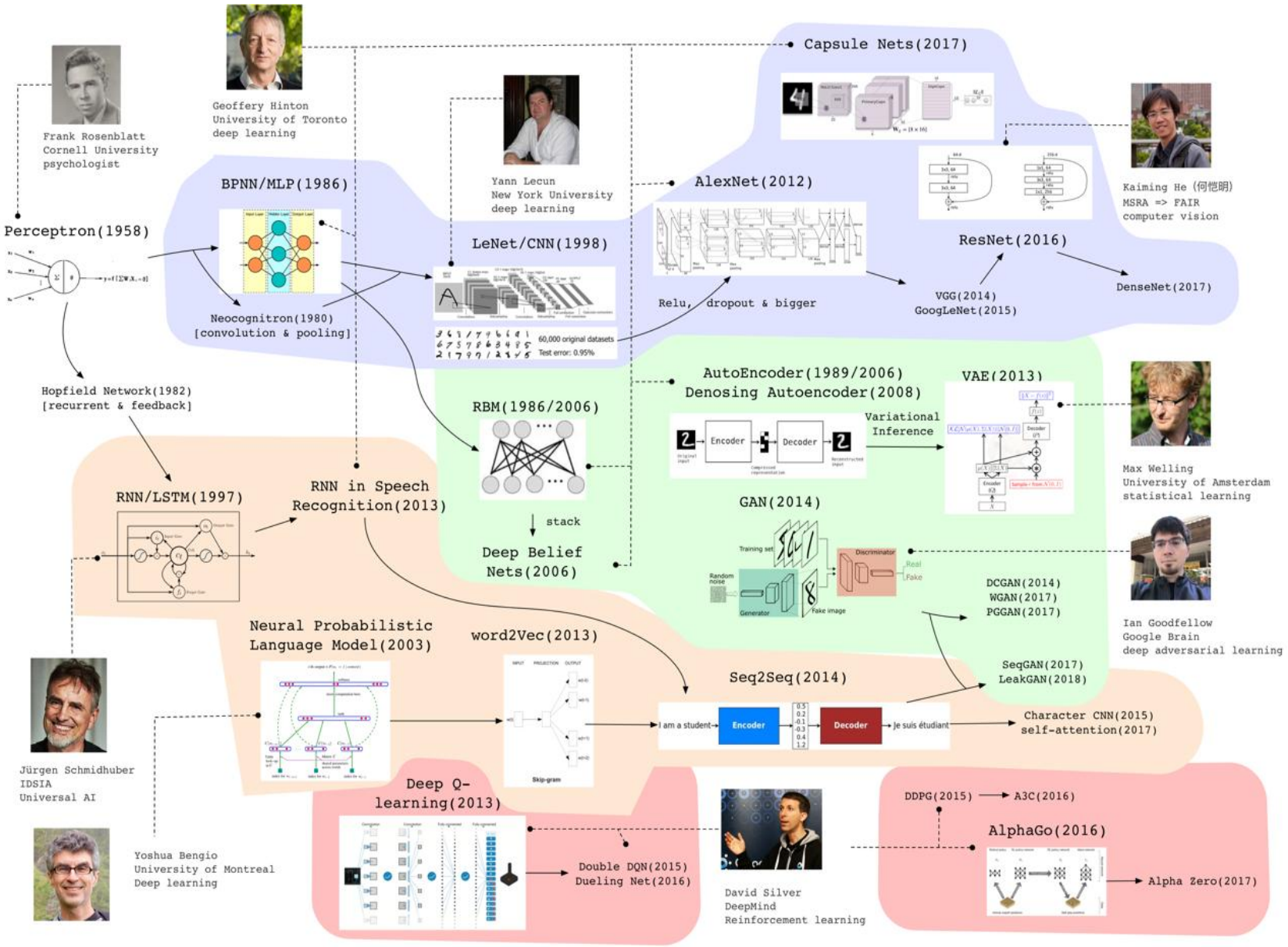

## VGGNet




## U-Net



《2020 中国人工智能白皮书》




## Computer Vision and CNN

1979, ***Neocognitron*** by Fukushima
1986, ***Backpropogation MLP*** by Hinton
1998, ***LeNet-5*** by LeCun
2012, ***AlexNet*** by Hinton
2016, ***ResNet*** by 何恺明

## Generative Models

1986-2006, ***RBM*** by Hinton
1989/2006, ***AutoEncoder*** by Hinton
2014, ***GAN*** by Goodfellow

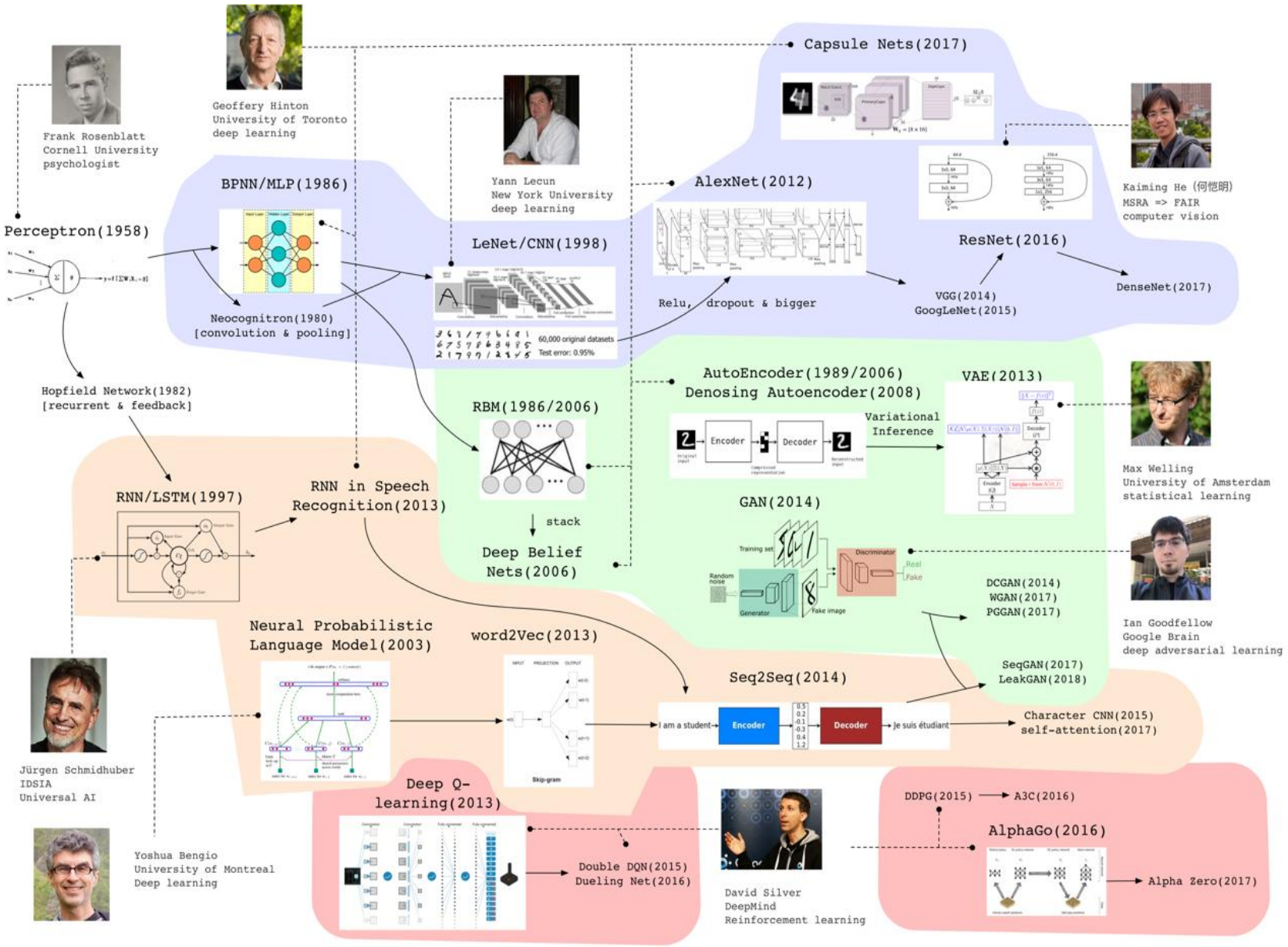
## Sequence Models

1982, ***Hopfield Network*** by Hopfield
1997, ***LSTM*** by Schmidhuber
2013, **RNN** by Hinton
2017, **transformer** by google

## Reinforcement Learning

2013, ***Deep Q-learning*** by Silver
2016, ***AlphaGo*** by deepmind

《2020 中国人工智能白皮书》




## Computer Vision and CNN

1979, *Neocognitron* by Fukushima
1986, *Backpropogation MLP* by Hinton
1998, *LeNet-5* by LeCun
2012, *AlexNet* by Hinton
2016, *ResNet* by 何恺明
2020, **ViT** by google

## Generative Models

1986-2006, *RBM* by Hinton
1989/2006, *AutoEncoder* by Hinton
2014, *GAN* by Goodfellow
2020, **Diffusion** model by Ho & Abbeel

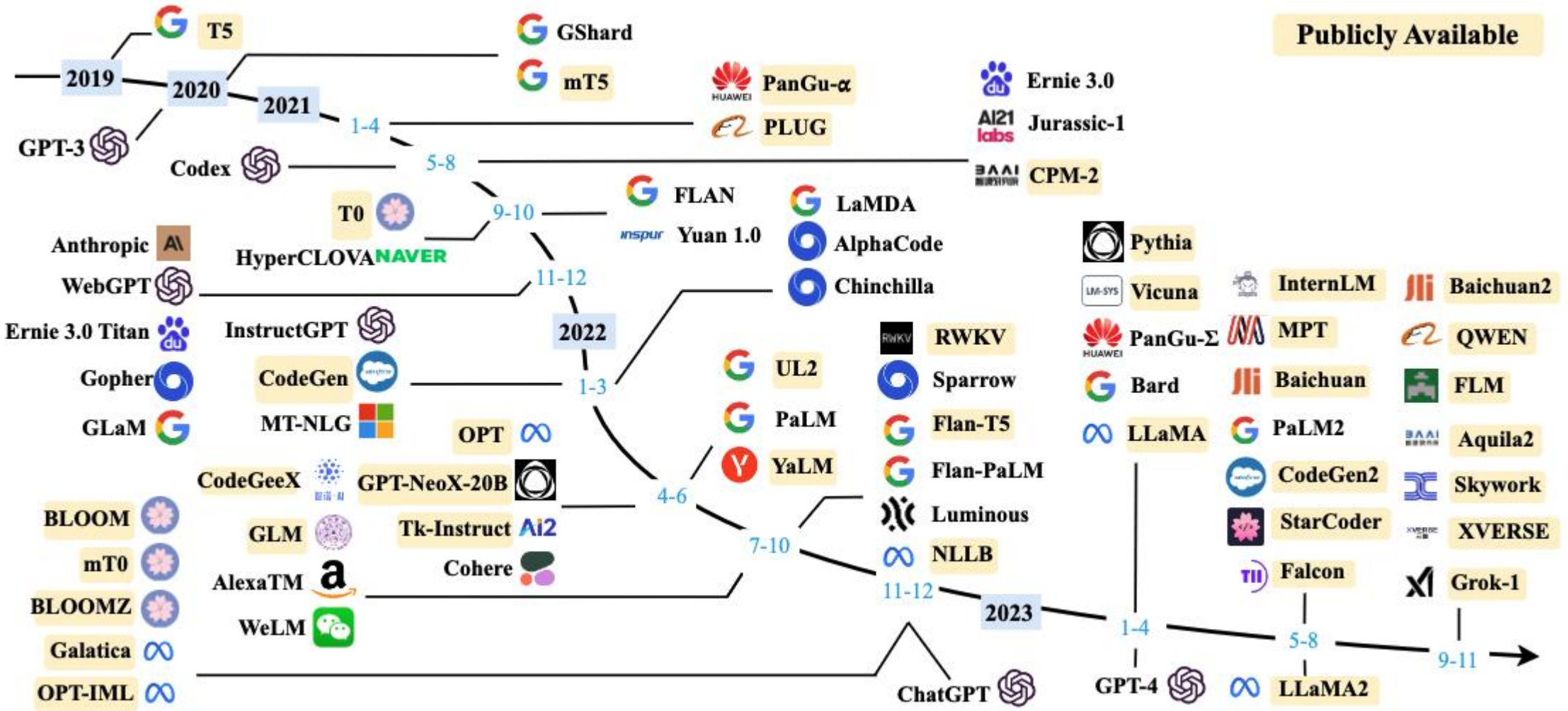
## Sequence Models

1982, *Hopfield Network* by Hopfield
1997, *LSTM* by Schmidhuber
2013, **RNN** by Hinton
2017, **transformer** by google
2022, **chatGPT** by openAI

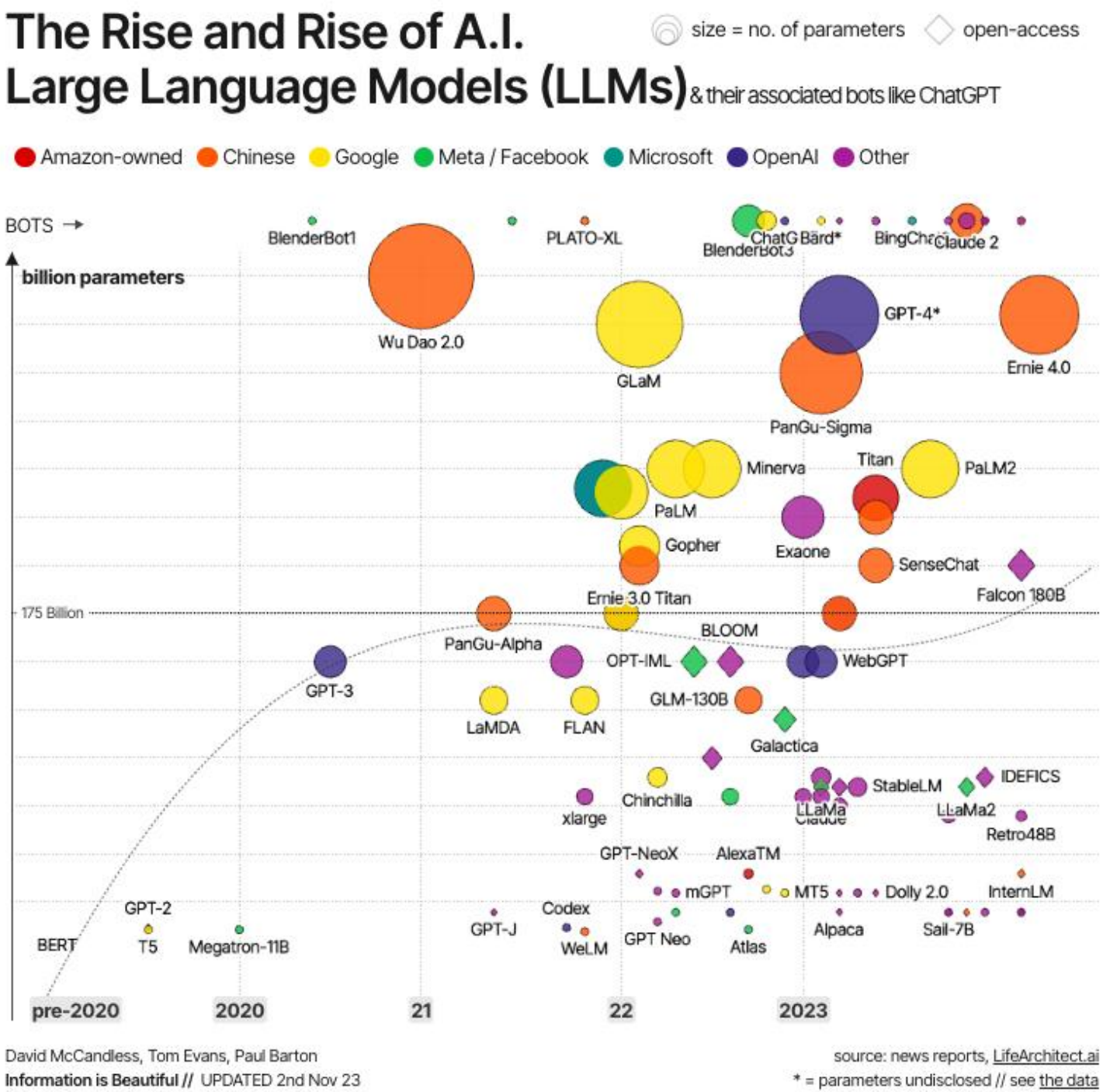
## Reinforcement Learning

2013, *Deep Q-learning* by Silver
2016, *AlphaGo* by deepmind
2020, **AlphaFold** by deepmind

# 大模型时代



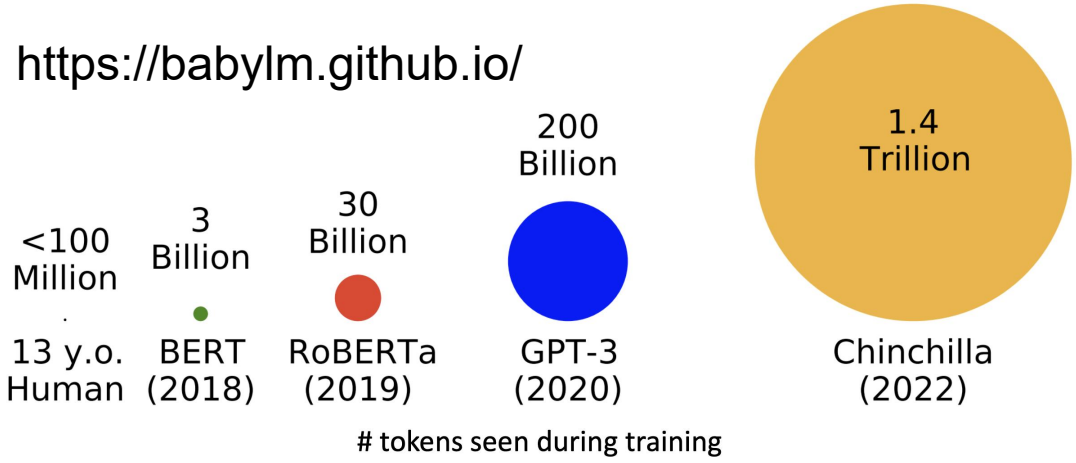

Zhao, et al. (2023). A survey of large language models. arXiv:2303.18223




Pre-LLM era

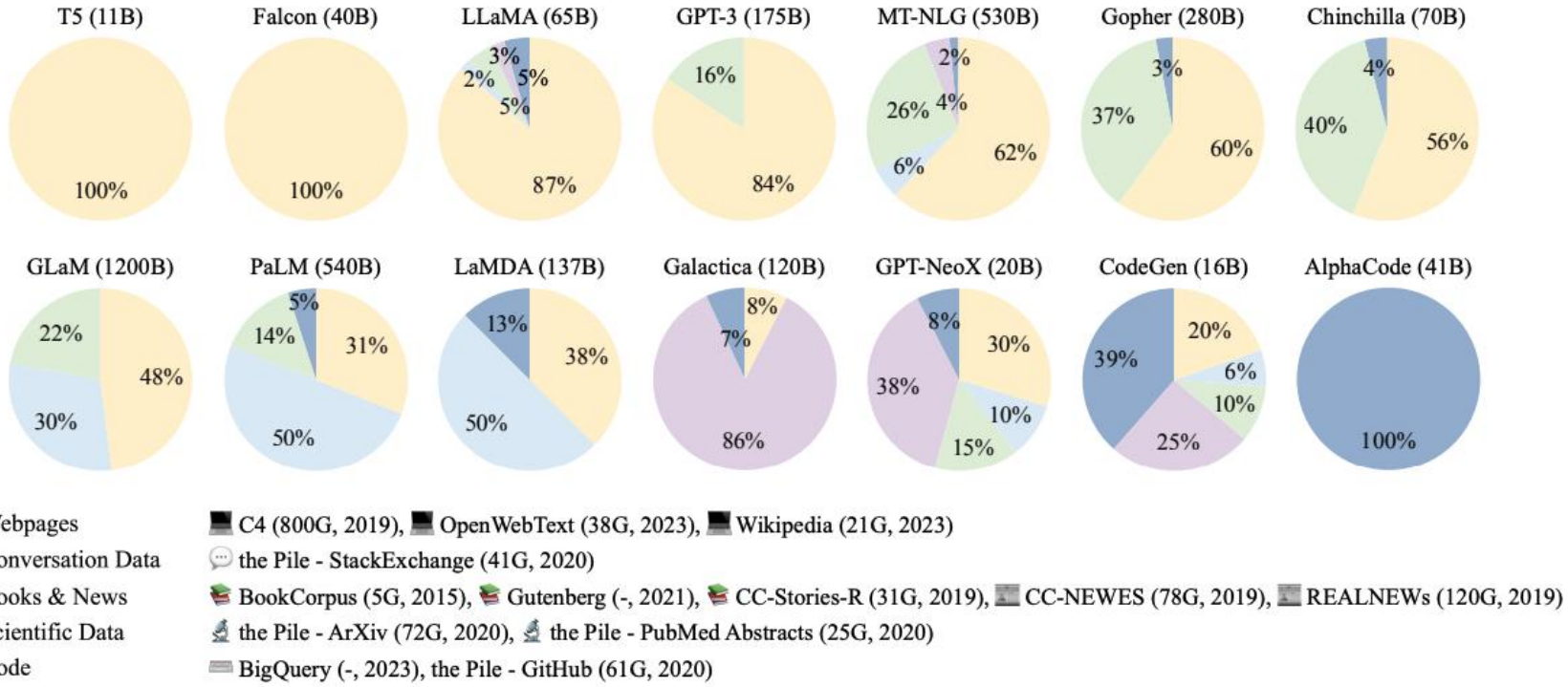
Post-LLM era

# AI模型的训练数据越来越大

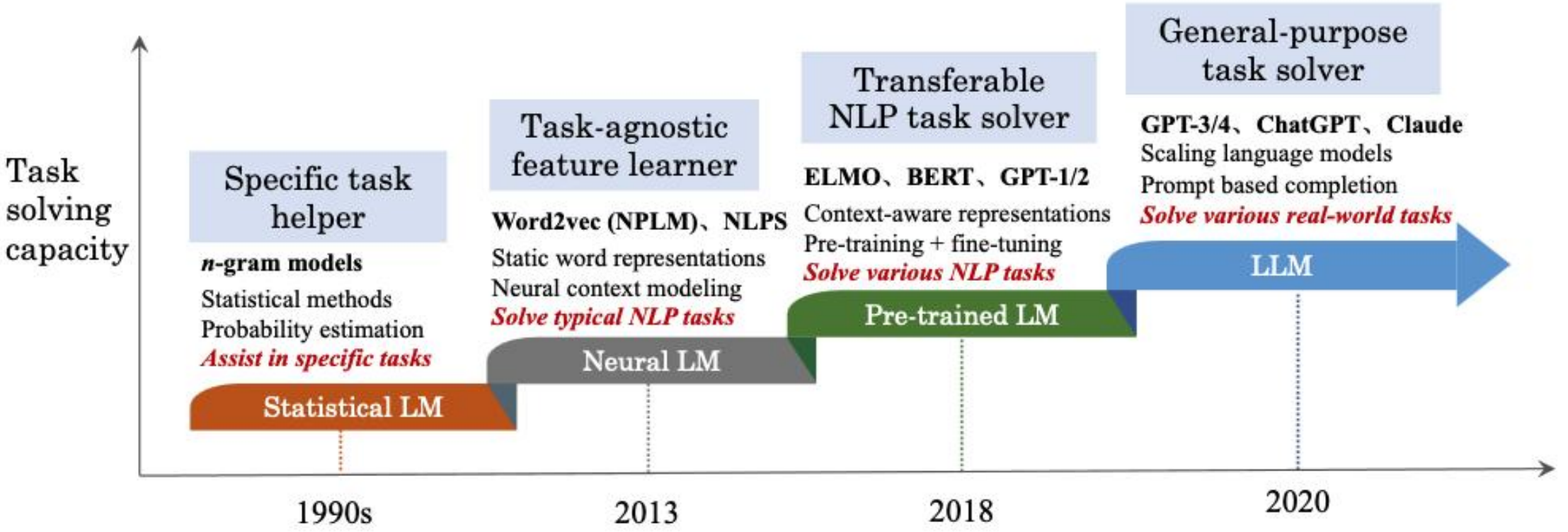



训练数据越来越大

数据种类越来越丰富



# AI模型能解决的任务越来越丰富




特定的任务

通用AI大模型

# How to learn?

## Supervised Learning

Given labels

e.g. classification task, regression task

## Unsupervised Learning

unknown labels

e.g. clustering task, data reconstruction task

## Self-supervised Learning

No labels

But the data itself can be the label: mask, next-word prediction

Tasks fall into two categories: pretext tasks, downstream tasks

# The goal of AI: minimize the loss function

Goal: to solve an optimization problem $\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}}\, L(\boldsymbol{\theta})$

$L$: loss function $\boldsymbol{\theta}$: parameters

**How to solve it?**

Take linear regression as an example:

1) Analytical solution
2) Gradient descent

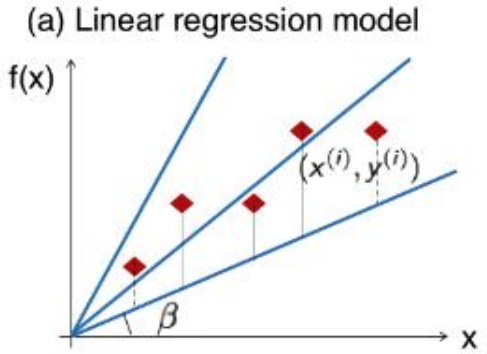
$$\beta^{(j+1)} \longleftarrow \beta^{(j)} - \eta \frac{d\mathcal{L}(\beta)}{d\beta}$$

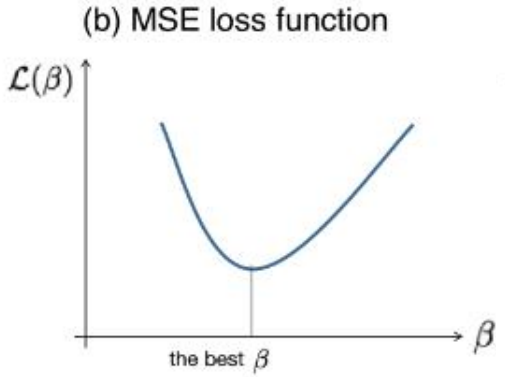
The simplest linear regression model: $y = \beta x$

Parameter: the slope $\beta$

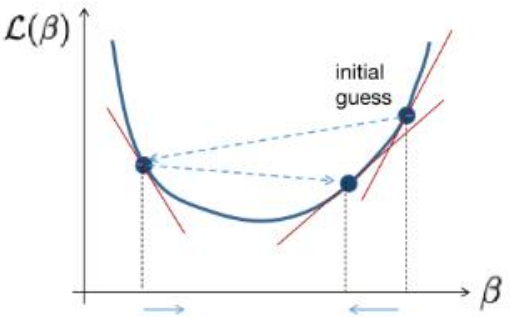
Loss function:

$$\mathcal{L}(\beta) = \frac{1}{n}\sum_{i=1}^{n}\left(y^{(i)} - \beta x^{(i)}\right)^2$$




$$\beta^{(j+1)} \longleftarrow \beta^{(j)} - \eta \frac{d\mathcal{L}(\beta)}{d\beta}$$



# Gradient Descent (GD)

Goal: to solve an optimization problem $\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} L(\boldsymbol{\theta})$

$L$: loss function $\boldsymbol{\theta}$: parameters

Suppose that $\boldsymbol{\theta}$ has two variables $\{\theta_1, \theta_2\}$

Randomly start at $\boldsymbol{\theta}^0 = \begin{bmatrix} \theta_1^0 \\ \theta_2^0 \end{bmatrix}$

$$\boldsymbol{\theta}^1 = \begin{bmatrix} \theta_1^1 \\ \theta_2^1 \end{bmatrix} = \begin{bmatrix} \theta_1^0 \\ \theta_2^0 \end{bmatrix} - \eta \begin{bmatrix} \partial L(\boldsymbol{\theta}^0)/\partial\theta_1 \\ \partial L(\boldsymbol{\theta}^0)/\partial\theta_2 \end{bmatrix} \Rightarrow \boldsymbol{\theta}^1 = \theta^0 - \eta\nabla L(\theta^0)$$

$$\boldsymbol{\theta}^2 = \begin{bmatrix} \theta_1^2 \\ \theta_2^2 \end{bmatrix} = \begin{bmatrix} \theta_1^1 \\ \theta_2^1 \end{bmatrix} - \eta \begin{bmatrix} \partial L(\boldsymbol{\theta}^1)/\partial\theta_1 \\ \partial L(\boldsymbol{\theta}^1)/\partial\theta_2 \end{bmatrix} \Rightarrow \boldsymbol{\theta}^2 = \theta^1 - \eta\nabla L(\theta^1)$$

……

until converge to $\theta^*$

**Gradient**:

$$\nabla L(\boldsymbol{\theta}) = \begin{bmatrix} \partial L(\boldsymbol{\theta})/\partial\theta_1 \\ \partial L(\boldsymbol{\theta})/\partial\theta_2 \end{bmatrix}$$

Learning rate: $\eta$

# An example to calculate the gradient of loss function

Loss function: $L(\boldsymbol{\theta}) = \frac{1}{2}(\boldsymbol{\theta} - \boldsymbol{y})^2$

$$L(\boldsymbol{\theta}) = \frac{1}{2}\left(\begin{bmatrix}\theta_1 - y_1\\ \theta_2 - y_2\end{bmatrix}\right)^2$$

Gradient:

$$\nabla L(\boldsymbol{\theta}) = \begin{bmatrix}\partial L(\boldsymbol{\theta})/\partial\theta_1\\ \partial L(\boldsymbol{\theta})/\partial\theta_2\end{bmatrix} = \ ?$$

$$\nabla L\left(\begin{bmatrix}\theta_1\\ \theta_2\end{bmatrix}\right) = \begin{bmatrix}\theta_1 - y_1\\ \theta_2 - y_2\end{bmatrix}$$

# Gradient Descent to train Neural Networks

Network parameters $\theta = \{w_1, w_2, \cdots, b_1, b_2, \cdots\}$

Initiate Parameters

$\theta^0 \longrightarrow \theta^1 \longrightarrow \theta^2 \longrightarrow \cdots\cdots$

$Compute\ \nabla L(\theta^0)$

$\theta^1 = \theta^0 - \eta \nabla L(\theta^0)$

$Compute\ \nabla L(\theta^1)$

$\theta^2 = \theta^1 - \eta \nabla L(\theta^1)$

$$\nabla L(\theta) = \begin{bmatrix} \partial L(\theta)/\partial w_1 \\ \partial L(\theta)/\partial w_2 \\ \vdots \\ \partial L(\theta)/\partial b_1 \\ \partial L(\theta)/\partial b_2 \\ \vdots \end{bmatrix}$$

ANNs have Millions of parameters ……

To compute the gradients efficiently in ANNs, we use the error ***back-propagation***.

# Some valuable materials for self-learning

**highly recommended by 刘泉影**

# Caltech Machine Learning course

## CS 156, by Prof. Yaser Abu-Mostafa

https://www.youtube.com/watch?v=mbyG85GZ0PI&list=PLD63A284B7615313A


Lecture 01 - The Learning Problem
caltech


2 Lecture 02 - Is Learning Feasible?
caltech


3 Lecture 03 -The Linear Model I
caltech


4 Lecture 04 - Error and Noise
caltech

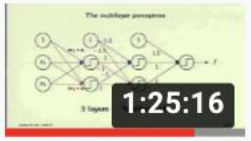

5 Lecture 05 - Training Versus Testing
caltech


6 Lecture 06 - Theory of Generalization
caltech

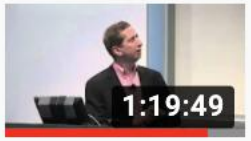

7 Lecture 07 - The VC Dimension
caltech


8 Lecture 08 - Bias-Variance Tradeoff
caltech


9 Lecture 09 - The Linear Model II
caltech

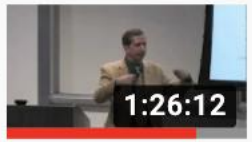



10 Lecture 10 - Neural Networks
caltech




11 Lecture 11 - Overfitting
caltech


12 Lecture 12 - Regularization
caltech




13 Lecture 13 - Validation
caltech


14 Lecture 14 - Support Vector Machines
caltech


15 Lecture 15 - Kernel Methods
caltech


16 Lecture 16 - Radial Basis Functions
caltech


17 Lecture 17 - Three Learning Principles
caltech


18 Lecture 18 - Epilogue
caltech

# Stanford Machine Learning course

## CS 229, by Andrew Ng

https://www.bilibili.com/video/BV19e411W7ga/

| | | |
|---|---|---|
| P1 | 1. Lecture 1 – Welcome | 1:15:20 |
| P2 | 2. Lecture 2 – Linear Regression and G… | 1:18:17 |
| P3 | 3. Lecture 3 – Locally Weighted & Logi… | 1:19:35 |
| P4 | 4. Lecture 4 – Perceptron & Generaliz… | 1:22:02 |
| P5 | 5. Lecture 5 – GDA & Naive Bayes | 1:18:52 |
| P6 | 6. Lecture 6 – Support Vector Machines | 1:20:57 |
| P7 | 7. Lecture 7 – Kernels | 1:20:25 |
| P8 | 8. Lecture 8 – Data Splits, Models & C… | 1:23:26 |
| P9 | 9. Lecture 9 – Approx_Estimation Erro… | 1:26:03 |
| P10 | 10. Lecture 10 – Decision Trees and E… | 1:20:41 |
| P11 | 11. Lecture 11 – Introduction to Neural … | 1:20:14 |
| P12 | 12. Lecture 12 – Backprop & Improvin… | 1:16:38 |
| P13 | 13. Lecture 13 – Debugging ML Model… | 1:18:55 |
| P14 | 14. Lecture 14 – Expectation–Maximiz… | 1:20:32 |
| P15 | 15. Lecture 15 – EM Algorithm & Fact… | 1:19:48 |
| P16 | 16. Lecture 16 – Independent Compon… | 1:18:10 |
| P17 | 17. Lecture 17 – MDPs & Value_Policy I… | 1:19:15 |
| P18 | 18. Lecture 18 – Continous State MDP… | 1:20:15 |
| P19 | 19. Lecture 19 – Reward Model & Line… | 1:21:07 |
| P20 | 20. Lecture 20 – RL Debugging and … | 1:12:43 |

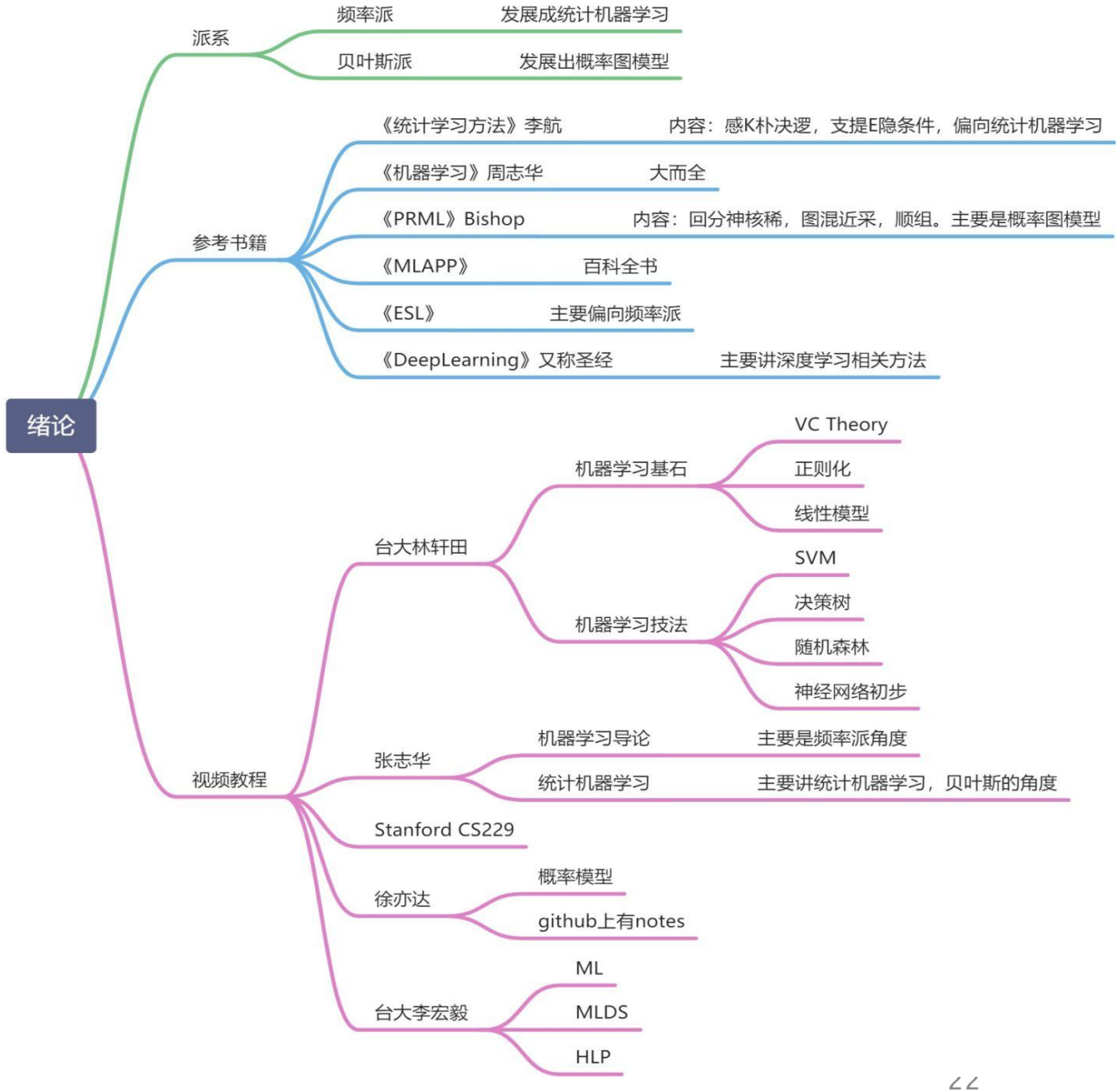
# B站大学 【机器学习】白板推导

by shuhai008

https://www.bilibili.com/video/BV1aE411o7qd

纯理论推导


# 机器学习白板推导



# 李飞飞 Stanford CS231n

2021 年

- A brief history of computer vision
- Deep Learning for Computer Vision Lecture 1 - Overview
- Lecture 2: Image Classification with Linear Classifiers
- Lecture 3: Regularization and Optimization
- Lecture 4: Neural Networks and Backpropagation
- Lecture 5: Image Classification with CNNs
- Lecture 6: CNN Architectures
- Lecture 7: Training Neural Networks
- Lecture 8: Visualizing and Understanding
- Lecture 9: Object Detection and Image Segmentation
- Lecture 10: Recurrent Neural Networks
- Lecture 11: Attention and Transformers
- Lecture 12: Video Understanding
- Lecture 13: Generative Models
- Lecture 14: Self-supervised Learning
- CS231N: Low-Level Vision
- Lecture 16: 3D Vision

最新的课程没用找到录屏

B站上只有2017年
https://www.bilibili.com/video/BV1nJ411z7fe/

2019年的课件
https://www.bilibili.com/video/BV1hi4y1t7kF/

# 李飞飞 - Stanford CS231n (Deep Learning for Computer Vision)

## 2022年 http://cs231n.stanford.edu/2022/schedule.html

| | | |
|---|---|---|
| --- | **Deep Learning Basics** | |
| 03/31 | **Lecture 2: Image Classification with Linear Classifiers**<br>The data-driven approach<br>K-nearest neighbor<br>Linear Classifiers<br>Algebraic / Visual / Geometric viewpoints<br>SVM and Softmax loss<br>[slides] | Image Classification Problem<br>Linear Classification |
| 04/01 | Python / Numpy Review Session<br>[Colab] [Tutorial] | 1:30-2:30pm PT |
| 04/05 | **Lecture 3: Regularization and Optimization**<br>Regularization<br>Stochastic Gradient Descent<br>Momentum, AdaGrad, Adam<br>Learning rate schedules<br>[slides] | Optimization |
| 04/07 | **Lecture 4: Neural Networks and Backpropagation**<br>Multi-layer Perceptron<br>Backpropagation<br>[slides] | Backprop<br>Linear backprop example<br>Suggested Readings:<br>1. Why Momentum Really Works |
| --- | **Perceiving and Understanding the Visual World** | |
| 04/12 | **Lecture 5: Image Classification with CNNs**<br>History<br>Higher-level representations, image features<br>Convolution and pooling<br>[slides] | Convolutional Networks |
| 04/14 | **Lecture 6: CNN Architectures**<br>Batch Normalization<br>Transfer learning<br>AlexNet, VGG, GoogLeNet, ResNet<br>[slides] | AlexNet, VGGNet, GoogLeNet, ResNet |
| 04/19 | **Lecture 7: Training Neural Networks**<br>Activation functions<br>Data processing<br>Weight initialization | Neural Networks, Parts 1, 2, 3<br>Suggested Readings:<br>1. Stochastic Gradient Descent Tricks<br>2. Efficient Backprop |

## 2023年 ht

| | | |
|---|---|---|
| --- | **Deep Learning Basics** | |
| 04/06 | **Lecture 2: Image Classification with Linear Classifiers**<br>The data-driven approach<br>K-nearest neighbor<br>Linear Classifiers<br>Algebraic / Visual / Geometric viewpoints<br>SVM and Softmax loss<br>[slides] | Image Classification Problem<br>Linear Classification |
| 04/07 | Python / Numpy Review Session<br>[Colab] [Tutorial] | 1:30-2:20pm PT |
| 04/11 | **Lecture 3: Regularization and Optimization**<br>Regularization<br>Stochastic Gradient Descent<br>Momentum, AdaGrad, Adam<br>Learning rate schedules<br>[slides] | Optimization |
| 04/13 | **Lecture 4: Neural Networks and Backpropagation**<br>Multi-layer Perceptron<br>Backpropagation<br>[slides] | Backprop<br>Linear backprop example<br>Suggested Readings:<br>1. Why Momentum Really Works |
| --- | **Perceiving and Understanding the Visual World** | |
| 04/18 | **Lecture 5: Image Classification with CNNs**<br>History<br>Higher-level representations, image features<br>Convolution and pooling<br>[slides] | Convolutional Networks |
| 04/20 | **Lecture 6: CNN Architectures**<br>Batch Normalization<br>Transfer learning<br>AlexNet, VGG, GoogLeNet, ResNet<br>[slides] | AlexNet, VGGNet, GoogLeNet, ResNet |
| 04/25 | **Lecture 7: Training Neural Networks**<br>Activation functions<br>Data processing<br>Weight initialization | Neural Networks, Parts 1, 2, 3<br>Suggested Readings:<br>1. Stochastic Gradient Descent Tricks<br>2. Efficient Backprop |

# 李飞飞 – Stanford CS231n (Deep Learning for Computer Vision)

## 2022年 http://cs231n.stanford.edu/2022/schedule.html

| | | |
|---|---|---|
| 04/21 | **Lecture 8: Visualizing and Understanding**<br>Feature visualization and inversion<br>Adversarial examples<br>DeepDream and style transfer<br>[slides] | |
| 04/26 | **Lecture 9: Object Detection and Image Segmentation**<br>Single-stage detectors<br>Two-stage detectors<br>Semantic/Instance/Panoptic segmentation<br>[slides] | FCN, R-CNN, Fast R-CNN, Faster R-CNN, YOLO |
| 04/28 | **Lecture 10: Recurrent Neural Networks**<br>RNN, LSTM, GRU<br>Language modeling<br>Image captioning<br>Sequence-to-sequence<br>[slides] | Suggested Readings:<br>1. DL book RNN chapter<br>2. Understanding LSTM Networks |
| 05/03 | **Lecture 11: Attention and Transformers**<br>Self-Attention<br>Transformers<br>[slides] | Suggested Readings:<br>1. Attention is All You Need [Original Transformers Paper]<br>2. Attention? Attention [Blog by Lilian Weng]<br>3. The Illustrated Transformer [Blog by Jay Alammar]<br>4. ViT: Transformers for Image Recognition [Paper] [Blog] [Video]<br>5. DETR: End-to-End Object Detection with Transformers [Paper] [Blog] [Video] |
| 05/5 | **Lecture 12: Video Understanding**<br>Video classification<br>3D CNNs<br>Two-stream networks<br>Multimodal video understanding<br>[slides] | |

## 2023年 http://

| | | |
|---|---|---|
| 04/27 | **Lecture 8: Recurrent Neural Networks**<br>RNN, LSTM, GRU<br>Language modeling<br>Image captioning<br>Sequence-to-sequence<br>[slides] | Suggested Readings:<br>1. DL book RNN chapter<br>2. Understanding LSTM Networks |
| 05/02 | **Lecture 9: Attention and Transformers**<br>Self-Attention<br>Transformers<br>[slides] | Suggested Readings:<br>1. Attention is All You Need [Original Transformers Paper]<br>2. Attention? Attention [Blog by Lilian Weng]<br>3. The Illustrated Transformer [Blog by Jay Alammar]<br>4. ViT: Transformers for Image Recognition [Paper] [Blog] [Video]<br>5. DETR: End-to-End Object Detection with Transformers [Paper] [Blog] [Video] |
| 05/04 | **Lecture 10: Video Understanding**<br>Video classification<br>3D CNNs<br>Two-stream networks<br>Multimodal video understanding<br>[slides] | |
| 05/09 | **Lecture 11: Object Detection and Image Segmentation**<br>Single-stage detectors<br>Two-stage detectors<br>Semantic/Instance/Panoptic segmentation<br>[slides] | FCN, R-CNN, Fast R-CNN, Faster R-CNN, YOLO |
| 05/11 | **Lecture 12: Visualizing and Understanding**<br>Feature visualization and inversion<br>Adversarial examples<br>DeepDream and style transfer<br>[slides] | |

# 李飞飞 – Stanford CS231n (Deep Learning for Computer Vision)

2022年 http://cs231n.stanford.edu/2022/schedule.html

| | | |
|---|---|---|
| --- | **Reconstructing and Interacting with the Visual World** | |
| 05/12 | **Lecture 13: Generative Models**<br>Supervised vs. Unsupervised learning<br>Pixel RNN, Pixel CNN<br>Variational Autoencoders<br>Generative Adversarial Networks<br>[slides] | Suggested Readings:<br>1. Image GPT: Generative Pretraining<br>Pixels [Paper] [Blog] |
| 05/17 | **Lecture 14: Self-supervised Learning**<br>Pretext tasks<br>Contrastive learning<br>Multisensory supervision<br>[slides] | Suggested Readings:<br>1. Lilian Weng Blog Post<br>2. DINO: Emerging Properties in Self-<br>Supervised Vision Transformers [Pa<br>[Blog] [Video] |
| 05/19 | **Lecture 15: Low-Level Vision**<br>**(Guest Lecture by Prof. Jia Deng from Princeton University)**<br>Optical flow<br>Depth estimation<br>Stereo vision<br>[slides] | |
| 05/24 | **Lecture 16: 3D Vision**<br>3D shape representations<br>Shape reconstruction<br>Neural implicit representations<br>[slides] | |

2023年 http://cs231n.stanford.edu/schedule.html

| | | |
|---|---|---|
| --- | **Generative and Interactive Visual Intelligence** | |
| 05/16 | **In-Class Midterm** | 12:00-1:20pm |
| 05/18 | **Lecture 13: Self-supervised Learning**<br>Pretext tasks<br>Contrastive learning<br>Multisensory supervision<br>[slides] | Suggested Readings:<br>1. Lilian Weng Blog Post<br>2. DINO: Emerging Properties in Self-<br>Supervised Vision Transformers [Paper]<br>[Blog] [Video] |
| 05/23 | **Lecture 14: Robot Learning**<br>Deep Reinforcement Learning<br>Model Learning<br>Robotic Manipulation<br>[slides] | |
| 05/25 | **Lecture 15: Generative Models (Guest Lecture by Dr. Ruiqi Gao from Google DeepMind)**<br>Generative Adversarial Network<br>Diffusion models<br>Autoregressive models<br>[slides] | |
| 05/30 | **Lecture 16: 3D Vision**<br>3D shape representations<br>Shape reconstruction<br>Neural implicit representations<br>[slides] | |

# 李飞飞 – Stanford CS231n (Deep Learning for Computer Vision)

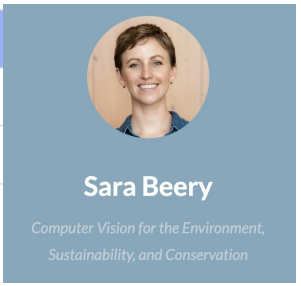
2022年 http://cs231n.stanford.edu/2022/schedule.html

| | | |
|---|---|---|
| --- | **Reconstructing and Interacting with the Visual World** | |
| 05/12 | **Lecture 13: Generative Models**<br>Supervised vs. Unsupervised learning<br>Pixel RNN, Pixel CNN<br>Variational Autoencoders<br>Generative Adversarial Networks<br>[slides] | Suggested Readings:<br>1. Image GPT: Generative Pretraining<br>Pixels [Paper] [Blog] |
| 05/17 | **Lecture 14: Self-supervised Learning**<br>Pretext tasks<br>Contrastive learning<br>Multisensory supervision<br>[slides] | Suggested Readings:<br>1. Lilian Weng Blog Post<br>2. DINO: Emerging Properties in Self-<br>Supervised Vision Transformers [Pa<br>[Blog] [Video] |
| 05/19 | **Lecture 15: Low-Level Vision**<br>**(Guest Lecture by Prof. Jia Deng from Princeton University)**<br>Optical flow<br>Depth estimation<br>Stereo vision<br>[slides] | |
| 05/24 | **Lecture 16: 3D Vision**<br>3D shape representations<br>Shape reconstruction<br>Neural implicit representations<br>[slides] | |
| --- | **Human-Centered Applications and Implications** | |
| 05/26 | **Lecture 17: Human-Centered Artificial Intelligence**<br>AI & healthcare | |
| 05/31 | **Lecture 18: Fairness in Visual Recognition**<br>**(Guest Lecture by Prof. Olga Russakovsky from Princeton University)** | |

2023年 http://cs231n.stanford.edu/schedule.html

| | | |
|---|---|---|
| --- | **Generative and Interactive Visual Intelligence** | |
| 05/16 | **In-Class Midterm** | 12:00-1:20pm |
| 05/18 | **Lecture 13: Self-supervised Learning**<br>Pretext tasks<br>Contrastive learning<br>Multisensory supervision<br>[slides] | Suggested Readings:<br>1. Lilian Weng Blog Post<br>2. DINO: Emerging Properties in Self-<br>Supervised Vision Transformers [Paper]<br>[Blog] [Video] |
| 05/23 | **Lecture 14: Robot Learning**<br>Deep Reinforcement Learning<br>Model Learning<br>Robotic Manipulation<br>[slides] | |
| 05/25 | **Lecture 15: Generative Models (Guest Lecture by Dr. Ruiqi Gao from Google DeepMind)**<br>Generative Adversarial Network<br>Diffusion models<br>Autoregressive models<br>[slides] | |
| 05/30 | **Lecture 16: 3D Vision**<br>3D shape representations<br>Shape reconstruction<br>Neural implicit representations<br>[slides] | |
| --- | **Human-Centered Applications and Implications** | |
| 06/01 | **Lecture 17: Human-Centered Artificial Intelligence** | |
| 06/06 | **Lecture 18: Guest Lecture by Prof. Sara Beery from MIT** | |

**机器人+强化学习+大模型**
**通往具身智能**



# Christ Manning – Stanford CS224n (Natural Language Processing with Deep Learning)

2023年 https://web.stanford.edu/class/cs224n/index.html#schedule

B站 2021年课程的视频合集 https://www.bilibili.com/video/BV18Y411p79k

| | | |
|---|---|---|
| Tue Jan 10<br><br>**Week 1** | Word Vectors *(by John Hewitt)*<br>[slides] [notes]<br><br>Gensim word vectors example:<br>[code] [preview] | Suggested Readings:<br>1. Efficient Estimation of Word Representations in Vector Space (original word2vec paper)<br>2. Distributed Representations of Words and Phrases and their Compositionality (negative sampling paper) |
| Thu Jan 12 | Word Vectors, Word Window Classification, Language Models<br>[slides] [notes] | Suggested Readings:<br>1. GloVe: Global Vectors for Word Representation (original GloVe paper)<br>2. Improving Distributional Similarity with Lessons Learned from Word Embeddings<br>3. Evaluation methods for unsupervised word embeddings<br><br>Additional Readings:<br>1. A Latent Variable Model Approach to PMI-based Word Embeddings<br>2. Linear Algebraic Structure of Word Senses, with Applications to Polysemy<br>3. On the Dimensionality of Word Embedding |

# Christ Manning – Stanford CS224n (Natural Language Processing with Deep Learning)

2023年 https://web.stanford.edu/class/cs224n/index.html#schedule

| | | |
|---|---|---|
| Tue Jan 17<br>**Week 2** | Backprop and Neural Networks<br>[slides] [notes] | Suggested Readings:<br>1. matrix calculus notes<br>2. Review of differential calculus<br>3. CS231n notes on network architectures<br>4. CS231n notes on backprop<br>5. Derivatives, Backpropagation, and Vectorization<br>6. Learning Representations by Backpropagating Errors (seminal Rumelhart et al. backpropagation paper)<br><br>Additional Readings:<br>1. Yes you should understand backprop<br>2. Natural Language Processing (Almost) from Scratch |
| Thu Jan 19 | Dependency Parsing<br>[slides] [notes]<br>[slides (annotated)] | Suggested Readings:<br>1. Incrementality in Deterministic Dependency Parsing<br>2. A Fast and Accurate Dependency Parser using Neural Networks<br>3. Dependency Parsing<br>4. Globally Normalized Transition-Based Neural Networks<br>5. Universal Stanford Dependencies: A cross-linguistic typology<br>6. Universal Dependencies website<br>7. Jurafsky & Martin Chapter 14 |

# Christ Manning – Stanford CS224n (Natural Language Processing with Deep Learning)

| | | |
|---|---|---|
| Tue Jan 24<br>**Week 3** | Recurrent Neural Networks and Language Models<br>[slides] [notes (lectures 5 and 6)] | Suggested Readings:<br>1. N-gram Language Models (textbook chapter)<br>2. The Unreasonable Effectiveness of Recurrent Neural Networks (blog post overview)<br>3. Sequence Modeling: Recurrent and Recursive Neural Nets (Sections 10.1 and 10.2)<br>4. On Chomsky and the Two Cultures of Statistical Learning<br>5. Sequence Modeling: Recurrent and Recursive Neural Nets (Sections 10.3, 10.5, 10.7-10.12)<br>6. Learning long-term dependencies with gradient descent is difficult (one of the original vanishing gradient papers)<br>7. On the difficulty of training Recurrent Neural Networks (proof of vanishing gradient problem)<br>8. Vanishing Gradients Jupyter Notebook (demo for feedforward networks)<br>9. Understanding LSTM Networks (blog post overview) |
| Thu Jan 26 | Seq2Seq, MT, Subword Models<br>[slides] [notes (lectures 5 and 6)] | Suggested Readings:<br>1. Statistical Machine Translation slides, CS224n 2015 (lectures 2/3/4)<br>2. Statistical Machine Translation (book by Philipp Koehn)<br>3. BLEU (original paper)<br>4. Sequence to Sequence Learning with Neural Networks (original seq2seq NMT paper)<br>5. Sequence Transduction with Recurrent Neural Networks (early seq2seq speech recognition paper)<br>6. Neural Machine Translation by Jointly Learning to Align and Translate (original seq2seq+attention paper)<br>7. Attention and Augmented Recurrent Neural Networks (blog post overview)<br>8. Massive Exploration of Neural Machine Translation Architectures (practical advice for hyperparameter choices)<br>9. Achieving Open Vocabulary Neural Machine Translation with Hybrid Word-Character Models<br>10. Revisiting Character-Based Neural Machine Translation |
| Tue Jan 31<br>**Week 4** | Final Projects: Custom and Default; Practical Tips<br>[slides] [notes] | Suggested Readings:<br>1. Practical Methodology (*Deep Learning* book chapter) |
| Thu Feb 2 | Self-Attention and Transformers *(by John Hewitt)*<br>[slides] [notes] | Suggested Readings:<br>1. Default Project Handout<br>2. Attention Is All You Need<br>3. The Illustrated Transformer<br>4. Transformer (Google AI blog post)<br>5. Layer Normalization<br>6. Image Transformer<br>7. Music Transformer: Generating music with long-term structure |
| Tue Feb 7<br>**Week 5** | Pretraining *(by John Hewitt)*<br>[slides] | Suggested Readings:<br>1. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding<br>2. Contextual Word Representations: A Contextual Introduction<br>3. The Illustrated BERT, ELMo, and co.<br>4. Martin & Jurafsky Chapter on Transfer Learning |
| Thu Feb 9 | Natural Language Generation *(by Xiang Lisa Li)*<br>[slides] | Suggested Readings:<br>1. The Curious Case of Neural Text Degeneration<br>2. Get To The Point: Summarization with Pointer-Generator Networks<br>3. Hierarchical Neural Story Generation<br>4. How NOT To Evaluate Your Dialogue System |

# Christ Manning – Stanford CS224n (Natural Language Processing with Deep Learning)

| | | |
|---|---|---|
| Tue Feb 14<br>**Week 6** | Prompting, Reinforcement Learning from Human Feedback *(by Jesse Mu)*<br>[slides] | Suggested Readings:<br>1. Language Models are Few-Shot Learners<br>2. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models<br>3. Finetuned Language Models Are Zero-Shot Learners<br>4. Learning to summarize from human feedback |
| Thu Feb 16 | Question Answering<br>[slides] | Suggested readings:<br>1. SQuAD: 100,000+ Questions for Machine Comprehension of Text<br>2. Bidirectional Attention Flow for Machine Comprehension<br>3. Reading Wikipedia to Answer Open-Domain Questions<br>4. Latent Retrieval for Weakly Supervised Open Domain Question Answering<br>5. Dense Passage Retrieval for Open-Domain Question Answering<br>6. Learning Dense Representations of Phrases at Scale |
| Tue Feb 21<br>**Week 7** | ConvNets, Tree Recursive Neural Networks and Constituency Parsing<br>[slides] | Suggested readings:<br>1. Convolutional Neural Networks for Sentence Classification<br>2. Improving neural networks by preventing co-adaptation of feature detectors<br>3. A Convolutional Neural Network for Modelling Sentences<br>4. Parsing with Compositional Vector Grammars.<br>5. Constituency Parsing with a Self-Attentive Encoder |
| Thu Feb 23 | Insights between NLP and Linguistics *(by Isabel Papadimitriou)*<br>[slides] | |
| Tue Feb 28<br>**Week 8** | Code Generation *(by Gabriel Poesia)*<br>[slides] | Suggested readings:<br>1. Program Synthesis with Large Language Models<br>2. Competition-level code generation with AlphaCode<br>3. Evaluating Large Language Models Trained on Code |
| Wed Mar 1 | Training Large Language Models *(by John Hewitt)* | 3:30pm - 4:20pm<br>Skilling Auditorium |
| Thu Mar 2 | Multimodal Deep Learning *(by Douwe Kiela)*<br>[slides] | |
| Tue Mar 7<br>**Week 9** | Coreference Resolution<br>[slides] | Suggested readings:<br>1. Coreference Resolution Chapter from Jurafsky and Martin<br>2. End-to-end Neural Coreference Resolution |
| Thu Mar 9 | Analysis and Interpretability Basics *(by John Hewitt)*<br>[slides] | |
| Fri Mar 10 | Latex Tutorial *(by Rishi Desai)* | 3:30pm - 4:20pm<br>Skilling Auditorium |
| Tue Mar 14<br>**Week 10** | Model Interpretability and Editing *(by Been Kim)*<br>[slides] | |
| Thu Mar 16 | Final Project Emergency Assistance (no lecture) | Extra project office hours available during usual lecture time, see Ed. |
| Sat Mar 18 | | |
| Monday Mar 20 | Poster Session | 5pm-9pm [More details]<br>Location: Tressider Oak Lounge |

# B站大学　跟李沐学AI

https://space.bilibili.com/1567748478

## 【完结】动手学深度学习 PyTorch版 76

播放全部　更多


【完结】73 – 课程总结和进阶学习【动手学深度学习v2】

24.9万　2021-8-25


72 优化算法【动手学深度学习v2】

6.9万　2021-8-24


71 目标检测竞赛总结【动手学深度学习v2】

3.5万　2021-8-22


70 BERT微调【动手学深度学习v2】

6.8万　2021-8-19

## 合集·【更新中】AI 论文精读 53

播放全部　更多


如何读论文【论文精读·1】

35.6万　2021-10-6


9年后重读深度学习奠基作之一：AlexNet【论文精读·2】

23.8万　2021-10-14


AlexNet论文逐段精读【论文精读】

22.8万　2021-10-15


撑起计算机视觉半边天的ResNet【论文精读】

17.6万　2021-10-21

# B站大学　bryanyzhu

https://space.bilibili.com/511378644

朱老师的视频以CV为主

后来转到多模态模型

# B站大学　清华NLP刘知远团队大模型公开课

https://www.bilibili.com/video/BV1UG411p7zv

# AI Hands-on (pytorch)

By 楼可心、王淞、钱富元

# Error Back-Propagation

# Recall Calculus: Chain Rule

***Case 1***

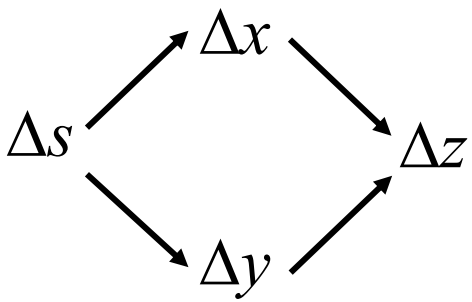
$$y = g(x) \quad z = h(y)$$

$$\Delta x \to \Delta y \to \Delta z$$

$$\frac{dz}{dx} = \frac{dz}{dy}\frac{dy}{dx}$$

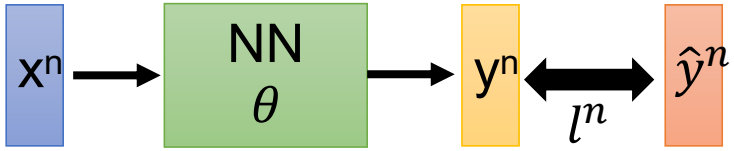
***Case 2***

$$x = g(s) \qquad y = h(s) \qquad z = k(x, y)$$



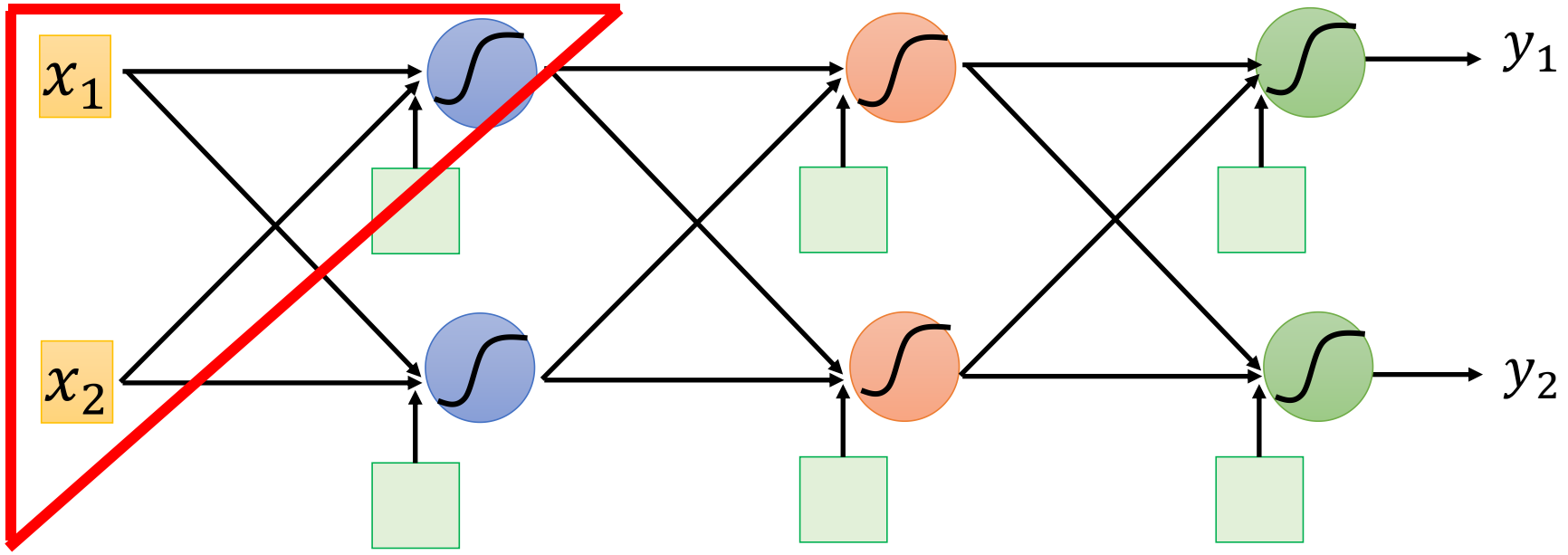

$$\frac{dz}{ds} = \frac{\partial z}{\partial x}\frac{dx}{ds} + \frac{\partial z}{\partial y}\frac{dy}{ds}$$

# Error Backpropagation

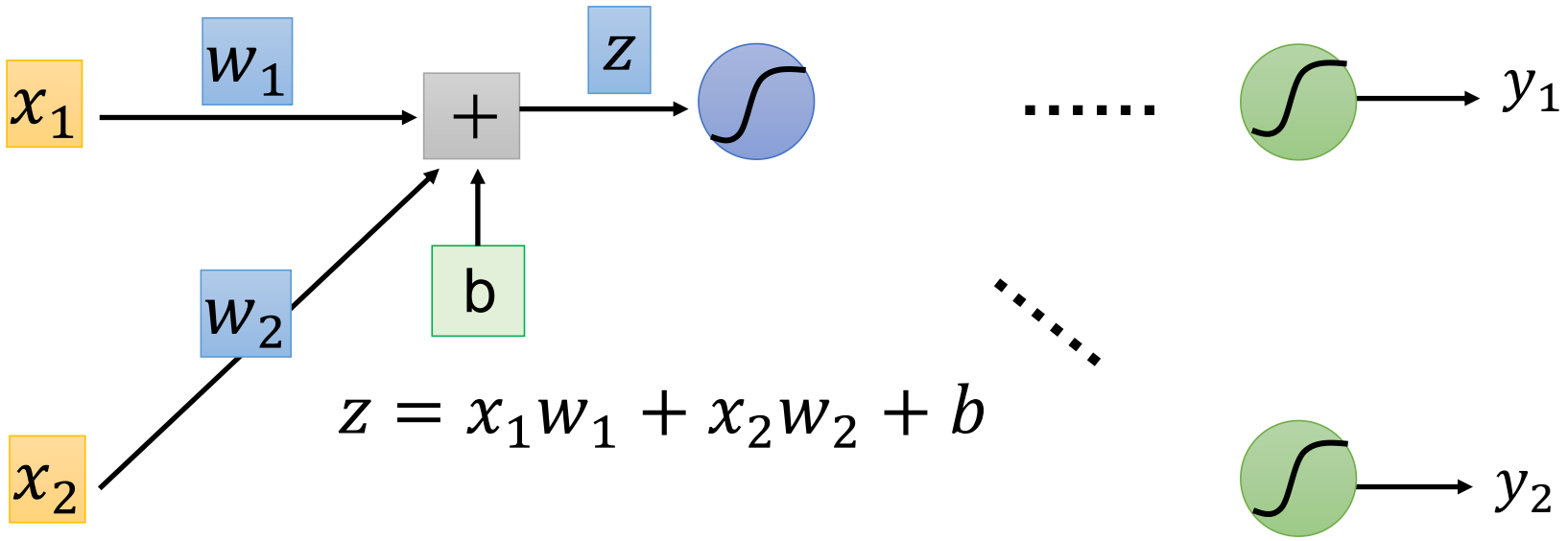



Backpropagation computes how slightly changing each synapse strength (**weight**) would change the network's **loss**, using the chain rule.

$$L(\theta) = \sum_{n=1}^{N} l^n(\theta) \quad \Rightarrow \quad \frac{\partial L(\theta)}{\partial w} = \sum_{n=1}^{N} \frac{\partial l^n(\theta)}{\partial w}$$



# BP in fully-connected neural networks (fcNN)




$$\frac{\partial l}{\partial w} = ? \quad \frac{\partial l}{\partial z}\frac{\partial z}{\partial w}$$

(Chain rule)

***Forward pass:***

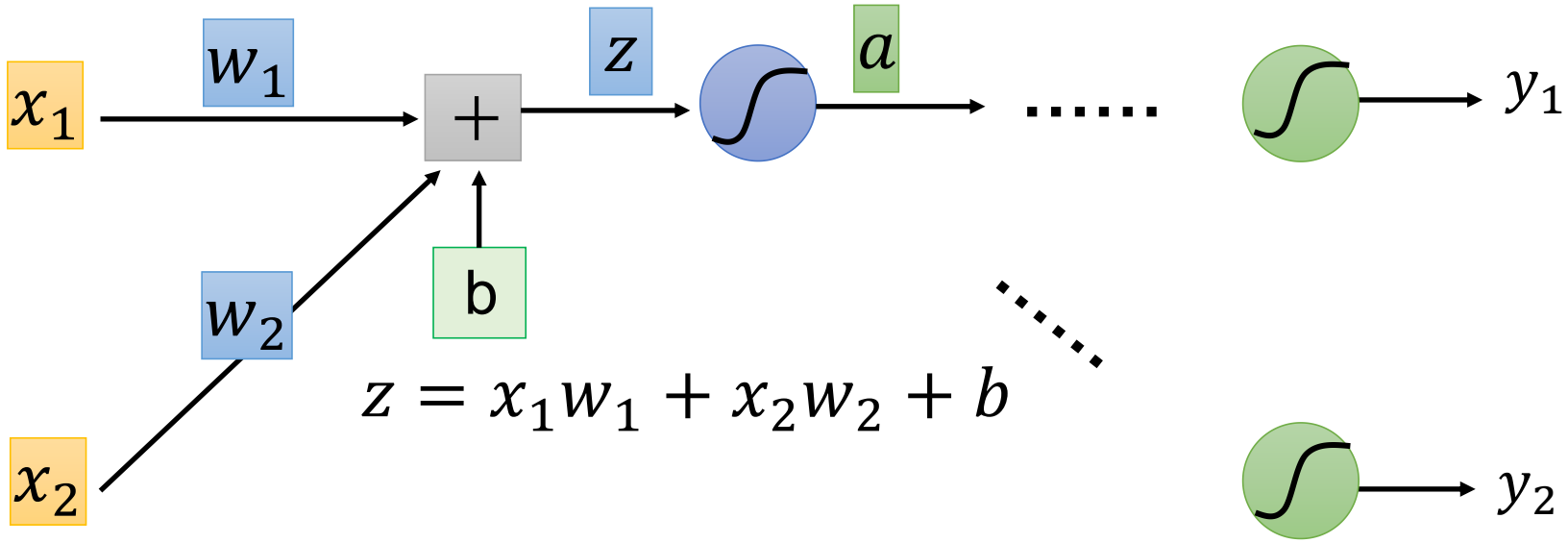
Compute $\partial z/\partial w$ for all parameters

***Backward pass:***

Compute $\partial l/\partial z$ for all activation function inputs z

# Backpropagation – Forward pass

Compute $\partial z/\partial w$ for all parameters



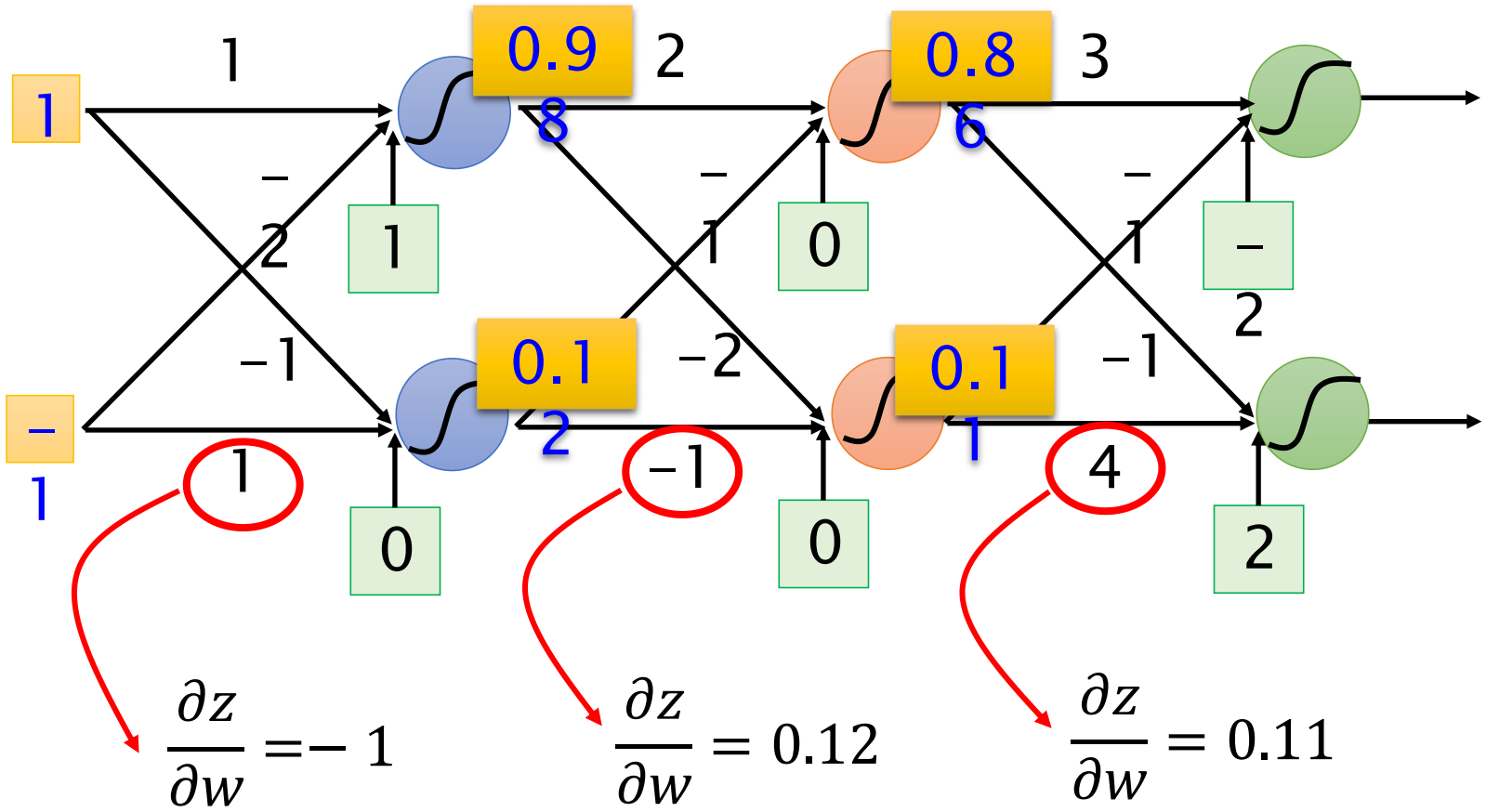

$\partial z/\partial w_1 = ?\ x_1$

$\partial z/\partial w_2 = ?\ x_2$

The value of the input connected by the weight

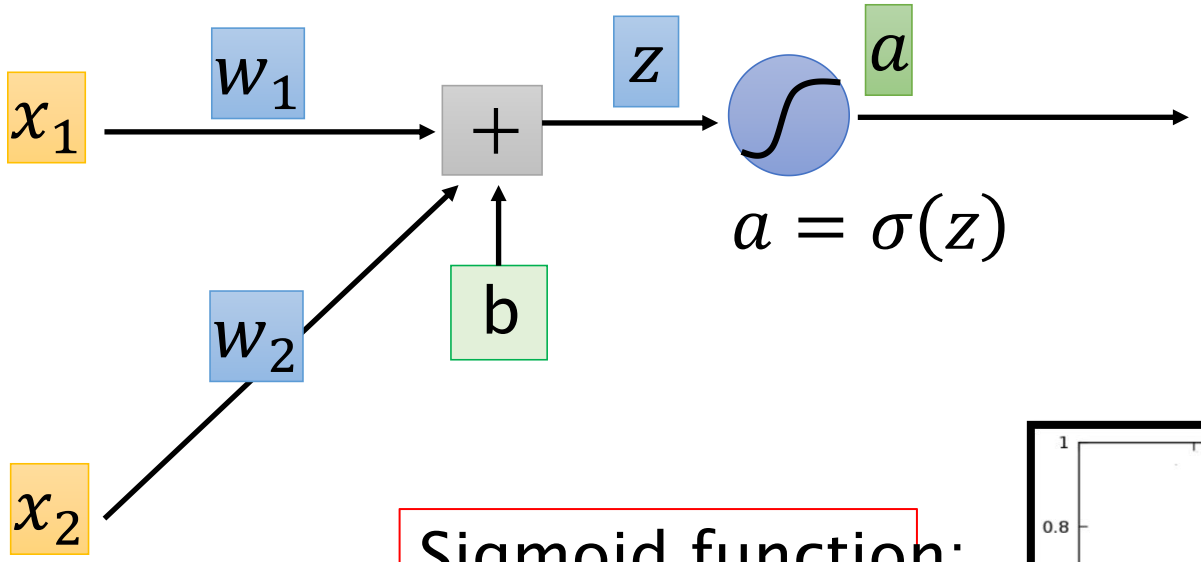
# Backpropagation – Forward pass

Compute $\partial z/\partial w$ for all parameters



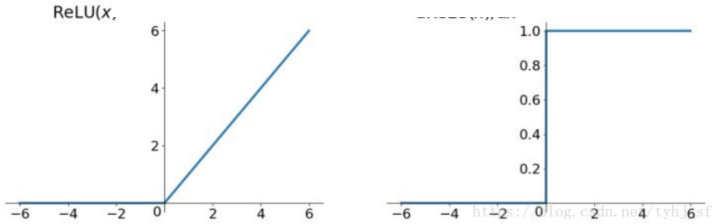
# Backpropagation – Backward pass

Compute $\partial a/\partial z$ for all *activation function* inputs z




$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

$$Relu = max(0, x)$$

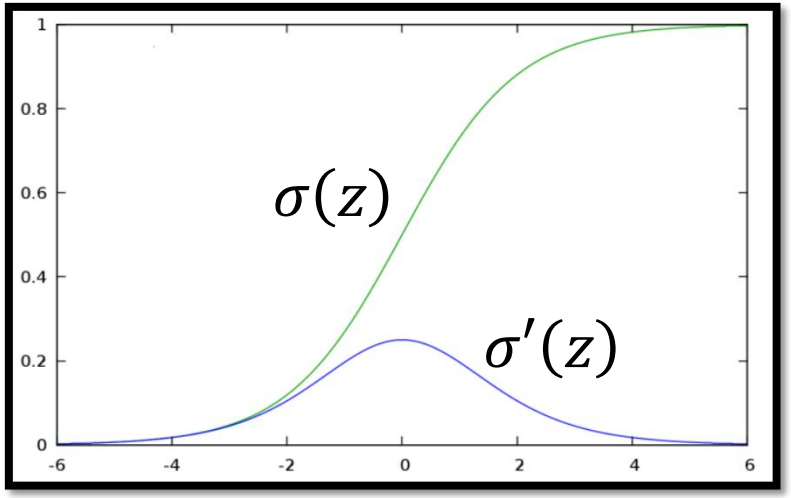



$$\frac{\partial l}{\partial z} = \frac{\partial l}{\partial a}\frac{\partial a}{\partial z}$$

Sigmoid function:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

$\sigma'(z)$



# Backpropagation – Backward pass

## Activation function and its derivative

- ReLU: $h(a) = \max\{0, a\}$ $\quad h'(a) = \begin{cases} 0, & \text{for } a < 0; \\ 1, & \text{for } a > 0 \end{cases}$
- Leaky ReLU: $h(a) = \begin{cases} \alpha a, & \text{for } a \leq 0 \\ a, & \text{for } a > 0 \end{cases}$

$$h'(a) = \begin{cases} \alpha, & \text{for } a \leq 0 \\ 1, & \text{for } a > 0 \end{cases}$$

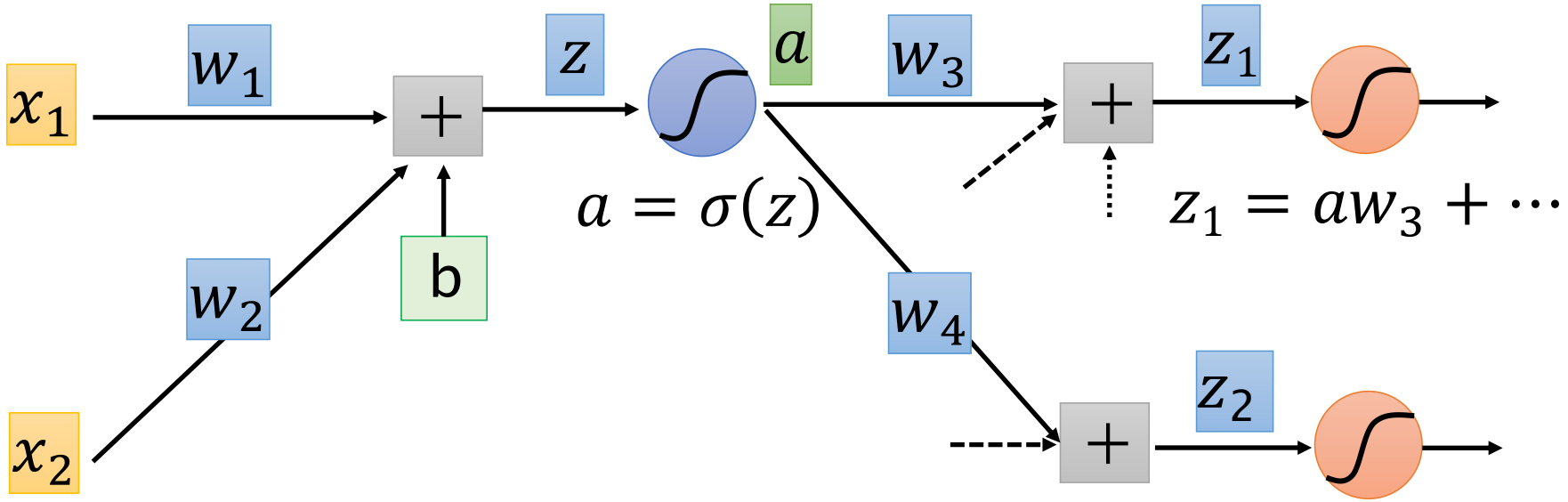
- Sigmoid function: $h(a) = \frac{1}{1+e^{-a}}$ $\quad h'(a) = h(a)(1 - h(a))$
- Softmax function: $h(\mathbf{a}) = \frac{e^{\mathbf{a}_k}}{\sum_{k=1}^{K} e^{\mathbf{a}_k}}$

$\frac{\partial h(\mathbf{a}_i)}{\partial \mathbf{a}_j} = h(\mathbf{a}_i)(\delta_{ij} - h(\mathbf{a}_j))$, where $\delta_{ij}$ is called Kronecker $\delta$

- TanH: $h(a) = \frac{e^{a}-e^{-a}}{e^{a}+e^{-a}}$ $\quad h'(a) = 1 - h^2(a)$

# Backpropagation – Backward pass

Compute $\partial a/\partial z$ for all *activation function* inputs z



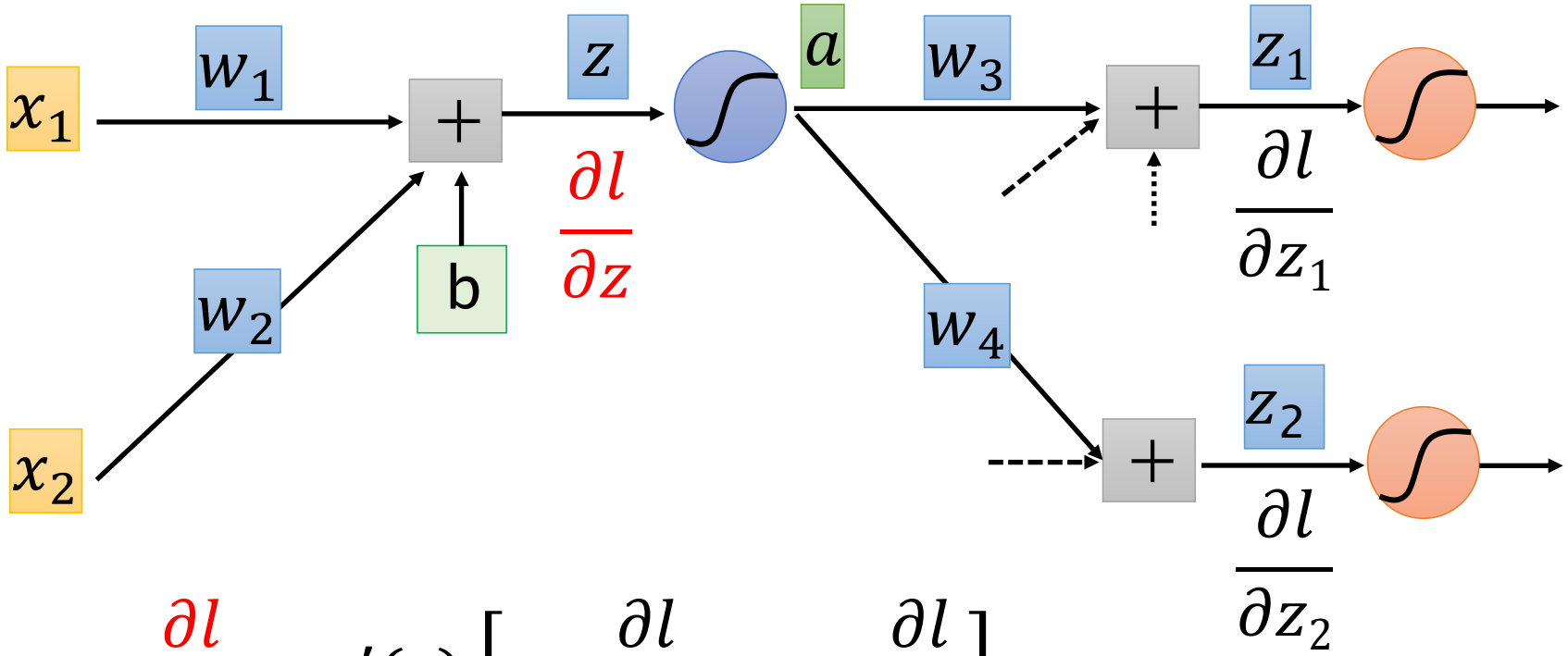

$$\frac{\partial l}{\partial z} = \frac{\partial l}{\partial a}\frac{\partial a}{\partial z} \qquad \frac{\partial l}{\partial a} = \frac{\partial l}{\partial z_1}\frac{\partial z_1}{\partial a} + \frac{\partial l}{\partial z_2}\frac{\partial z_2}{\partial a}$$ (Chain rule)

$\sigma'(z)$ ? $w_3$ ? $w_4$

Assumed it's known

# Backpropagation – Backward pass

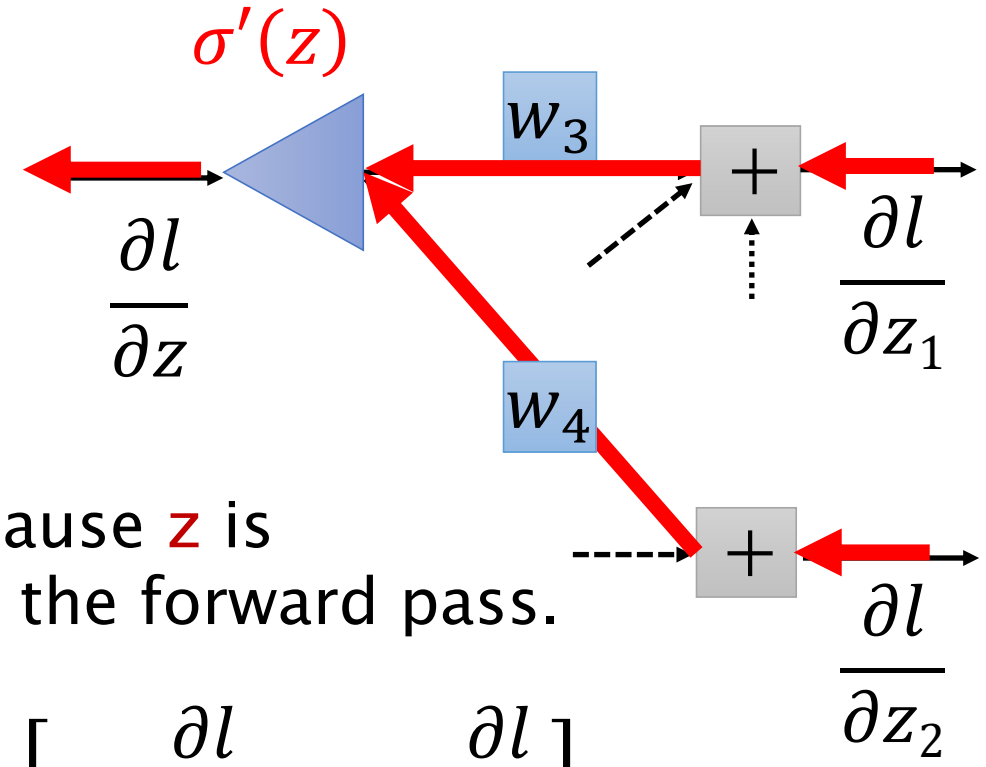
Compute $\partial l/\partial z$ for all *activation function* inputs z




$$\frac{\partial l}{\partial z} = \sigma'(z)\left[w_3 \frac{\partial l}{\partial z_1} + w_4 \frac{\partial l}{\partial z_2}\right]$$

# Backpropagation – Backward pass

Compute $\partial l/\partial z$ for all *activation function* inputs z

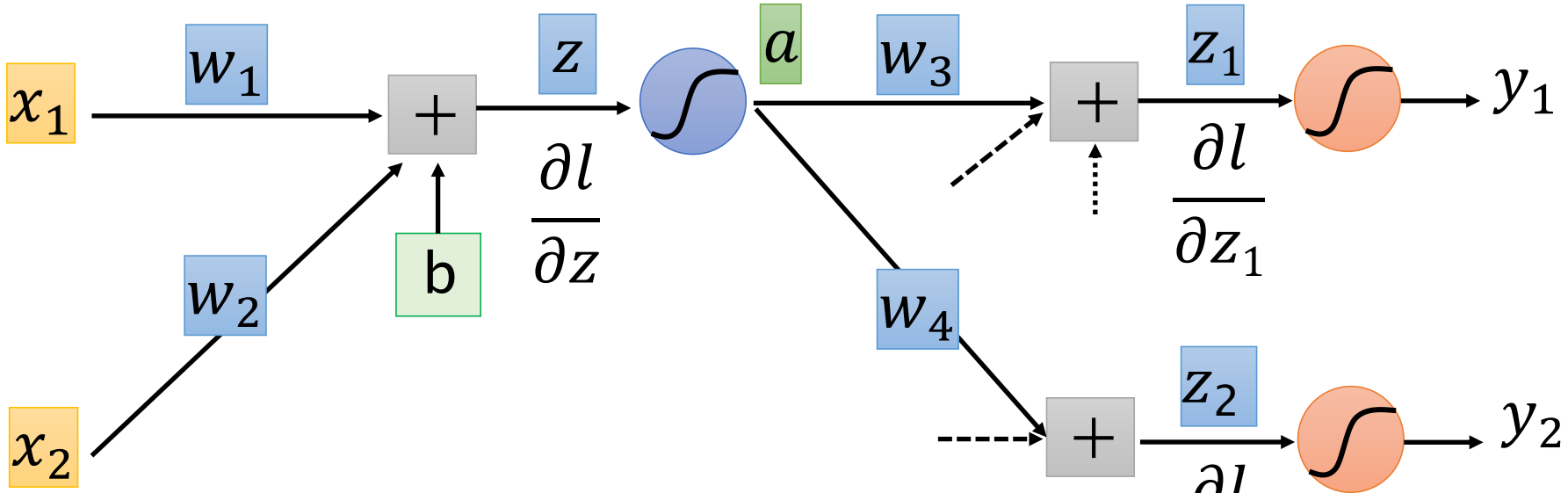



$\sigma'(z)$ is a constant because z is already determined in the forward pass.

$$\frac{\partial l}{\partial z} = \sigma'(z)\left[w_3 \frac{\partial l}{\partial z_1} + w_4 \frac{\partial l}{\partial z_2}\right]$$

# Backpropagation – Backward pass

Compute $\partial l/\partial z$ for all *activation function* inputs z




***Case 1. Output Layer***

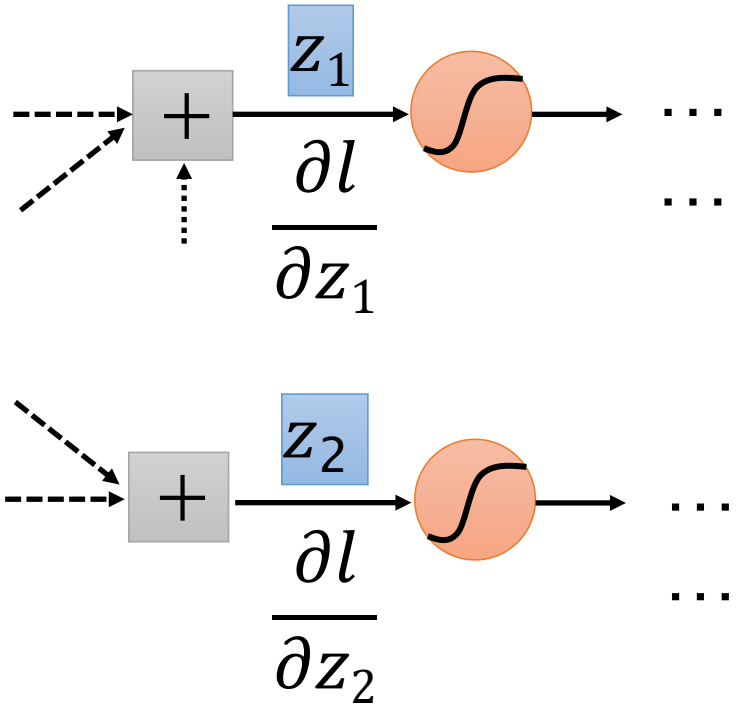
$$\frac{\partial l}{\partial z_1} = \frac{\partial l}{\partial y_1}\frac{\partial y_1}{\partial z_1} \qquad \frac{\partial l}{\partial z_2} = \frac{\partial l}{\partial y_2}\frac{\partial y_2}{\partial z_2}$$

Done!

# Backpropagation – Backward pass

Compute $\partial l/\partial z$ for all *activation function* inputs z

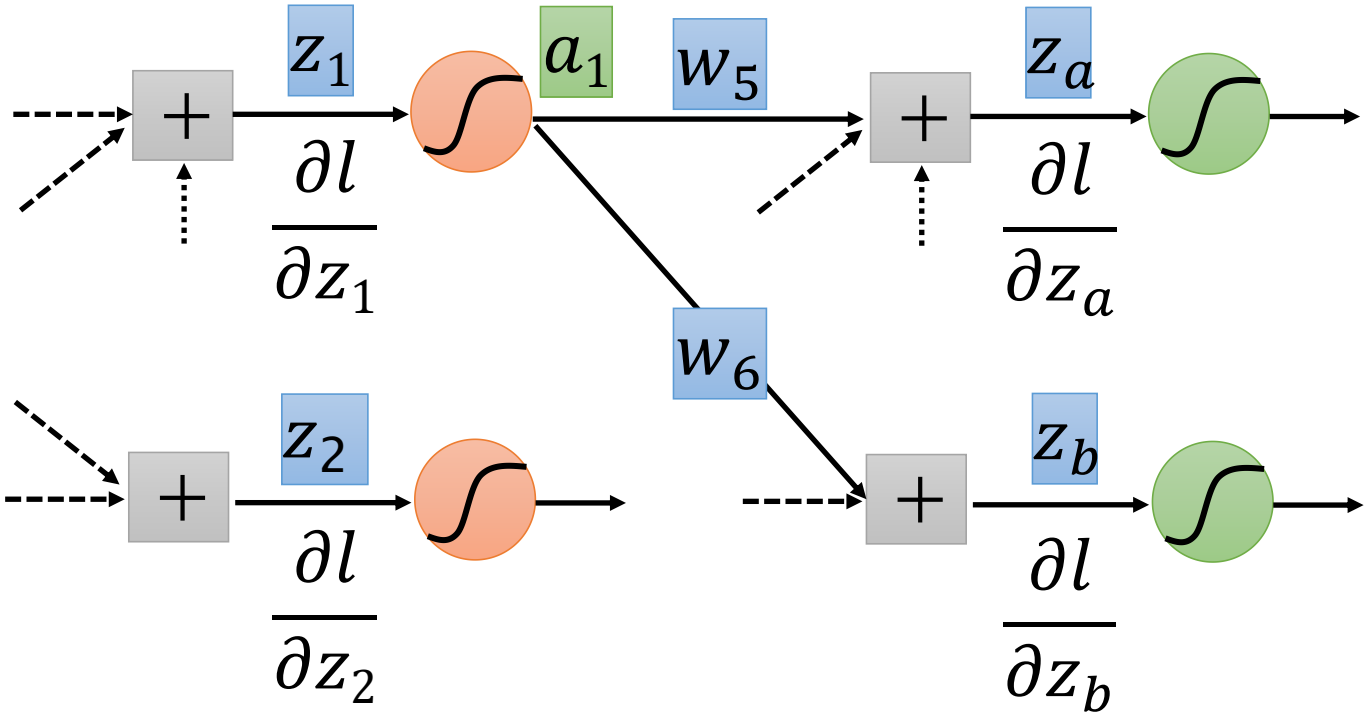
***Case 2. Not Output Layer***



# Backpropagation – Backward pass

Compute $\partial\sigma/\partial z$ for all *activation function* inputs z

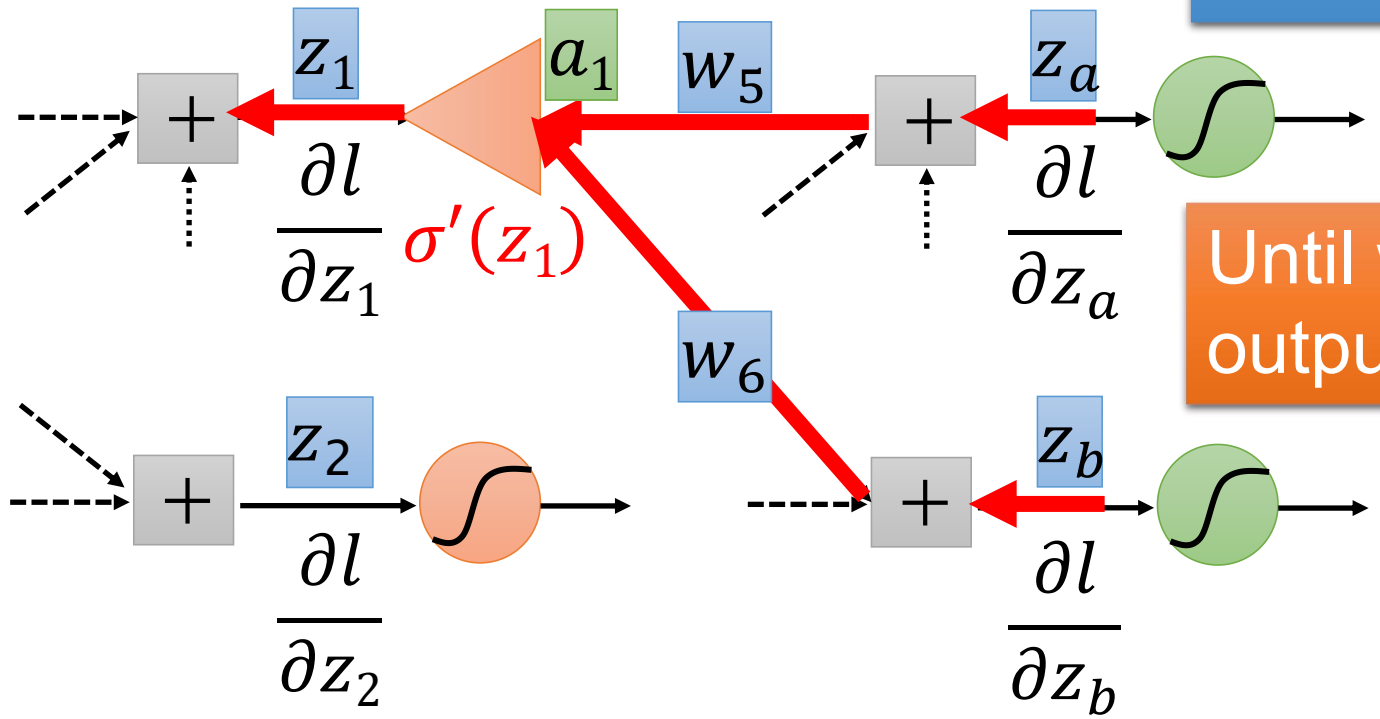
***Case 2. Not Output Layer***



# Backpropagation – Backward pass

Compute $\partial l/\partial z$ for all *activation function* inputs z

***Case 2. Not Output Layer***




Compute $\partial l/\partial z$ recursively

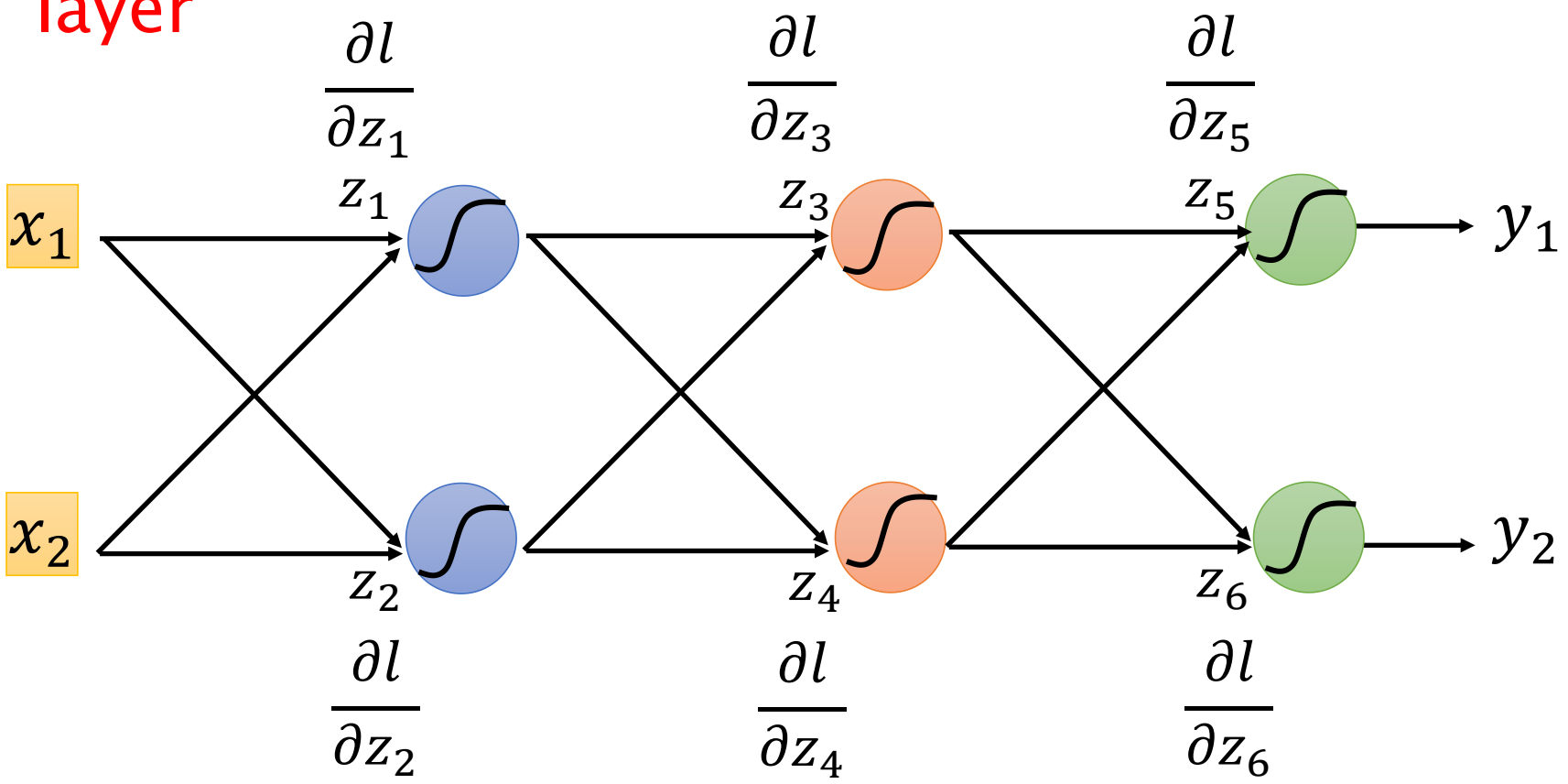
Until we reach the output layer ……

# Backpropagation – Backward pass

Compute $\partial l/\partial z$ for all *activation function* inputs z

Compute $\partial l/\partial z$ from the output layer

$$\frac{\partial l}{\partial z} = \sigma'(z)\left[w_3 \frac{\partial l}{\partial z_1} + w_4 \frac{\partial l}{\partial z_2}\right]$$

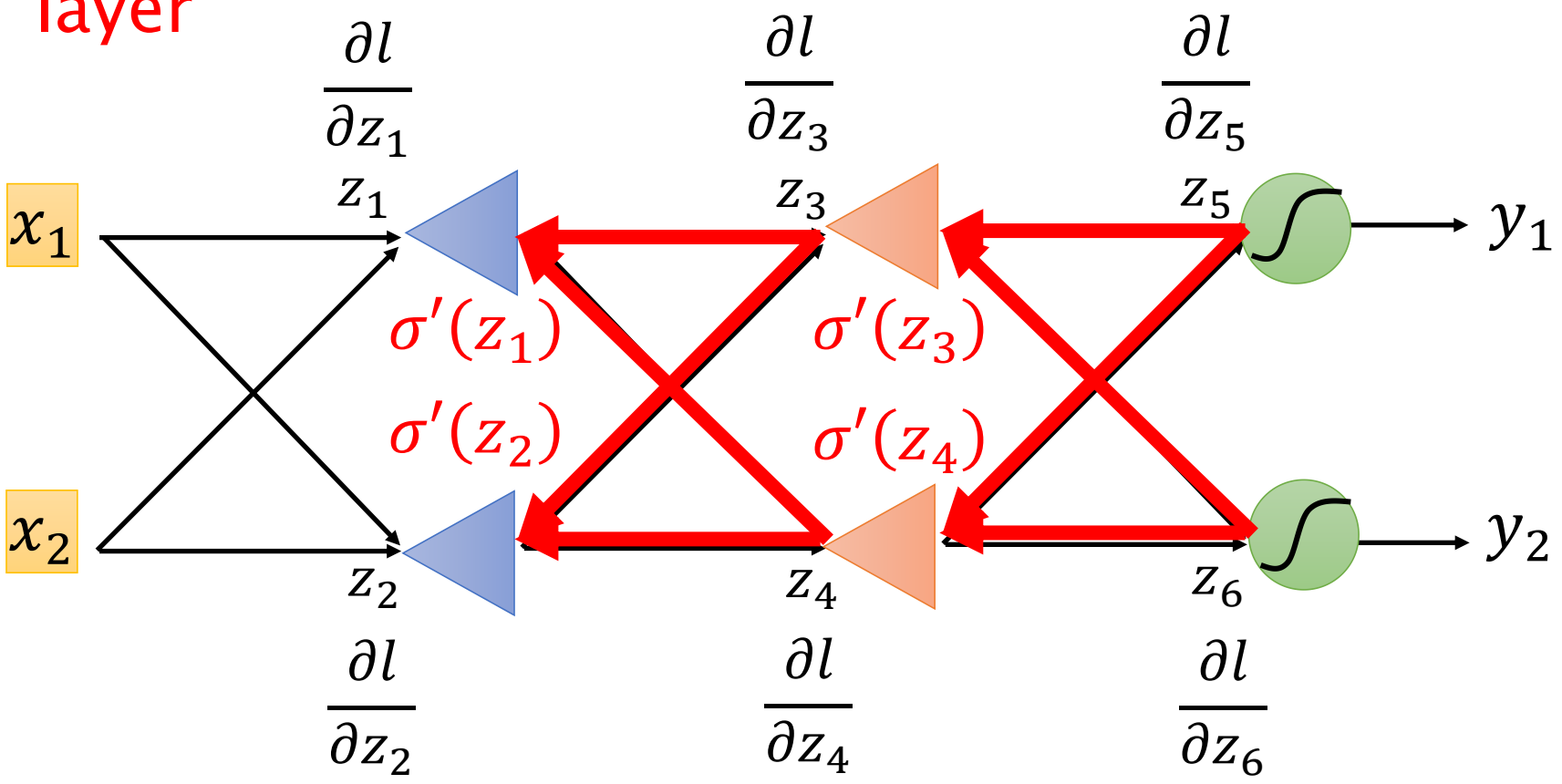


# Backpropagation – Backward pass

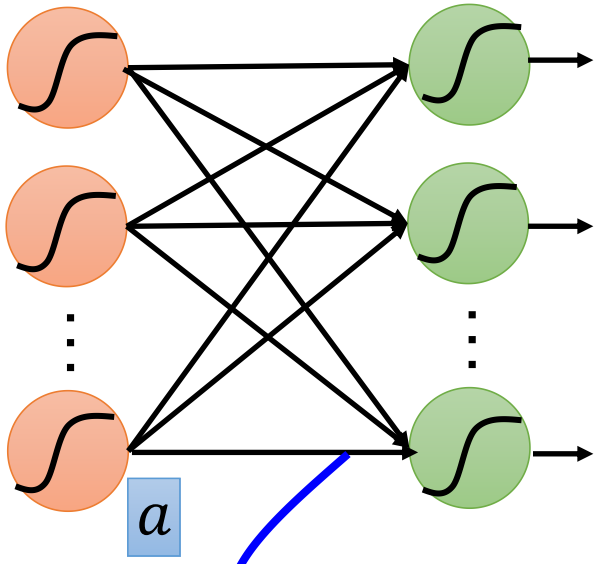
Compute $\partial l/\partial z$ for all *activation function* inputs z

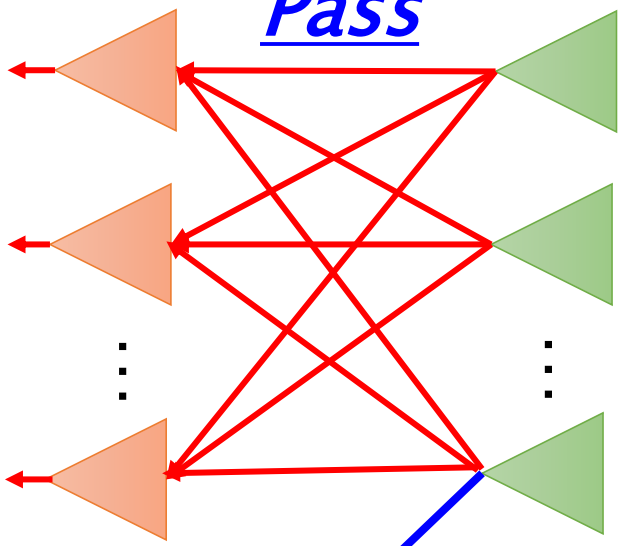
Compute $\partial l/\partial z$ from the output layer

$$\frac{\partial l}{\partial z} = \sigma'(z)\left[w_3 \frac{\partial l}{\partial z_1} + w_4 \frac{\partial l}{\partial z_2}\right]$$



# Backpropagation – Overview




$$\frac{\partial l}{\partial z_{i,j}} = \sigma'(z_{i,j})\left[\sum w_{i,j+1}\frac{\partial l}{\partial z_{i,j+1}}\right]$$

$$\frac{\partial l}{\partial z_{1,n}} = \frac{\partial l}{\partial y_{1,n}}\frac{\partial y_{1,n}}{\partial z_{1,n}}$$

- Initiate parameters all $w^0$
- Go forward pass to get all $a^0$
- $\frac{\partial z}{\partial w_{i,j}^0} = a_{i,j}^0$

$$\frac{\partial z}{\partial w} = a$$

$$\times \quad \frac{\partial l}{\partial z}$$

$$= \frac{\partial l}{\partial w}$$

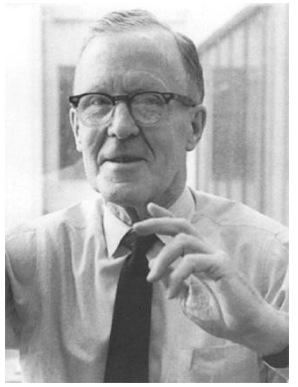
for all w

# Backpropagation (BP) in the Brain?

- There is no direct evidence that the brain uses a backprop-like algorithm for learning. → Most of researchers think BP in brain is biologically implausible.
- **Difficulties** in implementing BP in the Brain:
  1. Backprop demands synaptic symmetry (the same weights) in the forward and backward paths → how to design synaptic connections in forward and backward neural circuits?
  2. Error signals are signed and potentially extreme-valued. → how to convey signed and extreme-valued errors in real neuron spikes?
  3. Feedback in brains alters neural activity. → In NNs, feedback delivers error signals that do not influence the activity states of neurons produced by feedforward propagation. But it does change neural activity in the brain.

Timothy, Santoro, Marris, Akerman, Hinton (2020) Backpropagation and the brain, *Nature*

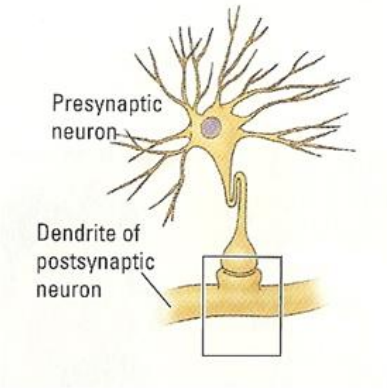
# Learning in the brain (Hebbian learning)



D. O. Hebb

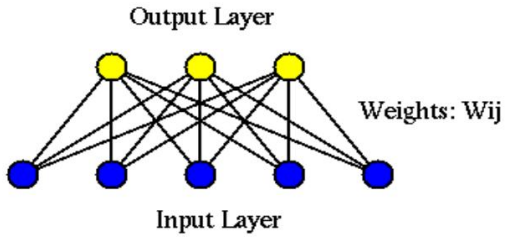
"When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased."

D. O. Hebb, *Organization of Behavior*, 1949



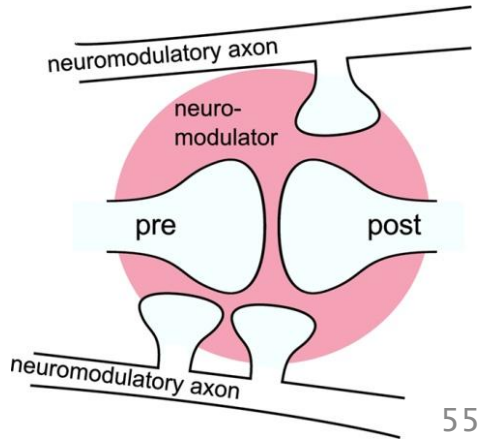

In other words: "**Cells that fire together wire together**."

Mathematically, this is often written as: $\Delta w_{ij} = \varepsilon x_i x_j$



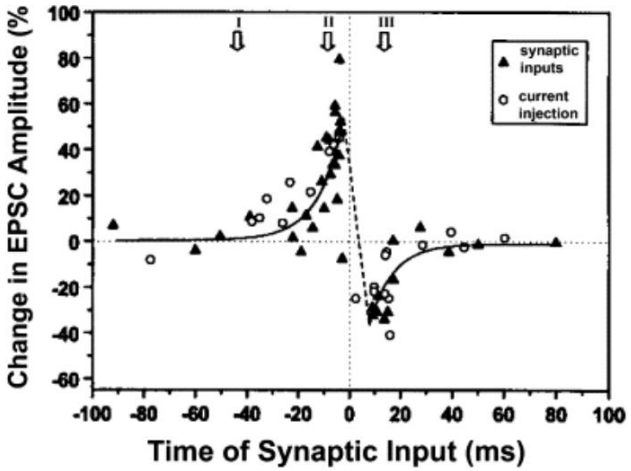

More complex and sophisticated ideas have been under continual exploration for over a half a century, including:

- Reward-modulated learning (reinforcement learning)
- Competitive learning
- Error correcting learning
- Spike-time dependent plasticity



# 大脑中的前馈学习 及 神经递质调控学习规则

双向STDP学习规则 （依赖放电时刻的突触可塑性）



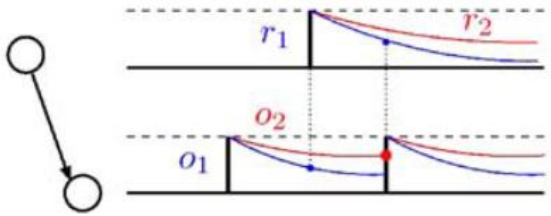

*Bi and Poo 1998 J. Neurosci.*

$$\tau_- do_1/dt = -o_1$$
$$\tau_y do_2/dt = -o_2$$
$$\tau_+ dp_1/dt = -p_1$$

$t = t^{post}$时：
$$p_1 = p_1 + 1$$
$$\Delta w_{LTP} = p_1 (t - \epsilon) A_3^+$$

$t = t^{pre}$时:
$$o_1 = o_1 + 1$$
$$\Delta w_{LTD} = -o_1 A_2^-$$
$$o_2 = o_2 + 1$$

**三向STDP：**脉冲神经元模型




**BCM学习率：**放电率模型

$$dw/dt = A_3^+ \tau_+ \tau_y \, r_{\text{pre}} r \left( r - \frac{\bar{r}(t)}{r_0} \bar{r}(t) \right)$$

**放电率平衡：**

$$A_2^-(t) = \frac{\tau_+ \tau_y \, \bar{r}(t)^2}{\tau_- r_0} A_3^+$$

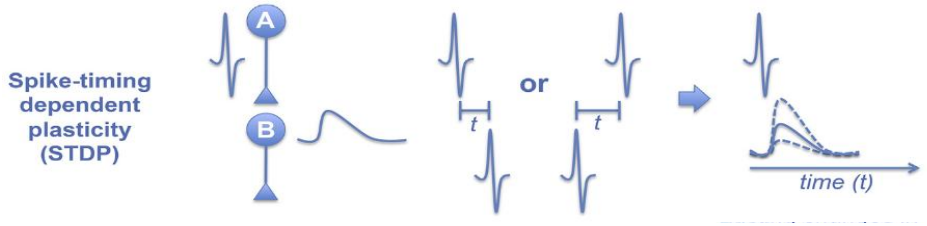
*三向STDP与BCM放电率学习法则的对应关系*
*Zenke et. al., 2013 PloS Comp. Biol.;*
*Pfister and Gerstner 2006 J. Neurosci.*

**神经递质与任务反馈和大脑状态息息相关，携带全局信用信息。**

**加入神经调质可解决基于前馈学习（STDP 或 BCM）的“信用分配”在时空上的局域性问题，以及任务相关性问题。**

乙酰胆碱，去甲肾上腺素，多巴胺，血清素等神经调质的受体的激活和抑制可以调控：

- 学习强度，开关（调整$A_3^+$，$A_2^-$ 即 $r_0$）
- 学习时间窗口大小（调整 $\tau_-$ 和 $\tau_y, \tau_+$）
- 脉冲时刻依赖关系（$t_{pre}$, $t_{post}$和 LTD, LTP的关系）




同时，这些神经调质的**浓度**和**释放时间**与$t_{pre}$, $t_{post}$ 可以决定调质的具体调控方向。
因此，为解决局部信用分配问题，**需建立前馈学习中相关参数受神经调质调控的动力学模型**

*神经调质如何调控STDP*
*Brzosko et. al., 2019 Neuron*

# 大脑与反向传播(Backpropagation and the brain)--上篇

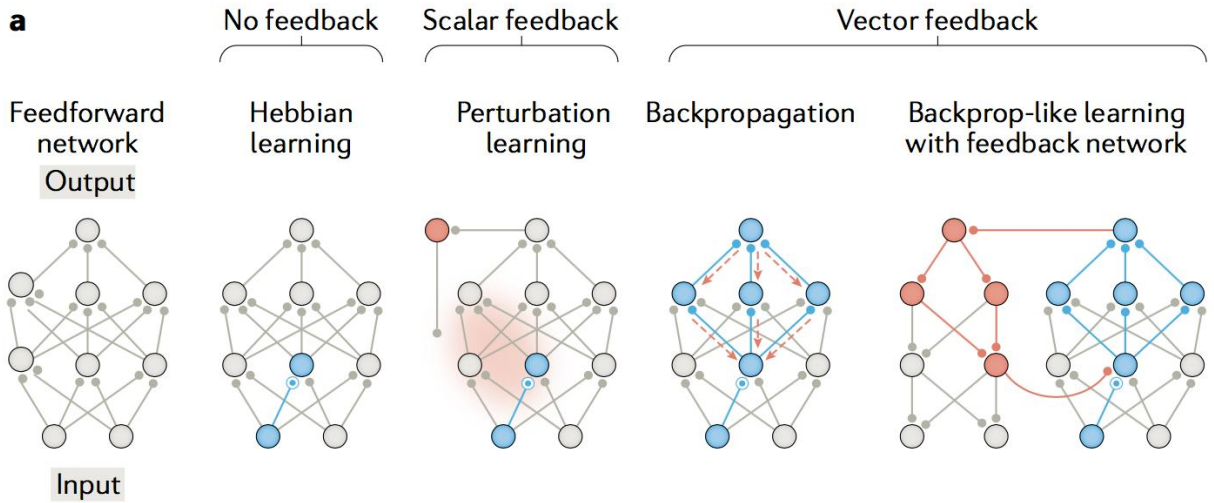
Original NCC lab 神经计算与控制实验室 2020-05-11 22:37

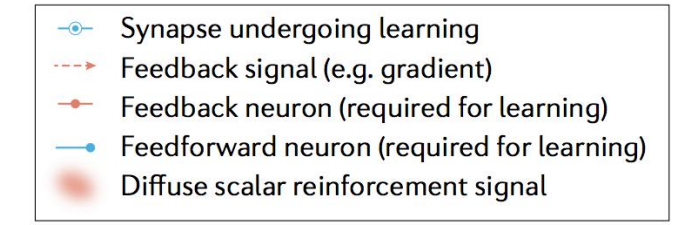
Timothy, Santoro, Marris, Akerman, Hinton (2020) Backpropagation and the brain, *Nature*

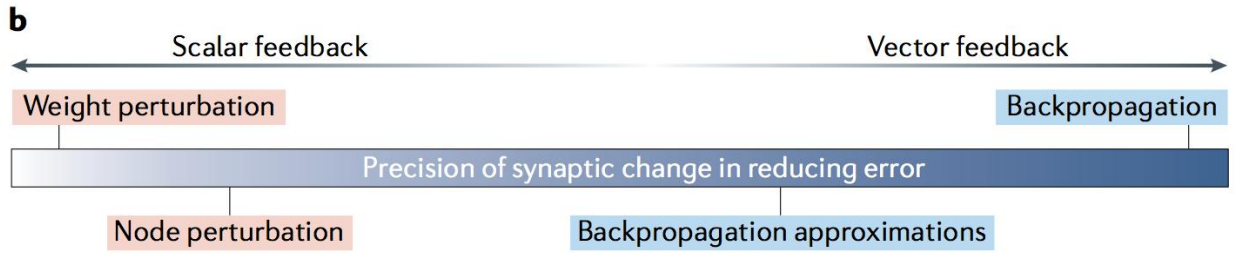
近日 Hinton 等人的研究认为，尽管人脑和神经网络存在明显的物理差异，但大脑仍具有执行反向传播的核心原理的能力。我们对其原文进行了完整的翻译。

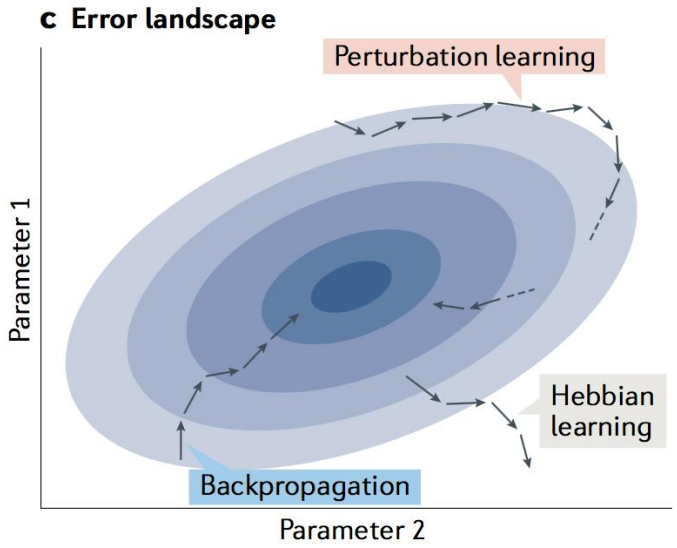
Scan to Follow

https://mp.weixin.qq.com/s/gDfDYWGEDK9uxkABuAQ6jg



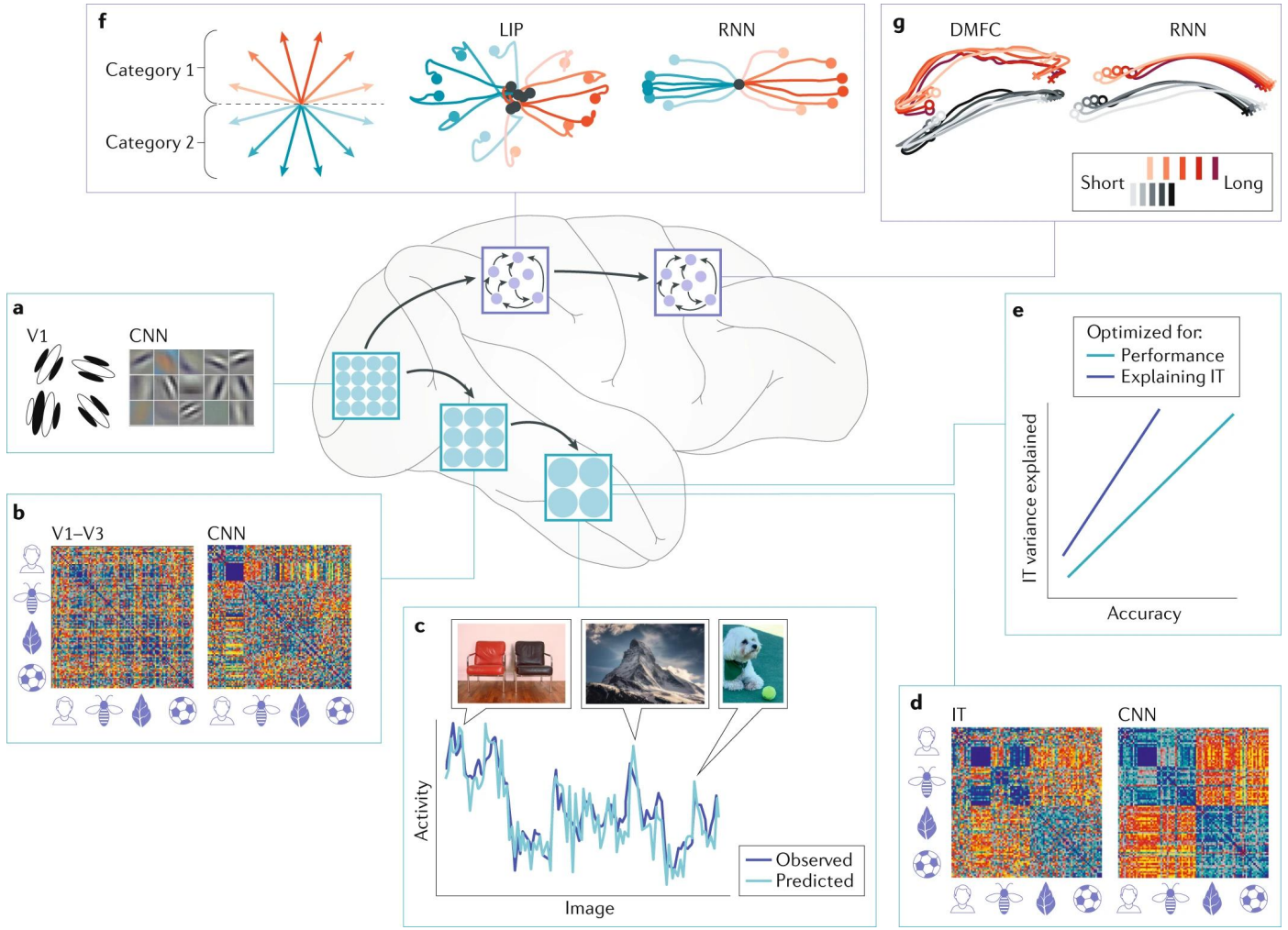



# AI与BI的相似与差异

- ✓ **组成单元相似**：都有神经元
- ✓ **网络结构相似**：都有层级结构
- ✓ **信息表征相似**：人工神经元活动与大脑神经元活动相关
- ✓ **功能实现相似**：能完成同样的任务，例如物体识别、语义理解、工作记忆等等

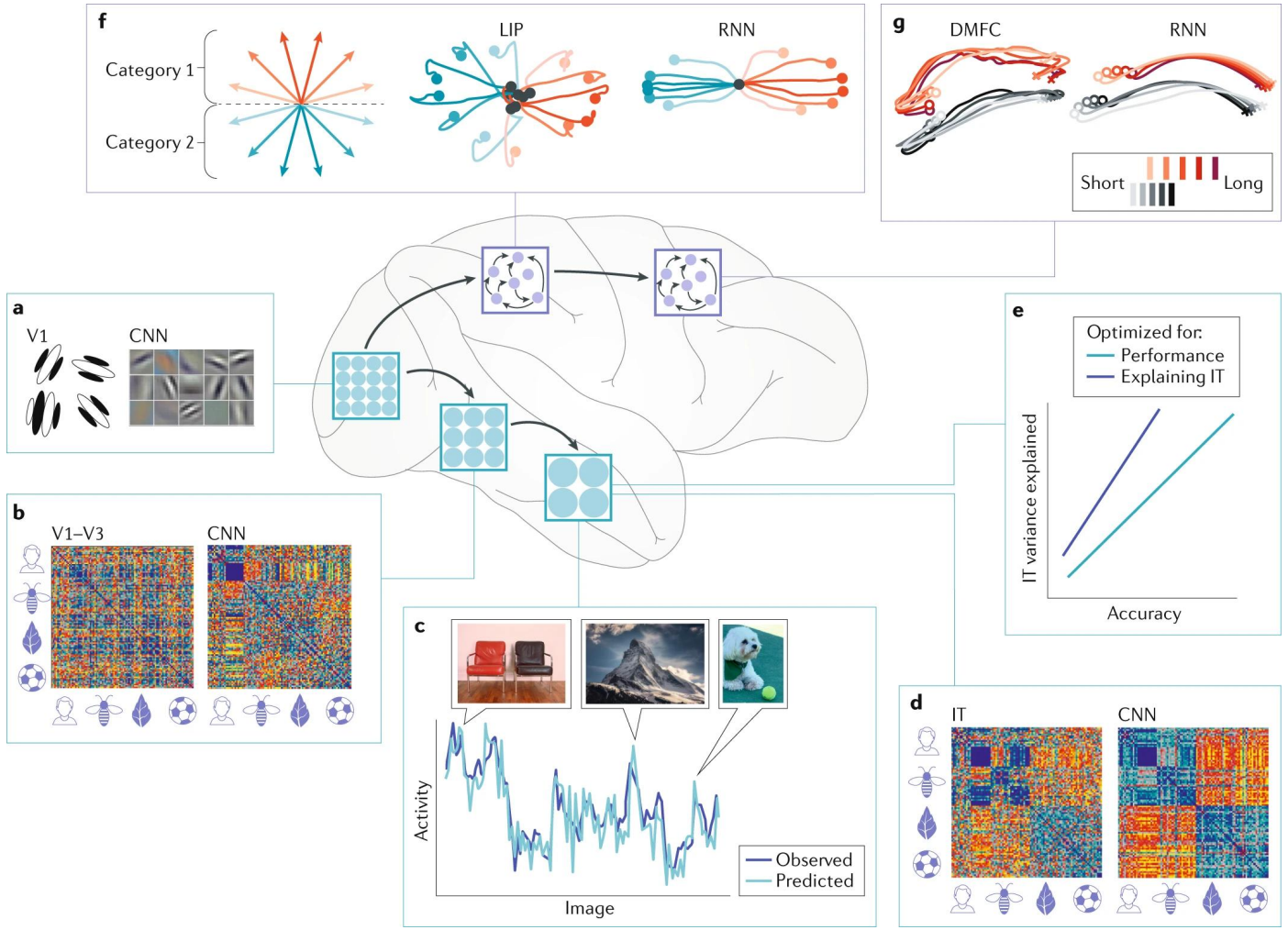



Saxe, A., Nelli, S., & Summerfield, C. (2021). If deep learning is the answer, what is the question?. *Nature Reviews Neuroscience*

# AI与BI的相似与差异

- ✓ **组成单元相似**：都有神经元
- ✓ **网络结构相似**：都有层级结构
- ✓ **信息表征相似**：人工神经元活动与大脑神经元活动相关
- ✓ **功能实现相似**：能完成同样的任务，例如物体识别、语义理解、工作记忆等等

- ❖ **学习方式**：局部反馈 vs 全局误差
- ❖ **所需数据量**：少量 vs 海量
- ❖ **能耗**：20瓦 vs 1287兆瓦(单次训练GPT3)
- ❖ **能力**：因果推理、知识泛化




Saxe, A., Nelli, S., & Summerfield, C. (2021). If deep learning is the answer, what is the question?. *Nature Reviews Neuroscience*