CS688: Web-Scale Image Search
Deep Neural Nets and Features

Sung-Eui Yoon
(윤성의)

Course URL:
http://sgvr.kaist.ac.kr/~sungeui/IR

KAIST

Class Objectives

- Browse main components of deep neural nets
  - Does not aim for giving in-depth knowledge, but for giving a quick review on the topic
  - Look for other materials if you want to know more
  - Remember: this is one of the prerequisite of taking this course

- At the prior class:
  - Automatic scale selection, and LoG/DoG
  - SIFT as a local descriptor

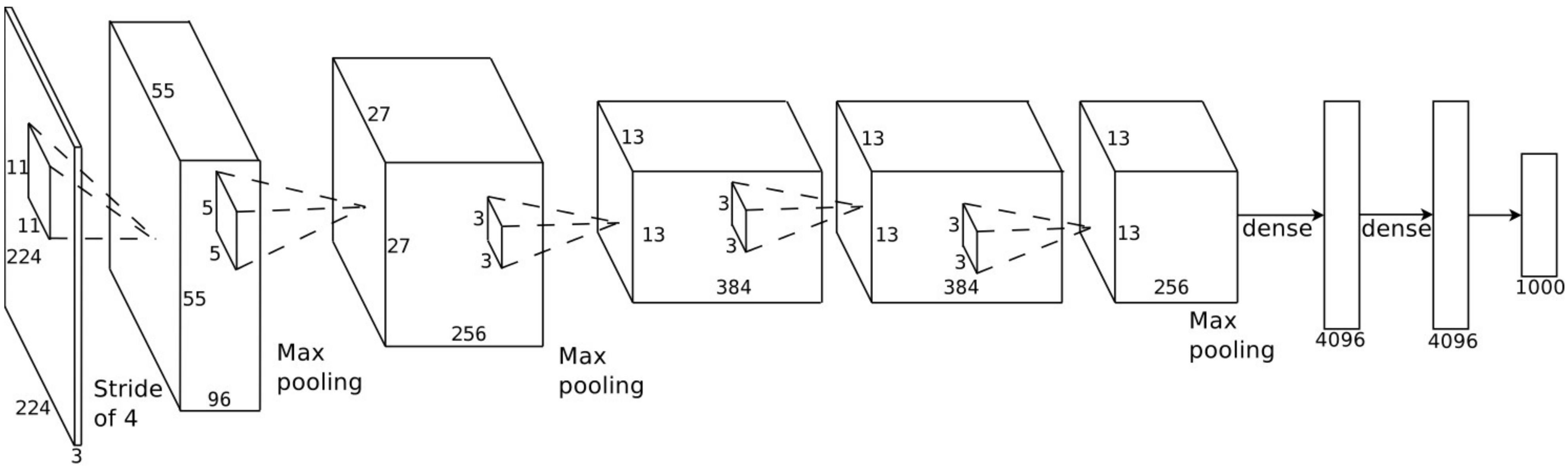
Questions?

- **What are the difference and relationship between CV and IR? Many applications you have talked about in the first class might be normally regarded as Computer Vision applications. I thought IR is a small part of CV before, but after the class I thought it could cover a very large part of CV. How do you think?**

KAIST

High-Level Messages

- **Deep neural nets provide low-level and high-level features**
  - **We can use those features for image search**
- **Achieve the best results in many computer vision related problems**

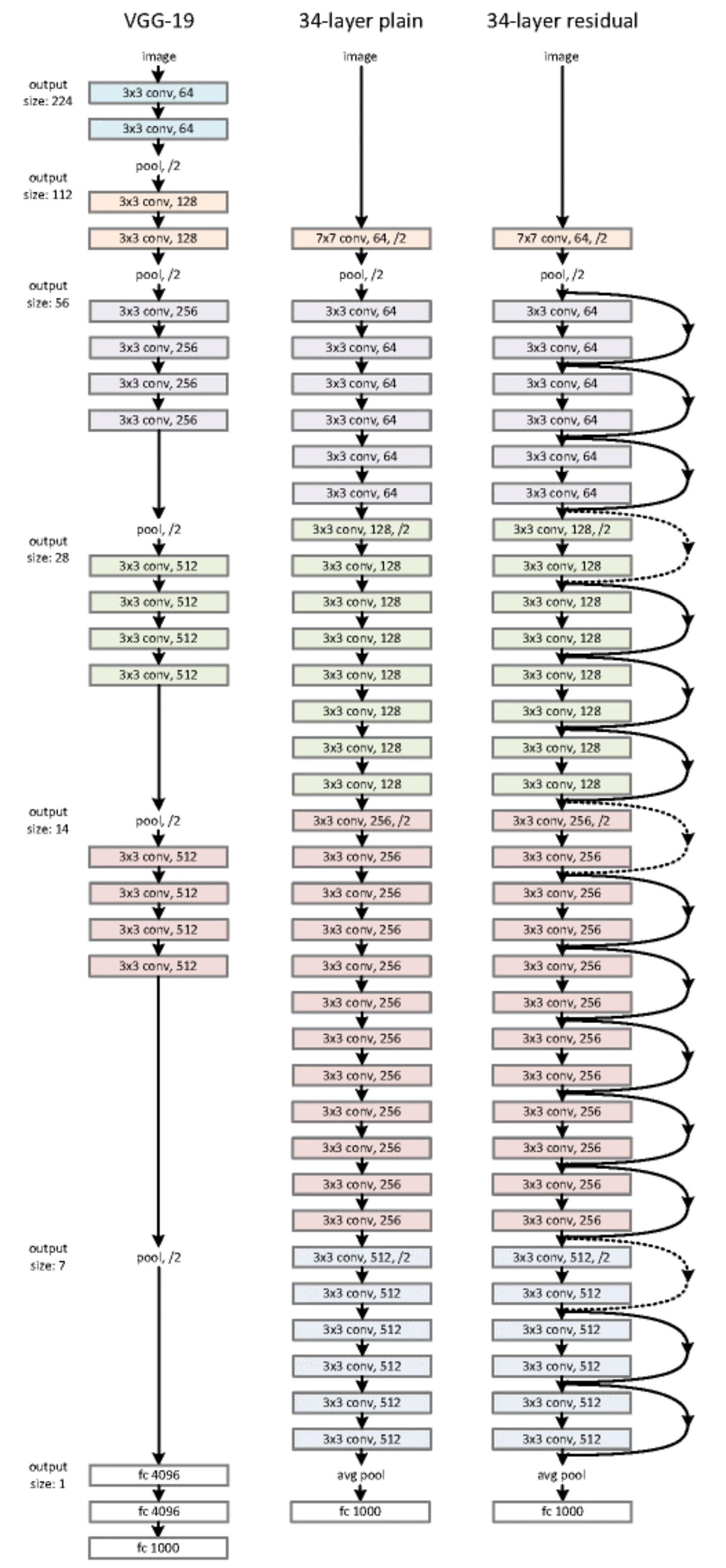


Krizhevsky et al., NIPS 2012

High-Level Messages

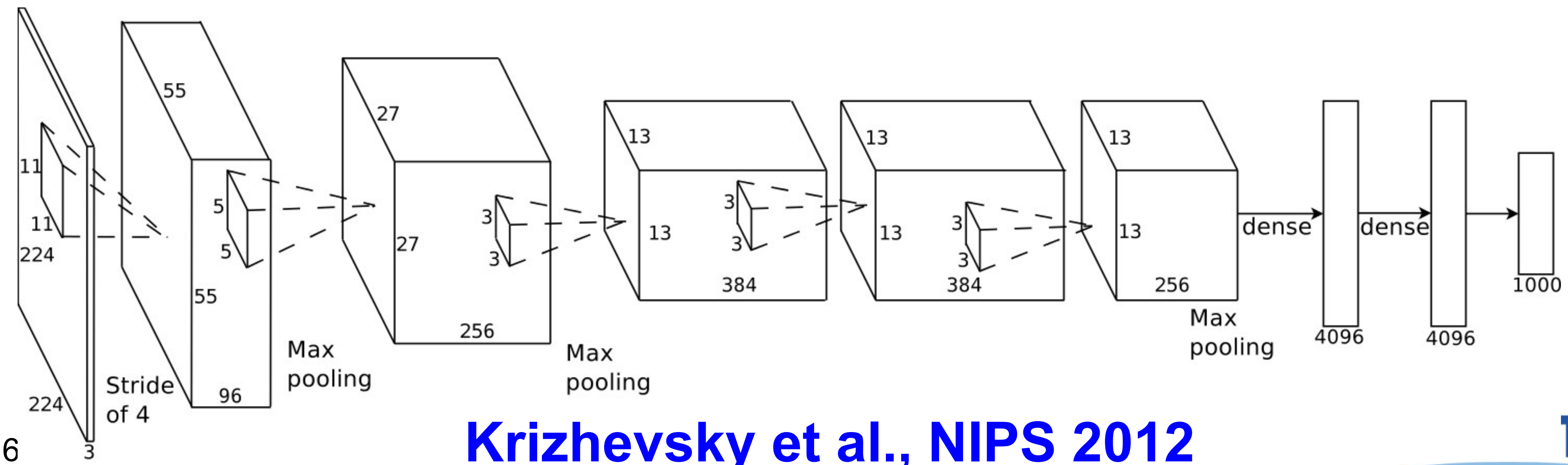
- **Many features and codes are available**
  - **Caffe [Krizhevsky et al., NIPS 2012]**
  - **Very deep convolutional networks [Simonyan et al., ICLR 15]; using up to 19 layers**
  - **Deep Residual Learning [He et al., CVPR 16]; using up to 152 layers**
- **Model Zoo**

  github.com/BVLC/caffe/wiki/Model-Zoo



High-Level Messages

- **Perform the end-to-end optimization w/ lots of training data**
  - **Aims not only features, but the accuracy of any end-to-end systems including image search**
  - **Different from manually created descriptors (e.g., SIFT)**



Krizhevsky et al., NIPS 2012

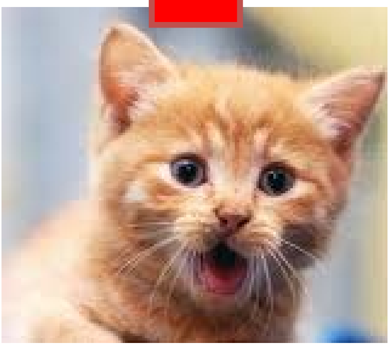
Deep Learning for Vision

Adam Coates
Stanford University
(Visiting Scholar: Indiana University, Bloomington)

What do we want ML to do?

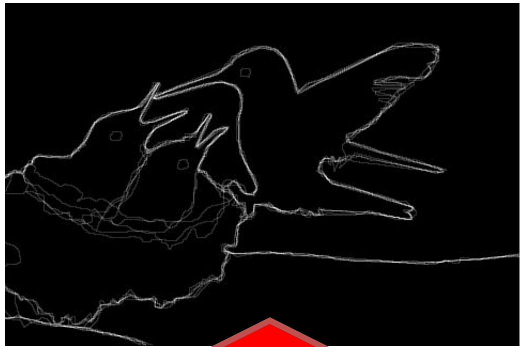
- Given image, predict complex high-level patterns:

"Cat"



Object recognition

Detection

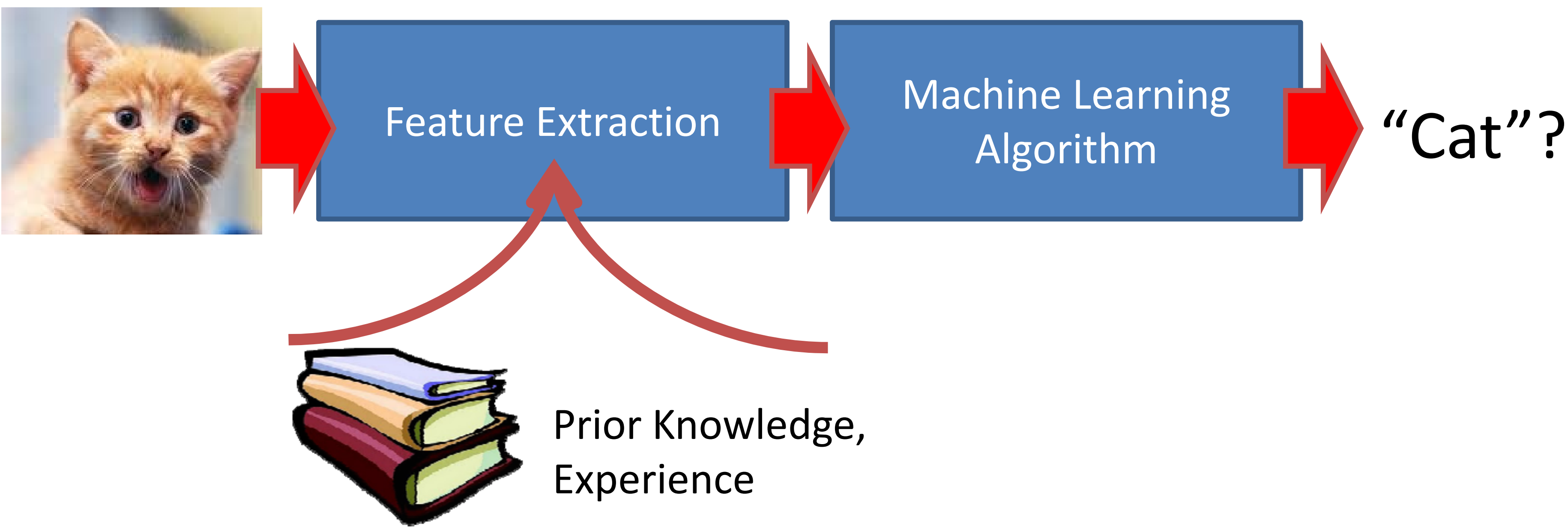


Segmentation

[Martin et al., 2001]

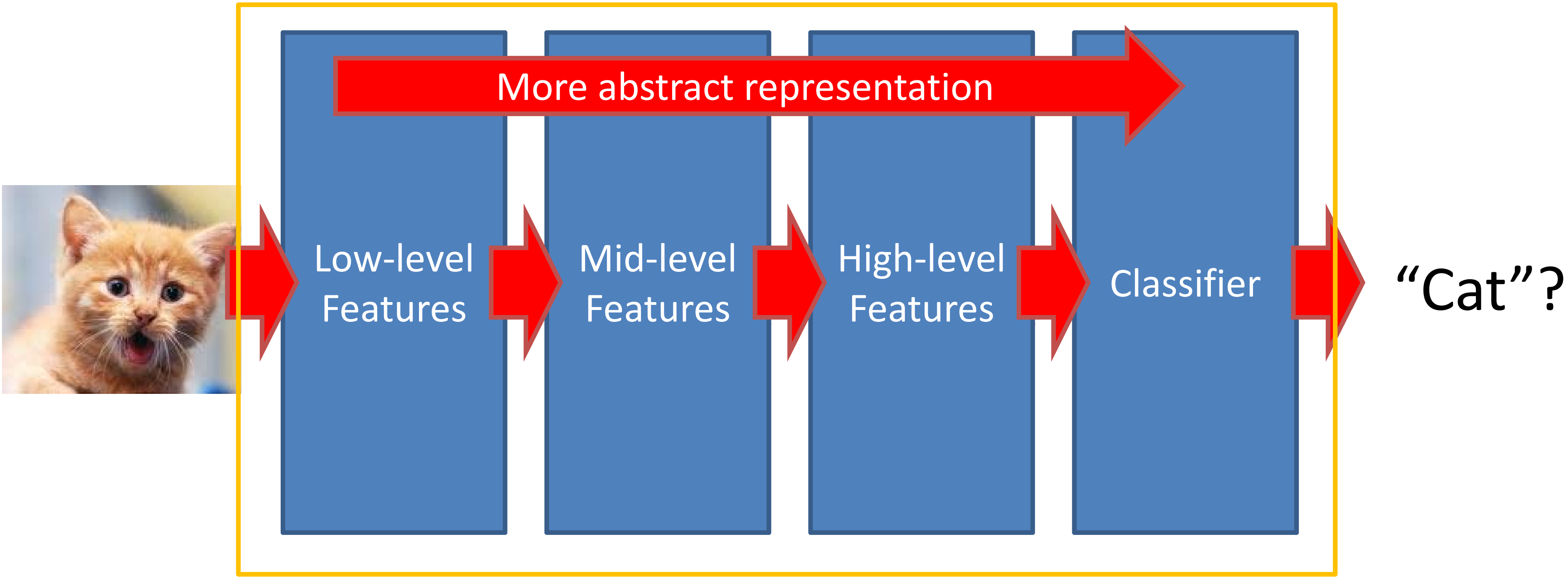
How is ML done?

- Machine learning often uses hand-designed feature extraction.



“Deep Learning”

- Deep Learning
  - Train *multiple layers* of features from data.
  - Try to discover useful *representations*



“Deep Learning”

- Why do we want “deep learning”?
  - Some decisions require many stages of processing.
  - We already hand-engineer “layers” of representation.
  - Algorithms scale well with data and computing power.
    - In practice, one of the most consistently successful ways to get good results in ML.

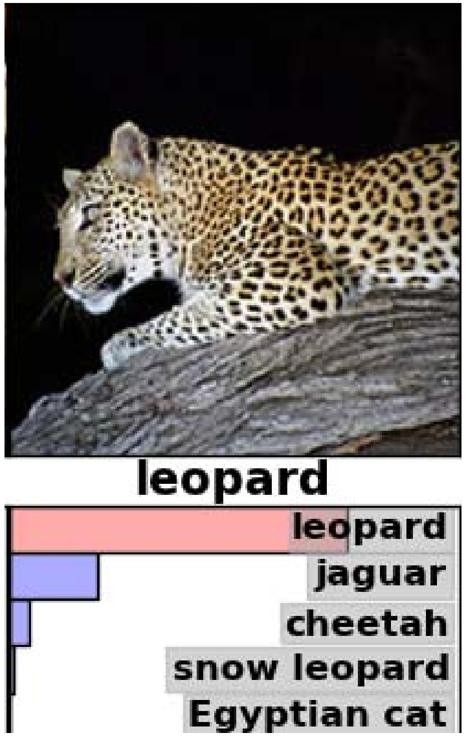
Have we been here before?

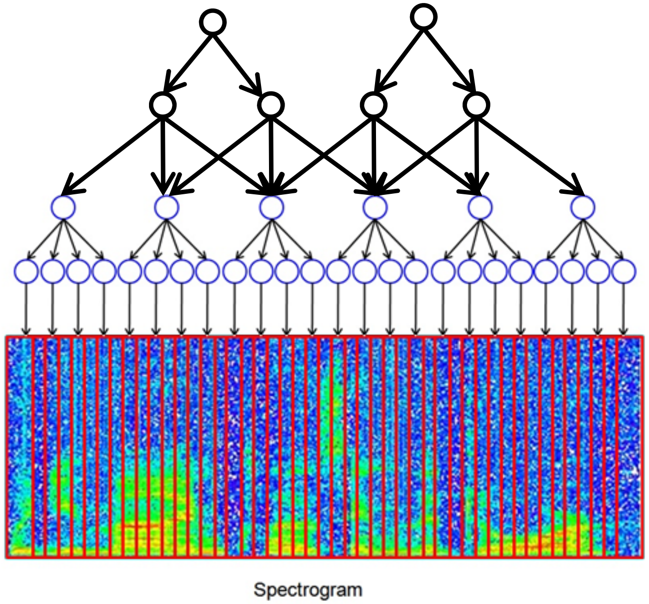
- Yes: Basic ideas common to past ML and neural networks research.

- No.
  - Faster computers; more data.
  - Better optimizers; better initialization schemes.
    - "Unsupervised pre-training" trick [Hinton et al. 2006; Bengio et al. 2006]
  - Lots of empirical evidence about what works.
    - Made useful by ability to "mix and match" components. [See, e.g., Jarrett et al., ICCV 2009]

Real impact

- DL systems are high performers in many tasks over *many domains*.





[Honglak Lee]

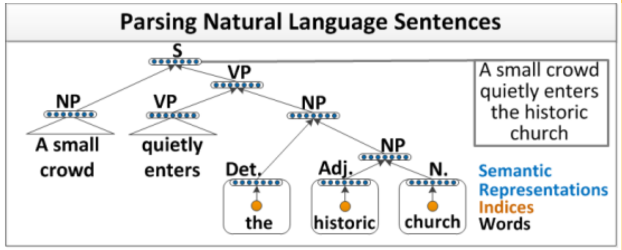


Image recognition
[E.g., Krizhevsky et al., 2012]

Speech recognition
[E.g., Heigold et al., 2013]

NLP
[E.g., Socher et al., ICML 2011; Collobert & Weston, ICML 2008]

Crash Course

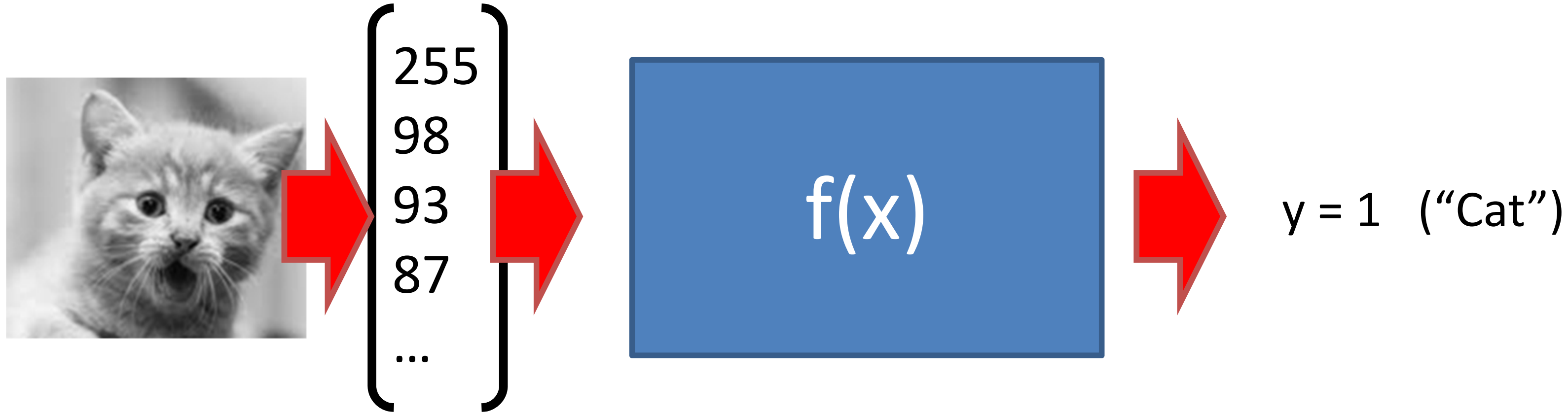
MACHINE LEARNING REFRESHER

Supervised Learning

- Given *labeled* training examples:

$$\mathcal{X} = \{(x^{(i)}, y^{(i)}) : i = 1, \ldots, m\}$$

- For instance: $x^{(i)}$ = vector of pixel intensities.
  $y^{(i)}$ = object class ID.

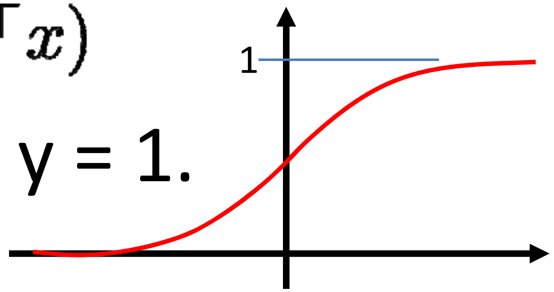


- Goal: find f(x) to predict y from x on training data.
  - Hopefully: learned predictor works on "test" data.

Logistic Regression

- Simple binary classification algorithm
  - Start with a function of the form:

$$f(x;\theta) \equiv \sigma(\theta^\top x) = \frac{1}{1+\exp(-\theta^\top x)}$$



  - Interpretation: f(x) is probability that y = 1.
  - Find choice of θ that minimizes objective:

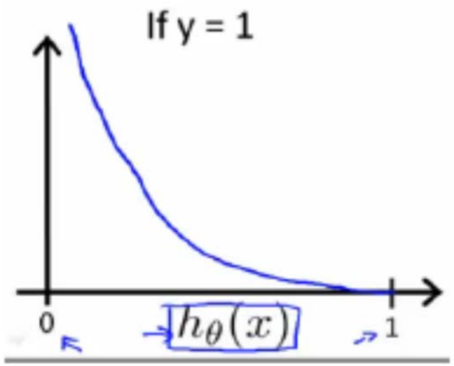
$$\mathcal{L}(\theta) = -\sum_i^m 1\{y^{(i)} = 1\} \log(f(x^{(i)};\theta)) +$$

$$1\{y^{(i)} = 0\} \log(1 - f(x^{(i)};\theta))$$

$\mathbb{P}(y^{(i)} = 1 | x^{(i)})$

$\mathbb{P}(y^{(i)} = 0 | x^{(i)})$

cost



From Ng's slide

Optimization

- How do we tune θ to minimize $\mathcal{L}(\theta)$?
- One algorithm: gradient descent
  - Compute gradient:

$$\nabla_\theta \mathcal{L}(\theta) = \sum_i^m x^{(i)} \cdot (y^{(i)} - f(x^{(i)}; \theta))$$

  - Follow gradient "downhill":

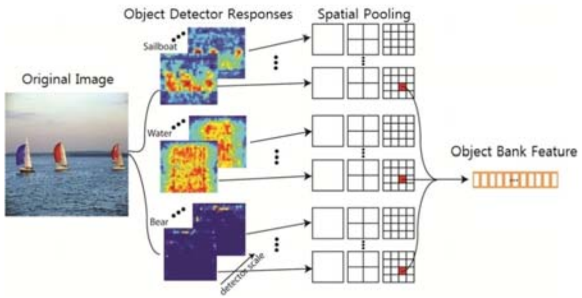
$$\theta := \theta - \eta \nabla_\theta \mathcal{L}(\theta)$$

- Stochastic Gradient Descent (SGD): take step using gradient from only small batch of examples.
  - Scales to larger datasets. [Bottou & LeCun, 2005]

Features

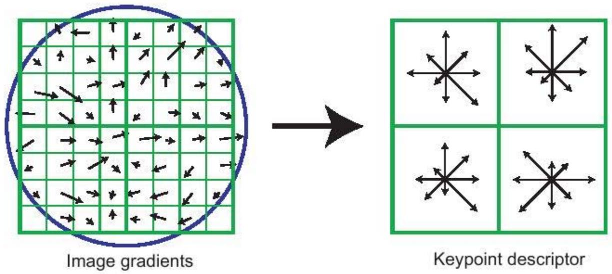
- Huge investment devoted to building application-specific feature representations.

Object Bank [Li et al., 2010]

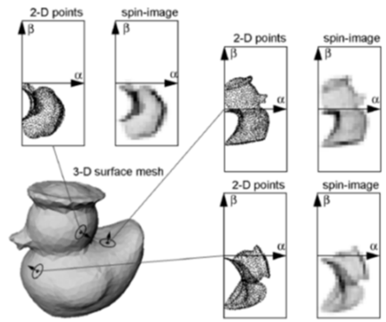


Super-pixels
[Gould et al., 2008; Ren & Malik, 2003]

SIFT [Lowe, 1999]



Spin Images [Johnson & Hebert, 1999]



Extension to neural networks

SUPERVISED DEEP LEARNING

Basic idea

- We saw how to do supervised learning when the "features" ϕ(x) are fixed.
  – Let's extend to case where features are given by tunable functions with their own parameters.

$$\mathbb{P}(y=1|x) = f(x;\theta,W) = \sigma(\theta^\top \underline{\sigma(Wx)})$$

Outer part of function is same as logistic regression.

Inputs are "features"---one feature for each row of W:

$$\begin{bmatrix} \sigma(w_1 x) \\ \sigma(w_2 x) \\ \cdots \\ \sigma(w_K x) \end{bmatrix}$$

Basic idea

- To do supervised learning for two-class classification, minimize:

$$\mathcal{L}(\theta, W) = -\sum_{i}^{m} 1\{y^{(i)} = 1\} \log(f(x^{(i)}; \theta, W)) + 1\{y^{(i)} = 0\} \log(1 - f(x^{(i)}; \theta, W))$$

- Same as logistic regression, but now f(x) has multiple stages ("layers", "modules"):

$$f(x; \theta, W) = \sigma(\theta^\top \sigma(Wx))$$

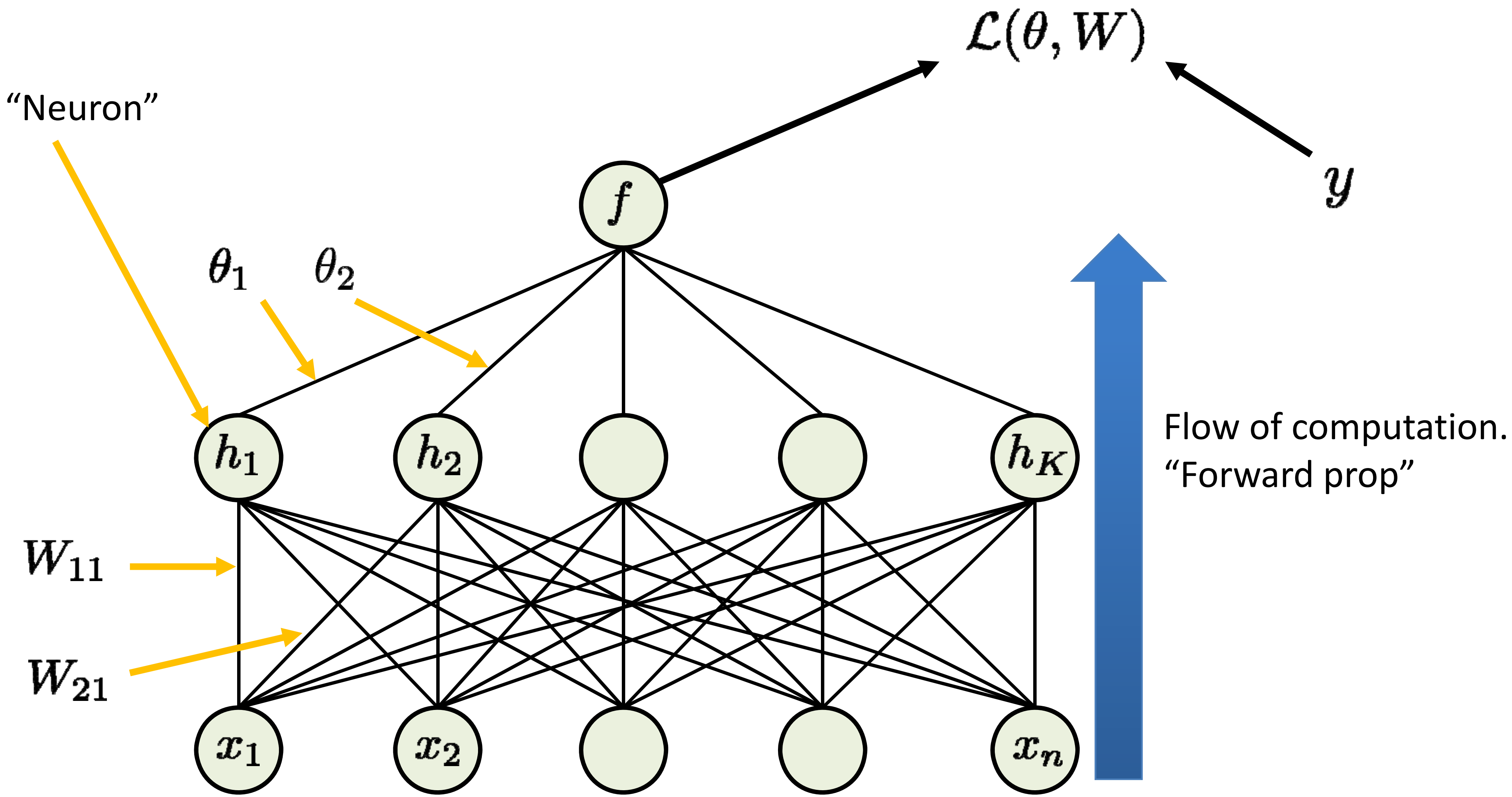
$$x \longrightarrow \boxed{\sigma(Wx)} \longrightarrow h \longrightarrow \boxed{\sigma(\theta^\top h)} \longrightarrow f$$

Intermediate representation ("features")

Prediction for $\mathbb{P}(y = 1|x)$

Neural network

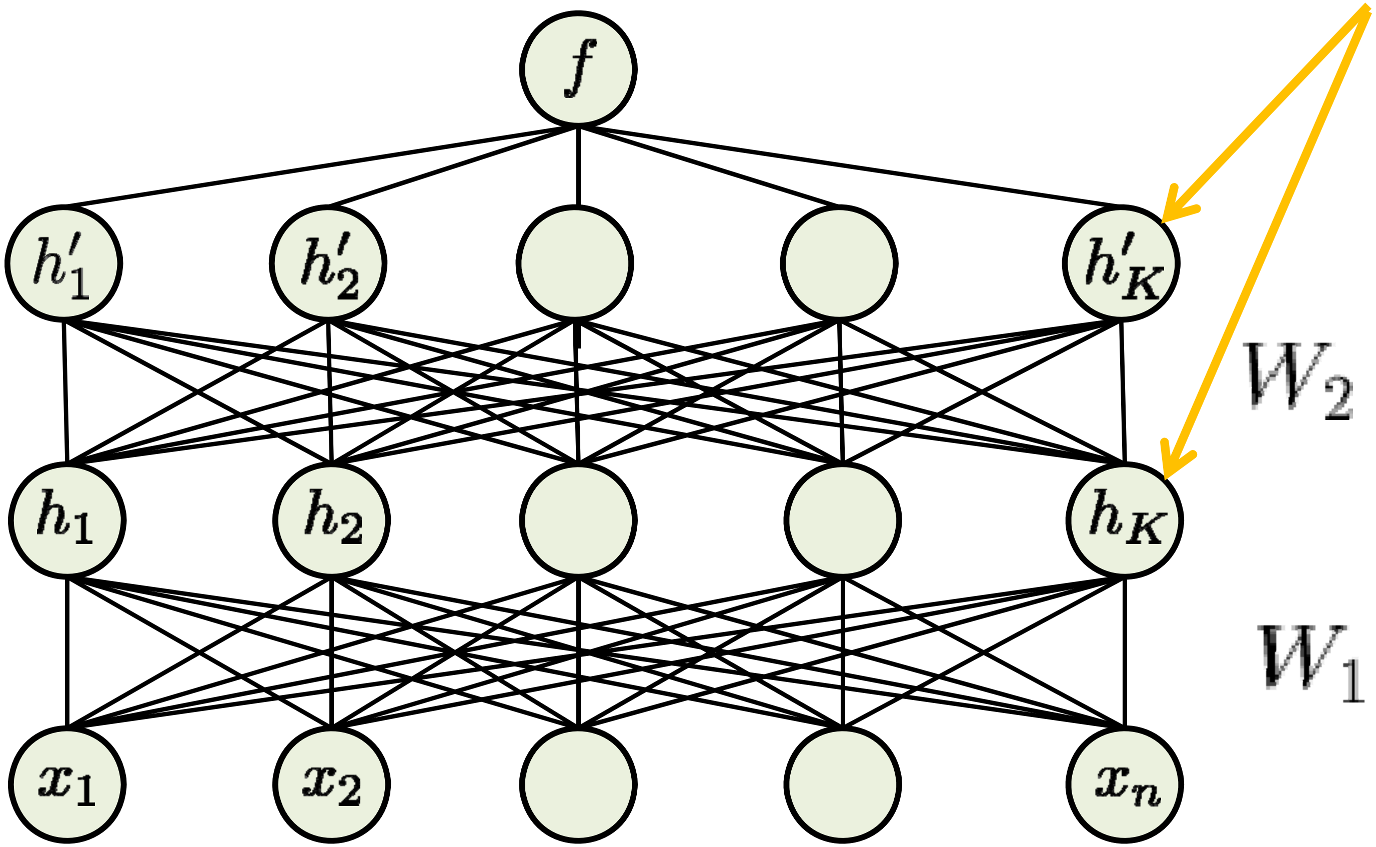
- This model is a sigmoid "neural network":



Neural network

- Can stack up several layers:

Must learn multiple stages of internal "representation".



$$x \longrightarrow \boxed{\sigma(W_1 x)} \longrightarrow h \longrightarrow \boxed{\sigma(W_2 h)} \longrightarrow h' \longrightarrow \boxed{\sigma(\theta^\top h')} \longrightarrow f$$

Back-propagation

- Minimize:

$$\mathcal{L}(\theta, W) = -\sum_{i}^{m} 1\{y^{(i)} = 1\} \log(f(x^{(i)}; \theta, W)) +$$
$$1\{y^{(i)} = 0\} \log(1 - f(x^{(i)}; \theta, W))$$

- To minimize $\mathcal{L}(\theta, W)$ we need gradients:

$$\nabla_{\theta}\mathcal{L}(\theta, W) \text{ and } \nabla_{W}\mathcal{L}(\theta, W)$$

  – Then use gradient descent algorithm as before.

- Formula for $\nabla_{\theta}\mathcal{L}(\theta, W)$ can be found by hand (same as before); but what about W?
  – Beyond the scope of this course

Training Procedure

- Collect labeled training data
  - For SGD: Randomly shuffle after each epoch!

$$\mathcal{X} = \{(x^{(i)}, y^{(i)}) : i = 1, \ldots, m\}$$

- For a batch of examples:
  - Compute gradient w.r.t. all parameters in network.

$$\Delta_\theta := \nabla_\theta \mathcal{L}(\theta, W)$$

$$\Delta_W := \nabla_W \mathcal{L}(\theta, W)$$

  - Make a small update to parameters.

$$\theta := \theta - \eta_\theta \Delta_\theta$$

$$W := W - \eta_W \Delta_W$$

  - Repeat until convergence.

Training Procedure

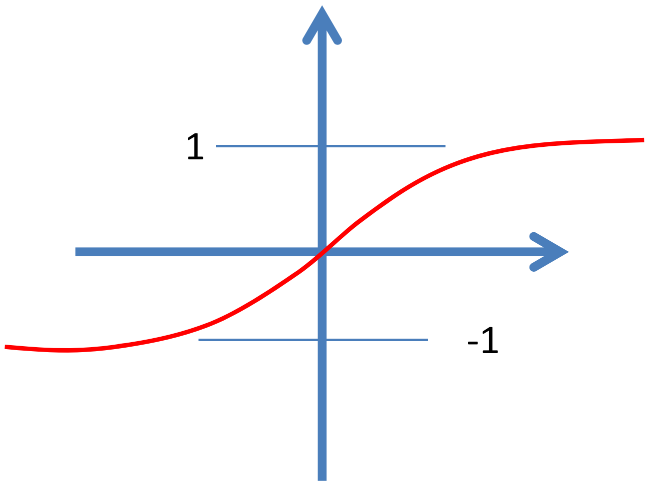
- Historically, this has not worked so easily.
  - Non-convex: Local minima; convergence criteria.
  - Optimization becomes difficult with many stages.
    - "Vanishing gradient problem"
  - Hard to diagnose and debug malfunctions.

- Many things turn out to matter:
  - Choice of nonlinearities.
  - Initialization of parameters.
  - Optimizer parameters: step size, schedule.

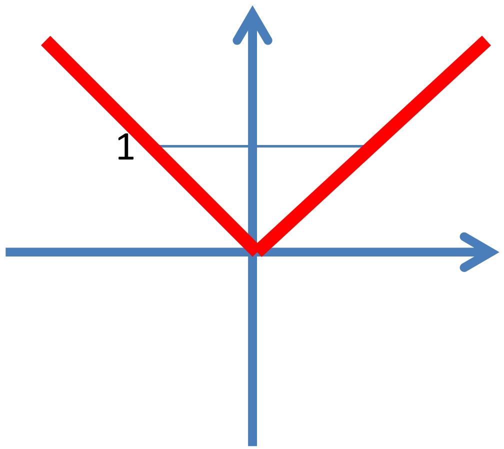
Nonlinearities

- Choice of functions inside network matters.
  - Sigmoid function turns out to be difficult.
  - Some other choices often used:

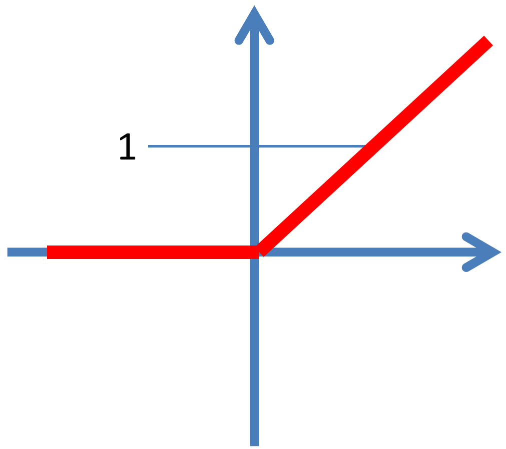
tanh(z)



abs(z)



ReLu(z) = max{0, z}



“Rectified Linear Unit”
→ Increasingly popular.

[Nair & Hinton, 2010]

Summary

- Supervised deep-learning
  - Practical and highly successful in practice. A general-purpose extension to existing ML.
  - Optimization, initialization, architecture matter!

Resources

Deep Learning

- *SPRING 2020 · NYU CENTER FOR DATA SCIENCE*
- *INSTRUCTORS: Yann LeCun & Alfredo Canziani*
- *https://atcold.github.io/pytorch-Deep-Learning/*

Stanford Deep Learning tutorial:
  http://ufldl.stanford.edu/wiki

Deep Learning tutorials list:
  http://deeplearning.net/tutorials

IPAM DL/UFL Summer School:
  http://www.ipam.ucla.edu/programs/gss2012/

ICML 2012 Representation Learning Tutorial
  http://www.iro.umontreal.ca/~bengioy/talks/deep-learning-tutorial-2012.html

References

http://www.stanford.edu/~acoates/bmvc2013refs.pdf

Overviews:

Yoshua Bengio,
"Practical Recommendations for Gradient-Based Training of Deep Architectures"

Yoshua Bengio & Yann LeCun,
"Scaling Learning Algorithms towards AI"

Yoshua Bengio, Aaron Courville & Pascal Vincent,
"Representation Learning: A Review and New Perspectives"

Software:

Theano GPU library: http://deeplearning.net/software/theano

SPAMS toolkit: http://spams-devel.gforge.inria.fr/

Class Objectives were:

- Browse main components of deep neural nets
  - Logistic regression w/ its loss function
  - Stack those ones by multiple layers
  - Optimize it w/ stochastic gradient descent
  - Use weights of a layer as features

Homework for Every Class

- **Go over the next lecture slides**
- **Come up with one question on what we have discussed today**
  - **1 for typical questions (that were answered in the class)**
  - **2 for questions with thoughts or that surprised me**
- **Write questions 3 times before the mid-term exam**
  - **Write a question about one out of every four classes**
  - **Multiple questions in one time will be counted as one time**
- **Common questions are compiled at the Q&A file**
  - **Some of questions will be discussed in the class**
- **If you want to know the answer of your question, ask me or TA on person**

KAIST