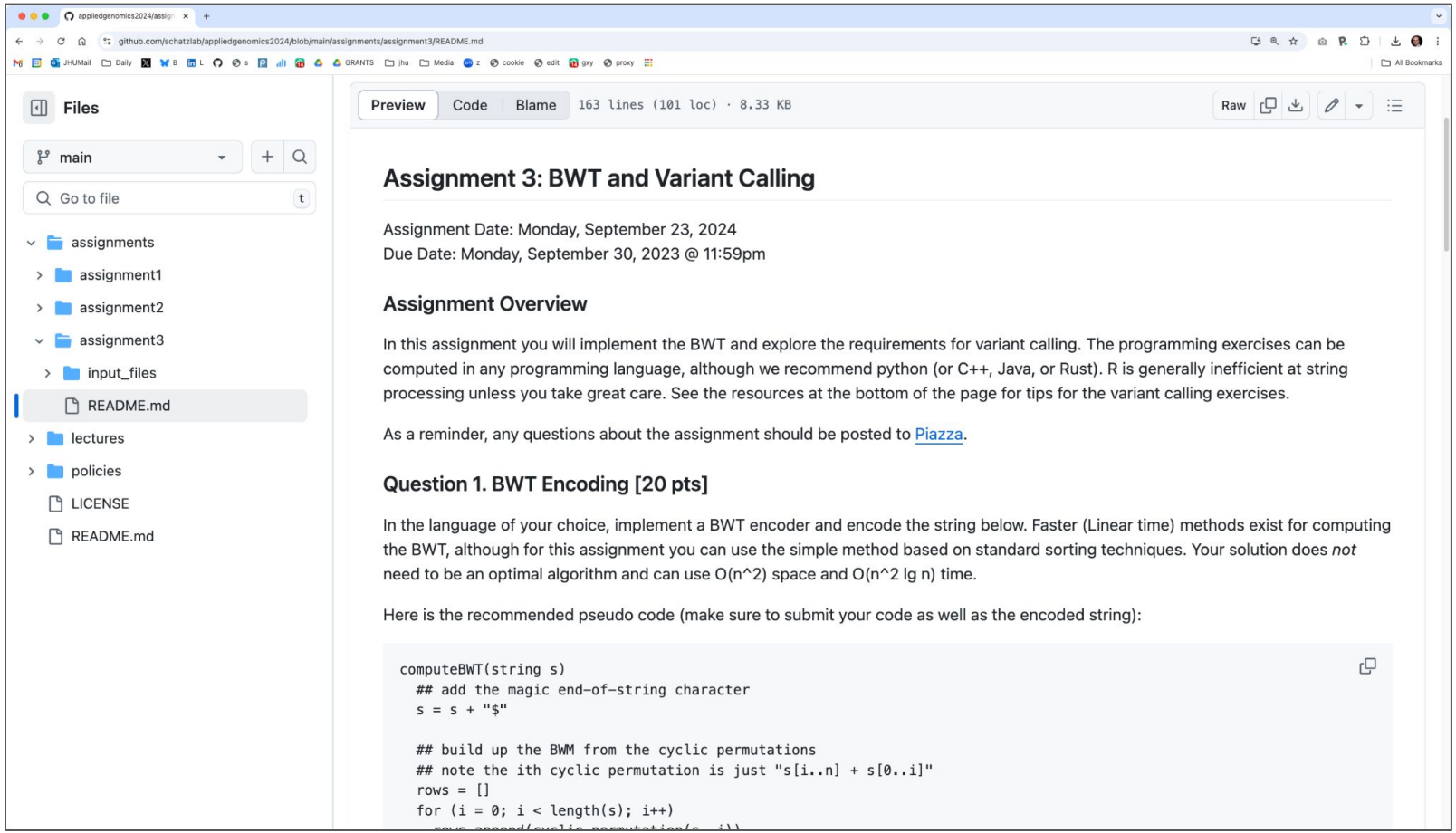
ImageNet Classification with Deep Convolutional Neural Networks

Michael Schatz
Sept 30, 2024

JHU EN.601.449/EN.601.649: Applied Comparative Genomics

Assignment 3 Due Monday September 30



Files

main

Go to file t

- assignments
  - assignment1
  - assignment2
  - assignment3
    - input_files
    - README.md
  - lectures
  - policies
  - LICENSE
  - README.md

Preview | Code | Blame | 163 lines (101 loc) · 8.33 KB | Raw

Assignment 3: BWT and Variant Calling

Assignment Date: Monday, September 23, 2024
Due Date: Monday, September 30, 2023 @ 11:59pm

Assignment Overview

In this assignment you will implement the BWT and explore the requirements for variant calling. The programming exercises can be computed in any programming language, although we recommend python (or C++, Java, or Rust). R is generally inefficient at string processing unless you take great care. See the resources at the bottom of the page for tips for the variant calling exercises.

As a reminder, any questions about the assignment should be posted to Piazza.

Question 1. BWT Encoding [20 pts]

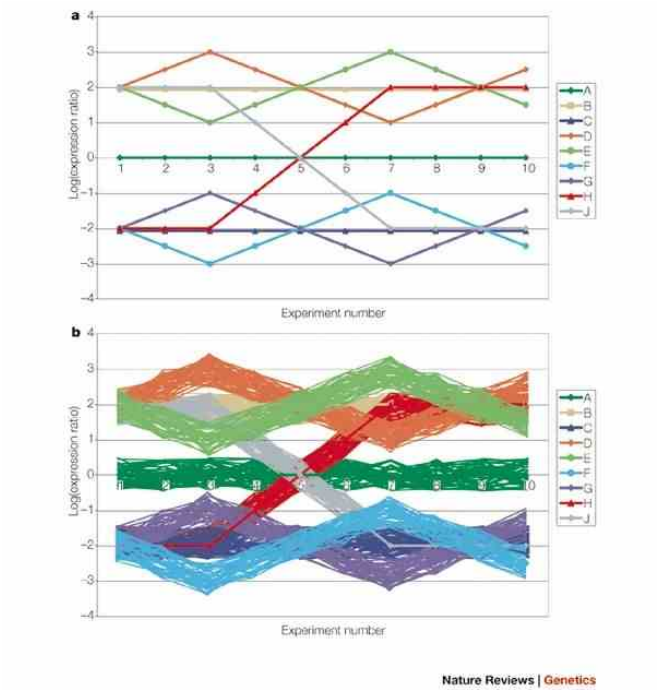
In the language of your choice, implement a BWT encoder and encode the string below. Faster (Linear time) methods exist for computing the BWT, although for this assignment you can use the simple method based on standard sorting techniques. Your solution does *not* need to be an optimal algorithm and can use O(n^2) space and O(n^2 lg n) time.

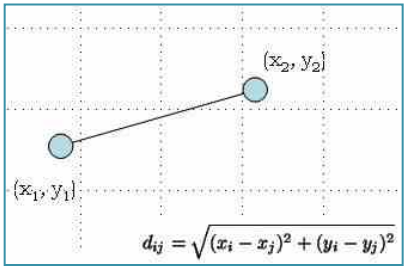
Here is the recommended pseudo code (make sure to submit your code as well as the encoded string):

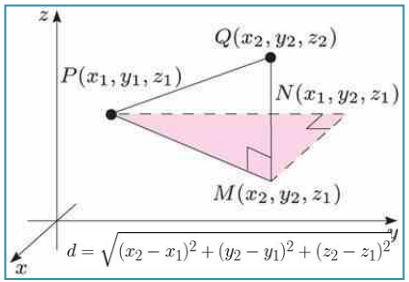
```
computeBWT(string s)
  ## add the magic end-of-string character
  s = s + "$"

  ## build up the BWM from the cyclic permutations
  ## note the ith cyclic permutation is just "s[i..n] + s[0..i]"
  rows = []
  for (i = 0; i < length(s); i++)
```

Recap

Clustering Refresher





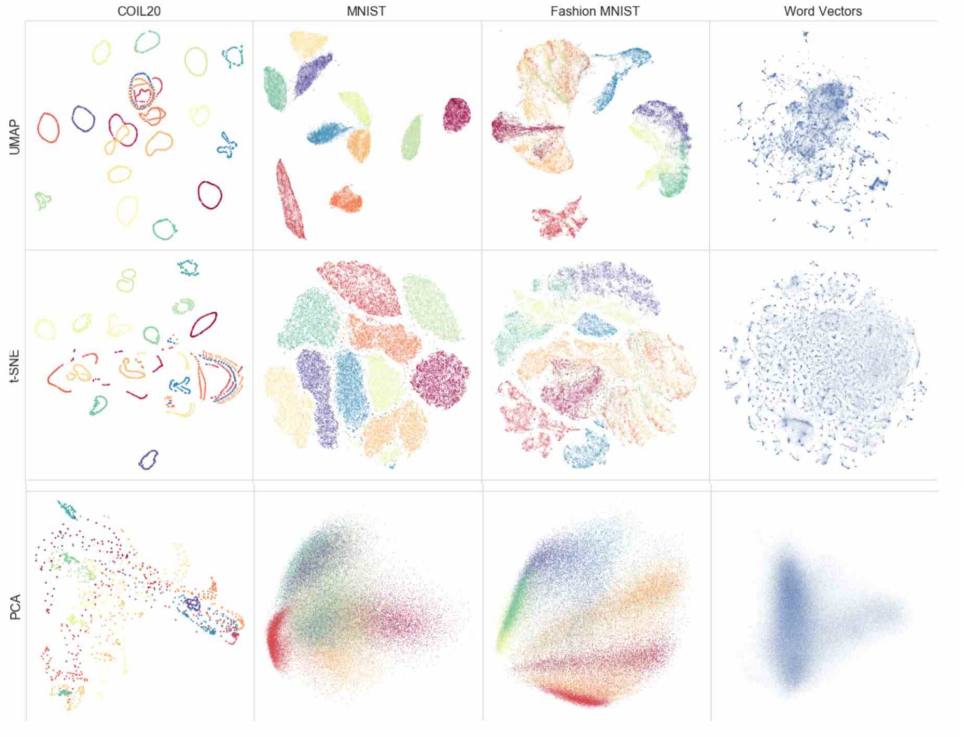


$$\mathrm{d}(\mathbf{p}, \mathbf{q}) = \mathrm{d}(\mathbf{q}, \mathbf{p}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2} = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}.$$

Computational genetics: Computational analysis of microarray data

Quackenbush (2001) *Nature Reviews Genetics.* doi:10.1038/35076576

Dimensionality Reduction



UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction
McInnes et al (2018) arXiv. 1802.03426
https://www.youtube.com/watch?v=nq6iPZVUxZU
https://towardsdatascience.com/how-exactly-umap-works-13e3040e1668

Classification Problems

Supervised learning problem:

- Given (many) labeled training examples, develop a classifier that will accurately predict an output
- Each example contains **features** used to form a prediction

Examples:

- Given a pileup of reads, predict if there really is a variant
- Given SNVs in a person, predict ancestry (or relatedness)
- Given expression matrix, predict sample type
- Given SNVs + cell type, predict changes in expression
- Given SNVs/expression, predict disease risk
- Given SNVs/expression & disease status, predict treatment
- Given an image (photograph, xray, MRI), predict what it is (object, tumor, etc)
- Given English text, classify the sentiment (happy/sad) or predict the next work
- …

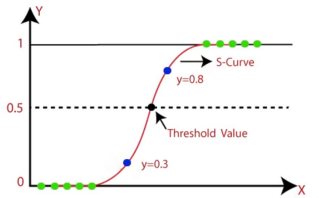
Machine Learning Models

Logistic Regression

Logistic regression assigns a weight to each feature to form a prediction

Pros	Cons
• Easy to understand • Good accuracy when data is linearly separable • Less inclined to **overfit**	• Assumes linearity between output and features • No multicollinearity between features

$$\hat{y}(w,x) = w_0 + w_1x_1 + \ldots + w_px_p$$



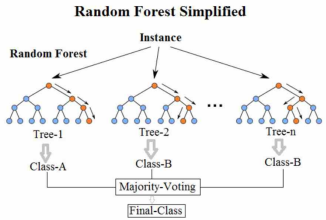
https://encord.com/blog/what-is-logistic-regression/

Random Forests

An ensemble of decision trees built from sample of training data

- Prediction is the averaged output of all decision trees

Pros	Cons
• Highly accurate • Intrinsically interpretable • Can learn non-linear decision boundaries • Parallel processing	• High computational complexity • Can overfit because of noise



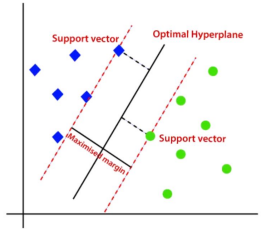
https://www.linkedin.com/pulse/random-forest-algorithm-interactive-discussion-niraj-kumar/

SVM

Support Vector Machines find a hyperplane that best separates the data into classes

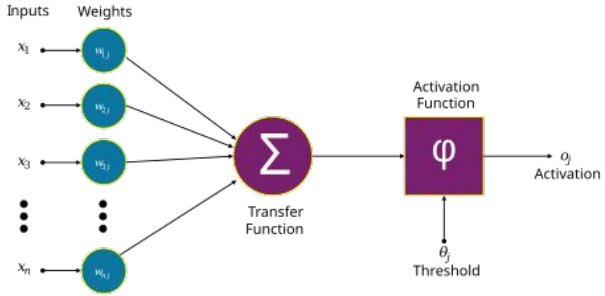
- Maximize margin between different classes

Pros	Cons
• Effectively handles high dimensional data • Can work well with small datasets • Can model non-linear decision boundaries	• Does not work for large data sets • Need to choose kernel • Limited to two-class problems • No probabilistic interpretation

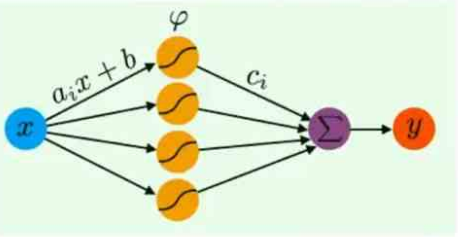


https://medium.com/@ambika199820/support-vector-machine-svm-algorithm-explained-c8facc10bcfa

Neural Networks



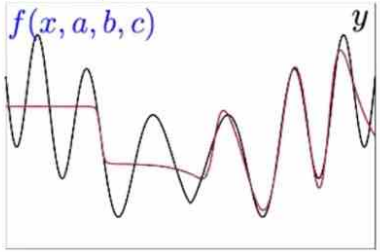
ANNs are “Universal Approximators”



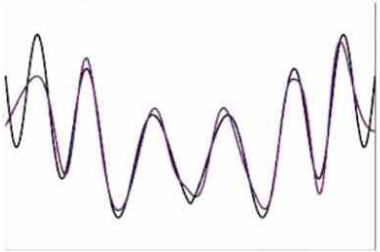
1 hidden layer perceptron:

$$y \approx f(x,a,b,c) \stackrel{\text{def.}}{=} \sum_{i=1}^{p} c_i \varphi(a_i x + b_i)$$

$p = 6$ neurons



$p = 20$ neurons

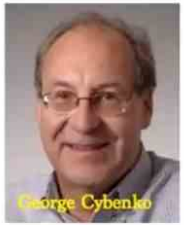


Approximation by Superpositions of a Sigmoidal Function*

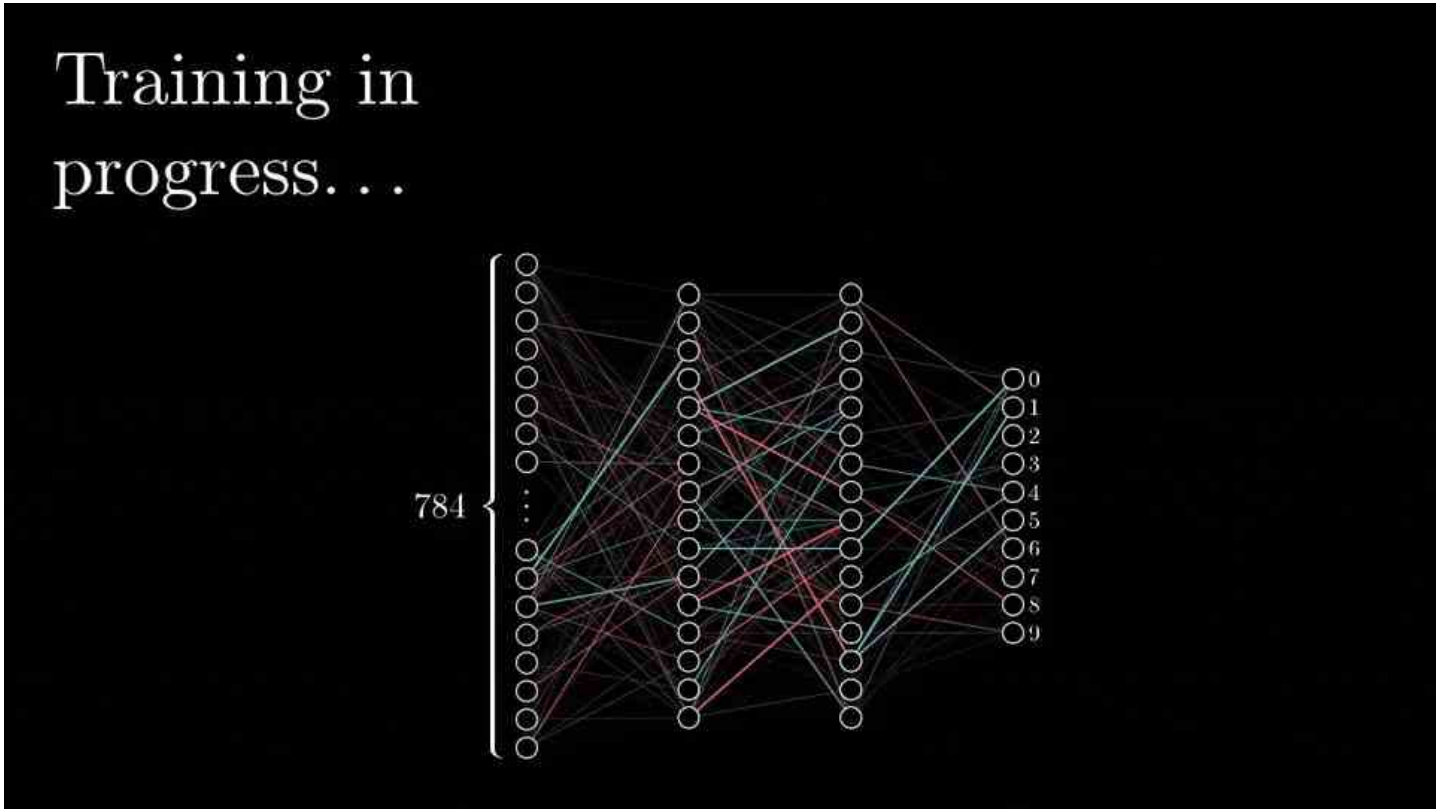
G. Cybenko†

Abstract. In this paper we demonstrate that finite linear combinations of compositions of a fixed, univariate function and a set of affine functionals can uniformly approximate any continuous function of n real variables with support in the unit hypercube; only mild conditions are imposed on the univariate function. Our results settle an open question about representability in the class of single hidden layer neural networks. In particular, we show that arbitrary decision regions can be arbitrarily well approximated by continuous feedforward neural networks with only a single internal, hidden layer and any continuous sigmoidal nonlinearity. The paper discusses approximation properties of other possible types of nonlinearities that might be implemented by artificial neural networks.

Key words. Neural networks, Approximation, Completeness.



Recap - Training



https://youtu.be/IHZwWFHWa-w

AlexNet

“AlexNet”

ImageNet Classification with Deep Convolutional Neural Networks

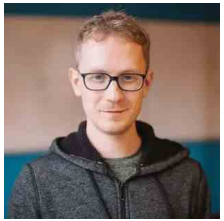
Alex Krizhevsky
University of Toronto
kriz@cs.utoronto.ca

Ilya Sutskever
University of Toronto
ilya@cs.utoronto.ca

Geoffrey E. Hinton
University of Toronto
hinton@cs.utoronto.ca

Abstract

We trained a large, deep convolutional neural network to classify the 1.2 million high-resolution images in the ImageNet LSVRC-2010 contest into the 1000 different classes. On the test data, we achieved top-1 and top-5 error rates of 37.5% and 17.0% which is considerably better than the previous state-of-the-art. The neural network, which has 60 million parameters and 650,000 neurons, consists of five convolutional layers, some of which are followed by max-pooling layers, and three fully-connected layers with a final 1000-way softmax. To make training faster, we used non-saturating neurons and a very efficient GPU implementation of the convolution operation. To reduce overfitting in the fully-connected layers we employed a recently-developed regularization method called “dropout” that proved to be very effective. We also entered a variant of this model in the ILSVRC-2012 competition and achieved a winning top-5 test error rate of 15.3%, compared to 26.2% achieved by the second-best entry.



Alex Krizhevsky
U. Toronto/Google

Ilya Sutskever
U. Toronto/OpenAI/
Safe Superintelligence Inc

Geoffrey Hinton
U. Toronto/Vector Institute

“AlexNet”

Imagenet classification with deep convolutional neural networks

A Krizhevsky, I Sutskever… - Advances in neural …, 2012 - proceedings.neurips.cc Paperpile

We trained a large, deep convolutional neural network to classify the 1.3 million high-resolution images in the LSVRC-2010 ImageNet training set into the 1000 different classes. On the test data, we achieved top-1 and top-5 error rates of 39.7\% and 18.9\% which is considerably better than the previous state-of-the-art results. The neural network, which has 60 million parameters and 500,000 neurons, consists of five convolutional layers, some of which are followed by max-pooling layers, and two globally connected layers with a final …

☆ Save 99 Cite Cited by 131231 Related articles All 98 versions Import into BibTeX »

Outline

1. The Problem
2. pre-AlexNet approaches
3. How do animals & humans approach this?
4. Artificial Neural Networks
5. Convolutional Neural Networks
6. AlexNet
7. Impact

Outline

1. The Problem
2. pre-AlexNet approaches
3. How do animals & humans approach this?
4. Artificial Neural Networks
5. Convolutional Neural Networks
6. AlexNet
7. Impact

What is this?


Ensignbus
Ensignbus
www.ensignbus.com
P185 OLC
01708 865656

What is this?

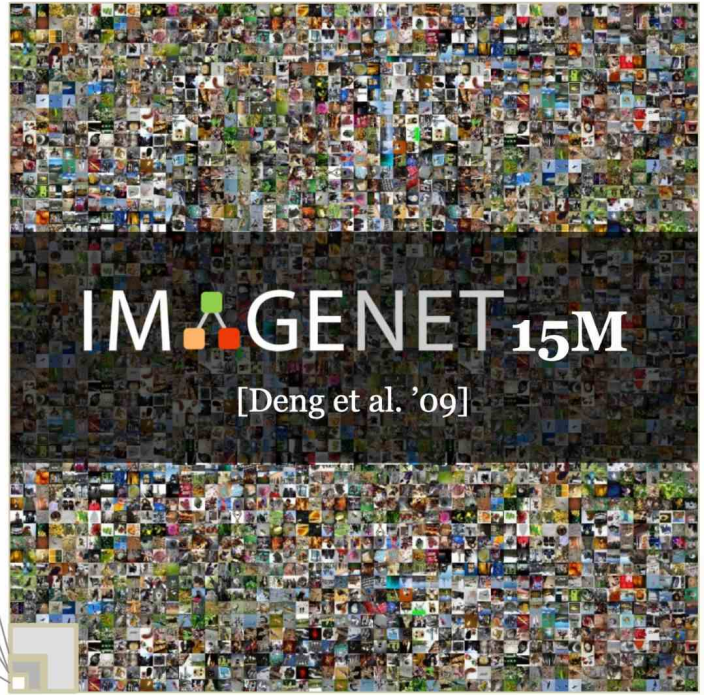
ImageNet Classification Challenge

SUN, 131K
[Xiao et al. '10]

LabelMe, 37K
[Russell et al. '07]

PASCAL VOC, 30K
[Everingham et al. '06-'12]

Caltech101, 9K
[Fei-Fei, Fergus, Perona, '03]



- 15M images organized into 21,841 categories
- ILSVRC (AlexNet) uses a subset:
  - 1000 images in 1000 categories
  - 1.2M training, 50k validation, 150k test

imagenet_ilsvrc2017_v1.0

ImageNet Examples

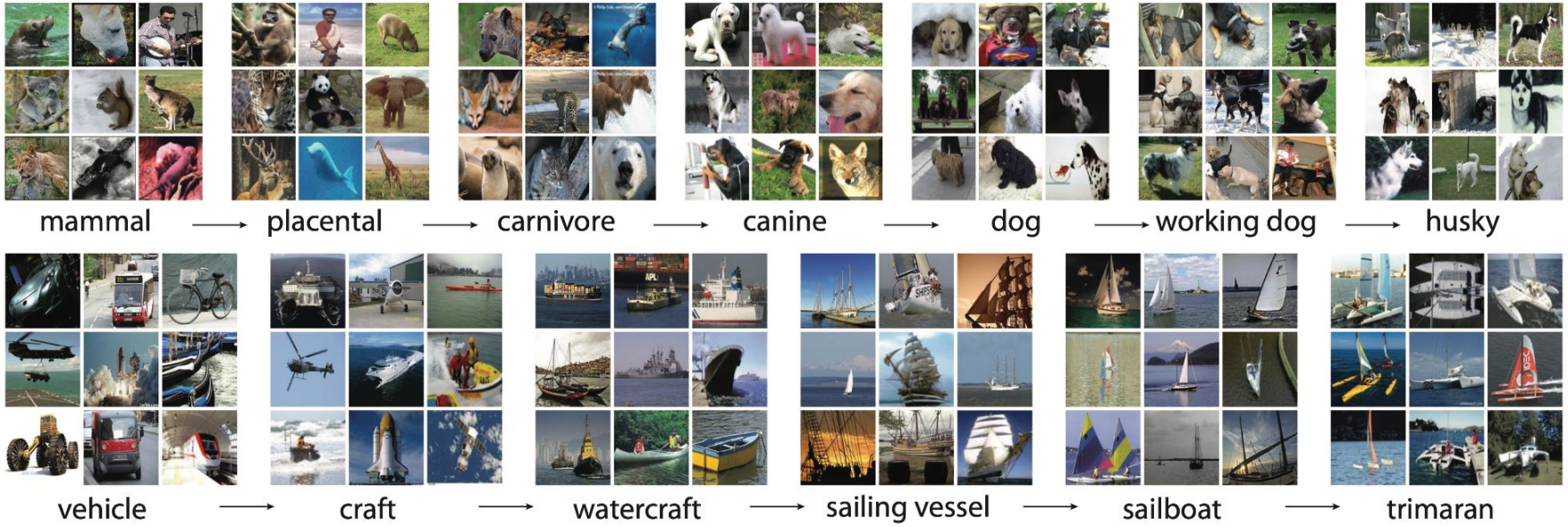


Figure 1: A snapshot of two root-to-leaf branches of ImageNet: the **top** row is from the mammal subtree; the **bottom** row is from the vehicle subtree. For each synset, 9 randomly sampled images are presented.

Images organized into nested categories determined by “WordNet”

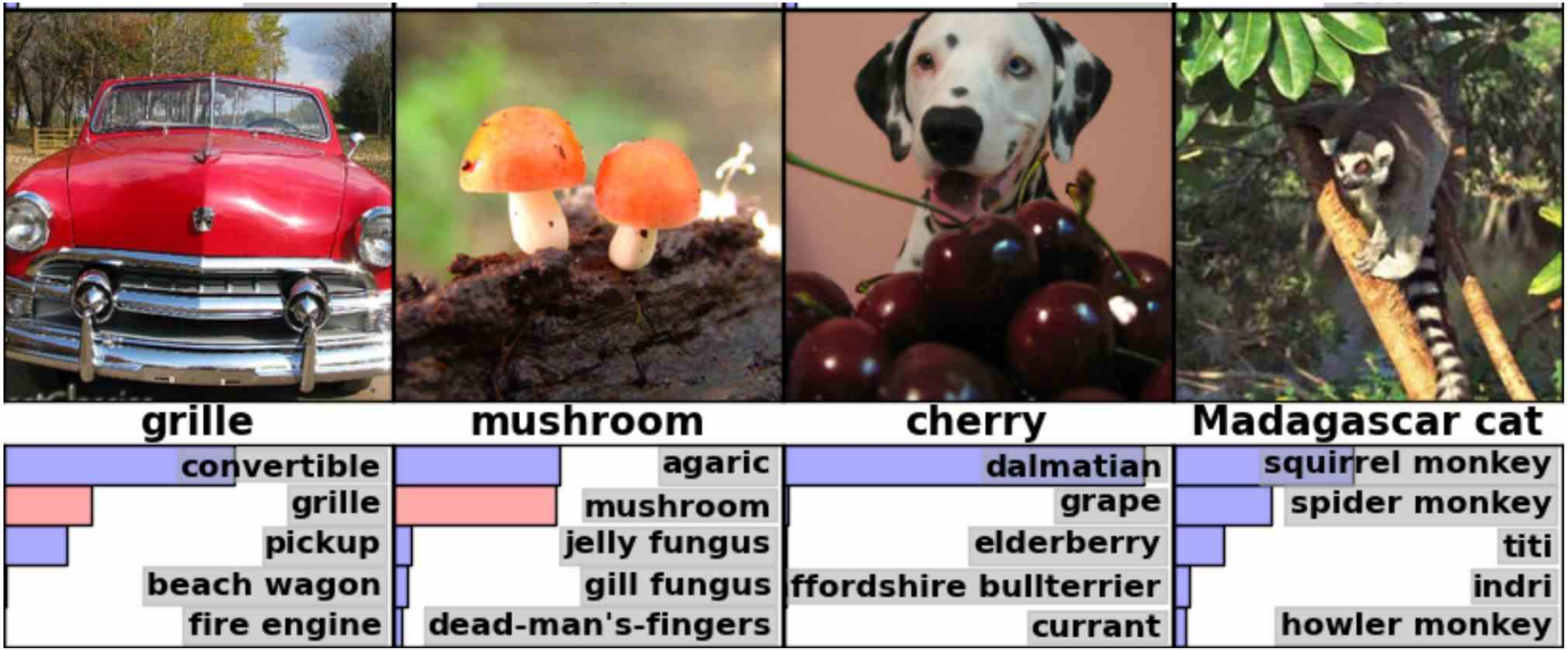
Classification Challenges

husky

- “Natural” variation between individuals
  - Different colors, shapes, sizes
  - Very similar categories: 90 different dog breeds in the challenge (120 total)

- “Image” variations
  - Orientations: side, front, behind, etc
  - Distance & position of the object
  - Other objects in the image

Classification Challenges



Classification “errors” from AlexNet

Outline

1. The Problem
2. pre-AlexNet approaches
3. How do animals & humans approach this?
4. Artificial Neural Networks
5. Convolutional Neural Networks
6. AlexNet
7. Impact

High-Dimensional Signature Compression for Large-Scale Image Classification

Jorge Sánchez and Florent Perronnin
Textual and Visual Pattern Analysis (TVPA) group
Xerox Research Centre Europe (XRCE)

Abstract

We address image classification on a large-scale, i.e. when a large number of images and classes are involved. First, we study classification accuracy as a function of the image signature dimensionality and the training set size. We show experimentally that the larger the training set, the higher the impact of the dimensionality on the accuracy. In other words, high-dimensional signatures are important to obtain state-of-the-art results on large datasets. Second, we tackle the problem of data compression on very large signatures (on the order of 10^5 dimensions) using two lossy compression strategies: a dimensionality reduction technique known as the hush kernel and an encoding technique based on product quantizers. We explain how the gain in storage can be traded against a loss in accuracy and / or an increase in CPU cost. We report results on two large databases – ImageNet and a dataset of 1M Flickr images – showing that we can reduce the storage of our signatures by a factor 64 to 128 with little loss in accuracy. Integrating the decompression in the classifier learning yields an efficient and scalable training algorithm. **On ILSVRC2010 we report a 74.3% accuracy at top-5, which corresponds to a 2.5% absolute improvement with respect to the state-of-the-art. On a subset of 10K classes of ImageNet we report a top-1 accuracy of 16.7%, a relative improvement of 160% with respect to the state-of-the-art.**

1. Introduction

Scaling-up image classification systems is a problem which is receiving an increasing attention as larger labeled image datasets are becoming available. For instance, ImageNet (www.image-net.org) consists of more than 12M images of 17K concepts [7] and Flickr contains thousands of groups (www.flickr.com/groups) – some of which with hundreds of thousands of pictures – which can be readily used to learn object classifiers [31, 22].

The focus in the image classification community was initially on developing systems which would yield the best possible accuracy fairly independently of their cost. The winners of the PASCAL VOC 2007 [8] and 2008 [9] competitions used a similar paradigm: many types of low-level local features are extracted (referred to as "channels"), one bag-of-visual-words (BOV) histogram is computed for each channel and non-linear kernel classifiers such as SVMs are used to perform classification [38, 29]. The use of many channels and costly non-linear SVMs was made possible by the modest size of the available databases.

In recent years only has the computational cost become a central issue in image classification / object detection. In [19], Maji *et al.* showed that the runtime cost of an intersection kernel (IK) SVM could be made independent of the number of support vectors. Maji and Berg [18] and Wang *et al.* [31] then proposed efficient algorithms to learn IKSVMs. Vedaldi and Zisserman [30] and Perronnin *et al.* [21] subsequently generalized this principle to any additive classifier. Another line of research consists in computing image representations which are directly amenable to costless linear classification. Yang *et al.* [36], Wang *et al.* [32] and Boureau *et al.* [4] showed that replacing the average pooling stage in the BOV computation by a max-pooling yielded excellent results. To go beyond the BOV, *i.e.* beyond counting, it has been proposed to include higher order statistics in the image signature. This includes modeling an image by a probability distribution [17, 35] or using the Fisher kernel framework [20]. Especially, it was shown that the Fisher Vector (FV) could yield high accuracy with linear classifiers [22].

If one wants to stick to efficient linear classifiers, the image representations should be high-dimensional to ensure linear separability of the classes. Therefore, we argue that *the storage/memory cost is becoming a central issue in large-scale image classification*. As an example, in this paper we consider almost dense image representations – based on the improved FV of [22] – with up to $524K$ dimensions. Using a 4 byte floating point representation, a single signature requires 2MB of storage. Storing the ILSVRC2010 dataset [2] would take approximately 2.8TBs and storing the full ImageNet dataset around 23TBs. Obviously, these numbers have to be multiplied by the number of channels, *i.e.* feature types. As another example, the

The winner in 2011

1. Break an image into (many) patches

2. Transform local features into "visual words" using a "code book"

3. Use the visual words to create a sparse feature vector

4. Compare the vector to the vectors precomputed from your database

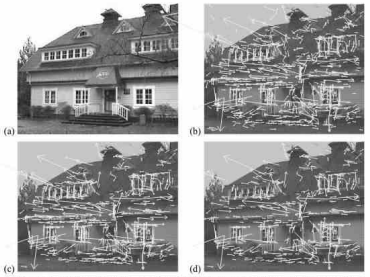
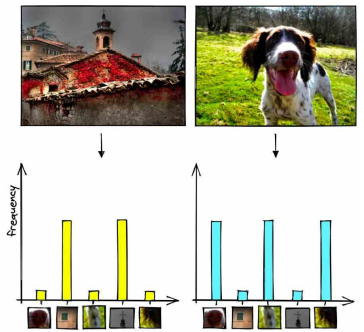


Figure 5: This figure shows the stages of keypoint selection. (a) The 233x189 pixel original image. (b) The initial 832 keypoints locations at maxima and minima of the difference-of-Gaussian function. Keypoints are displayed as vectors indicating scale, orientation, and location. (c) After applying a threshold on minimum contrast, 729 keypoints remain. (d) The final 536 keypoints that remain following an additional threshold on ratio of principal curvatures.

"SIFT descriptors"



"Visual word histogram"

High-Dimensional Signature Compression for Large-Scale Image Classification

Jorge Sánchez and Florent Perronnin
Textual and Visual Pattern Analysis (TVPA) group
Xerox Research Centre Europe (XRCE)

Abstract

We address image classification on a large-scale, i.e. *when a large number of images and classes are involved. First, we study classification accuracy as a function of the image signature dimensionality and the training set size. We show experimentally that the larger the training set, the higher the impact of the dimensionality on the accuracy. In other words, high-dimensional signatures are important to obtain state-of-the-art results on large datasets. Second, we tackle the problem of data compression on very large signatures (on the order of 10^5 dimensions) using two lossy compression strategies: a dimensionality reduction technique known as the hush kernel and an encoding technique based on product quantizers. We explain how the gain in storage can be traded against a loss in accuracy and / or an increase in CPU cost. We report results on two large databases – ImageNet and a dataset of 1M Flickr images – showing that we can reduce the storage of our signatures by a factor 64 to 128 with little loss in accuracy. Integrating the decompression in the classifier learning yields an efficient and scalable training algorithm.* **On ILSVRC2010 we report a 74.3% accuracy at top-5, which corresponds to a 2.5% absolute improvement with respect to the state-of-the-art. On a subset of 10K classes of ImageNet we report a top-1 accuracy of 16.7%, a relative improvement of 160% with respect to the state-of-the-art.**

1. Introduction

Scaling-up image classification systems is a problem which is receiving an increasing attention as larger labeled image datasets are becoming available. For instance, ImageNet (www.image-net.org) consists of more than 12M images of 17K concepts [7] and Flickr contains thousands of groups (www.flickr.com/groups) – some of which with hundreds of thousands of pictures – which can be readily used to learn object classifiers [31, 22].

The focus in the image classification community was initially on developing systems which would yield the best possible accuracy fairly independently of their cost. The winners of the PASCAL VOC 2007 [8] and 2008 [9] competitions used a similar paradigm: many types of low-level local features are extracted (referred to as "channels"), one bag-of-visual-words (BOV) histogram is computed for each channel and non-linear kernel classifiers such as SVMs are used to perform classification [38, 29]. The use of many channels and costly non-linear SVMs was made possible by the modest size of the available databases.

In recent years only has the computational cost become a central issue in image classification / object detection. In [19], Maji *et al.* showed that the runtime cost of an intersection kernel (IK) SVM could be made independent of the number of support vectors. Maji and Berg [18] and Wang *et al.* [31] then proposed efficient algorithms to learn IKSVMs. Vedaldi and Zisserman [30] and Perronnin *et al.* [21] subsequently generalized this principle to any additive classifier. Another line of research consists in computing image representations which are directly amenable to costless linear classification. Yang *et al.* [36], Wang *et al.* [32] and Boureau *et al.* [4] showed that replacing the average pooling stage in the BOV computation by a max-pooling yielded excellent results. To go beyond the BOV, *i.e.* beyond counting, it has been proposed to include higher order statistics in the image signature. This includes modeling an image by a probability distribution [17, 35] or using the Fisher kernel framework [20]. Especially, it was shown that the Fisher Vector (FV) could yield high accuracy with linear classifiers [22].

If one wants to stick to efficient linear classifiers, the image representations should be high-dimensional to ensure linear separability of the classes. Therefore, we argue that *the storage/memory cost is becoming a central issue in large-scale image classification*. As an example, in this paper we consider almost dense image representations – based on the improved FV of [22] – with up to $524K$ dimensions. Using a 4 byte floating point representation, a single signature requires 2MB of storage. Storing the ILSVRC2010 dataset [2] would take approximately 2.8TBs and storing the full ImageNet dataset around 23TBs. Obviously, these numbers have to be multiplied by the number of channels, *i.e.* feature types. As another example, the

1665

The winner in 2011

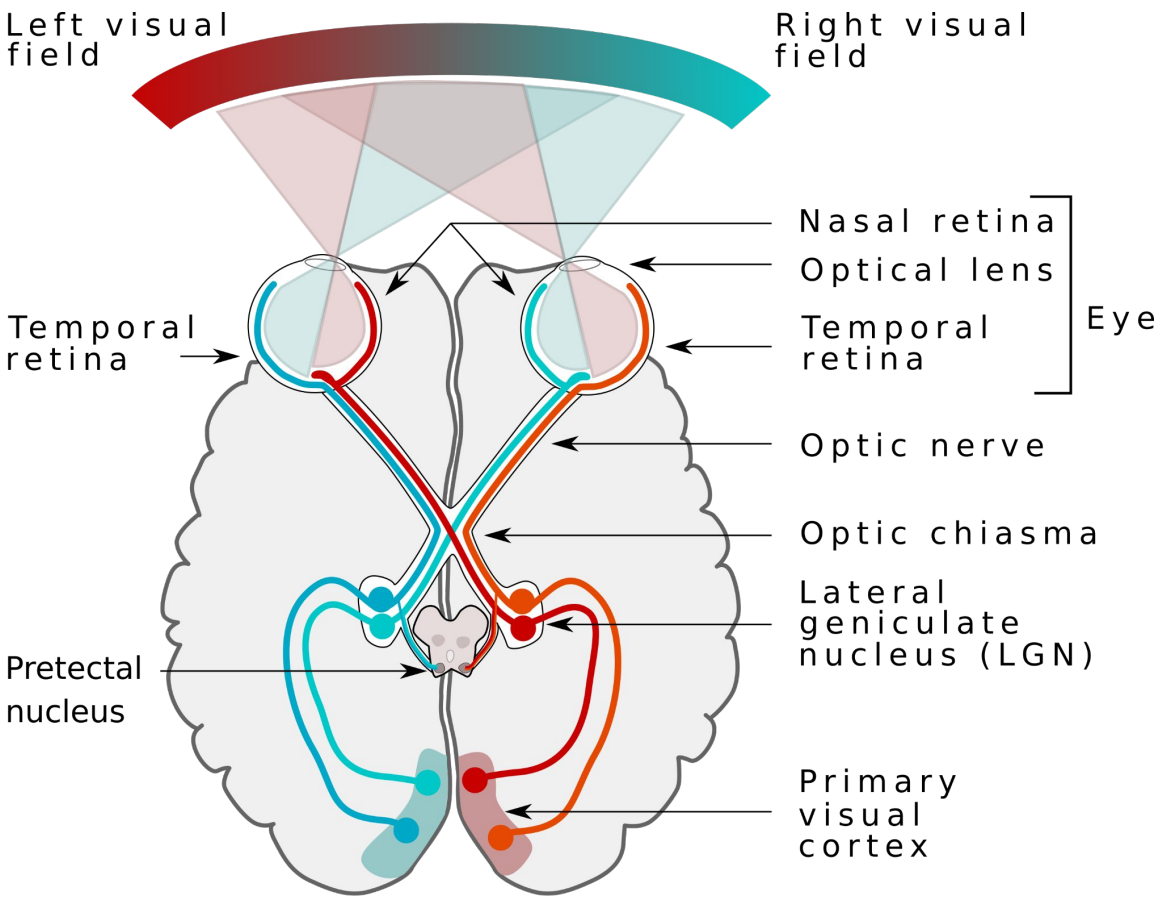
Limitations

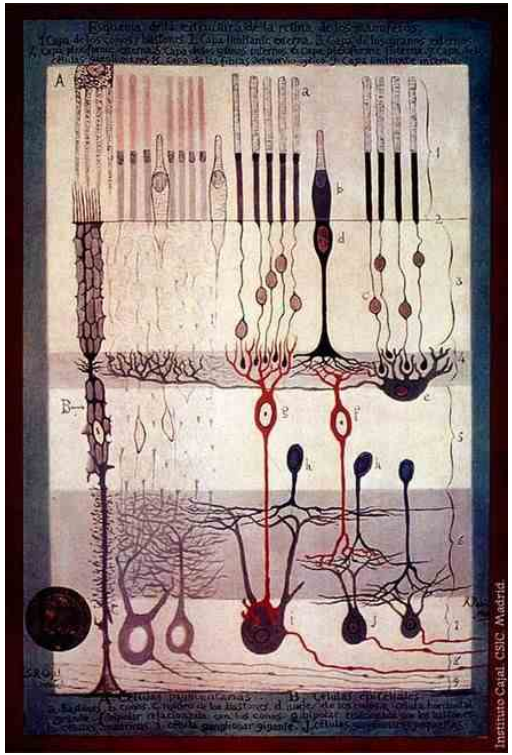
- Ignores spatial relationships between patches ("a bag of words")
- High dependency of local features that are (often) hand crafted to maximize performance on the training data
- Computationally demanding to extract features, perform high dimensional comparisons
- >25% error

Outline

1. The Problem
2. pre-AlexNet approaches
3. How do animals & humans approach this?
4. Artificial Neural Networks
5. Convolutional Neural Networks
6. AlexNet
7. Impact

Human Visual Cortex





Structure of the Mammalian Retina
S. Ramón y Cajal (1900)

Vision & Pattern Recognition

Receptive Fields, Binocular Interaction and Functional Architecture in the Cat's Visual Cortex

D. H. HUBEL AND T. N. WIESEL • *Neurophysiology Laboratory, Department of Pharmacology Harvard Medical School, Boston, Massachusetts*

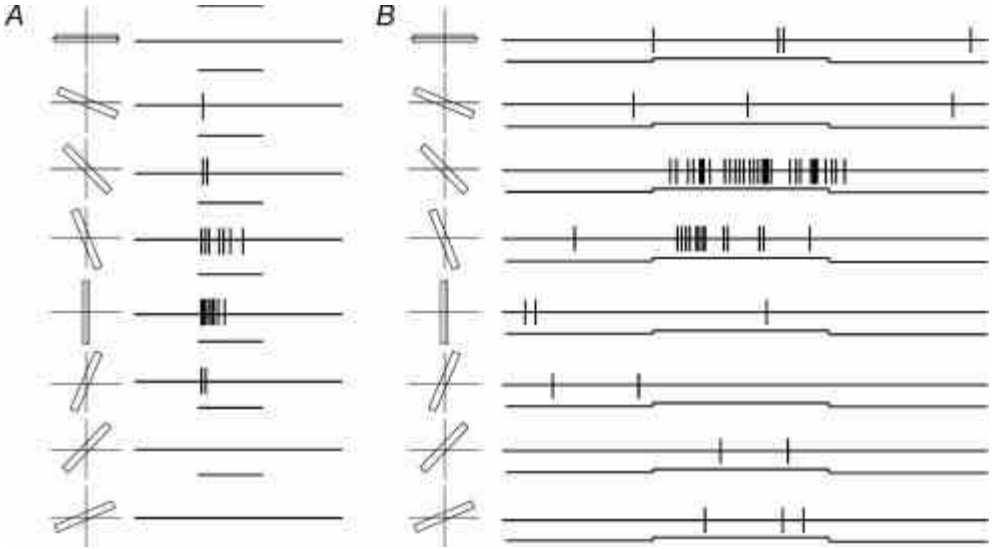


Table 1. Simple Cortical Fields

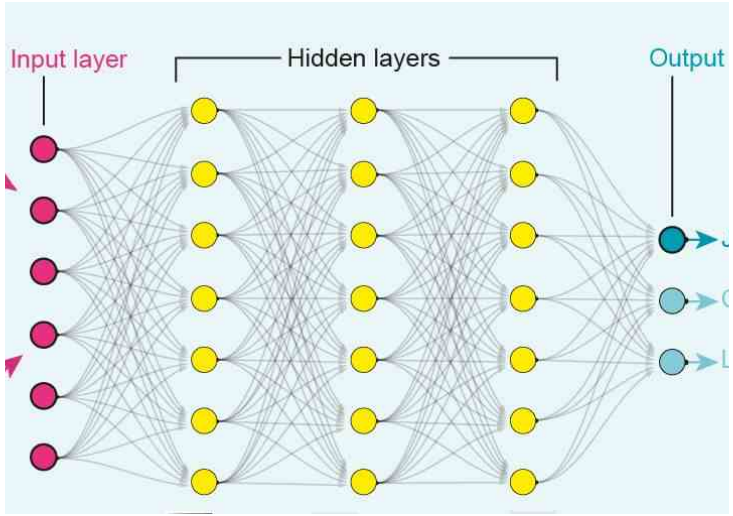
	Text-fig.	*No. of cells*
(*a*) Narrow concentrated centres		
(i) Symmetrical flanks		
Excitatory centres	*2C*	23
Inhibitory centres	*2D*	17
(ii) Asymmetrical flanks		
Excitatory centres	—	28
Inhibitory centres	*2E*	10
(*b*) Large centres; concentrated flanks	*2F*	21
(*c*) One excitatory region and one inhibitory	*2G*	17
(*d*) Uncategorized	—	117
Total number of simple fields		233

Example simple cells in the striate cortex of anaesthetized paralysed cat (A) and the awake fixating monkey (B). The comparison illustrates the similar response to oriented slits of light in the anaesthetized, paralysed cats and awake behaving monkeys. The example from the awake monkey shows the same qualitative orientation tuning but a slightly higher background rate that from the anaesthetized cat. Traced from Fig. 3A of Hubel & Wiesel (1959) and from Fig. 5 of (Wurtz, 1969c).

Outline

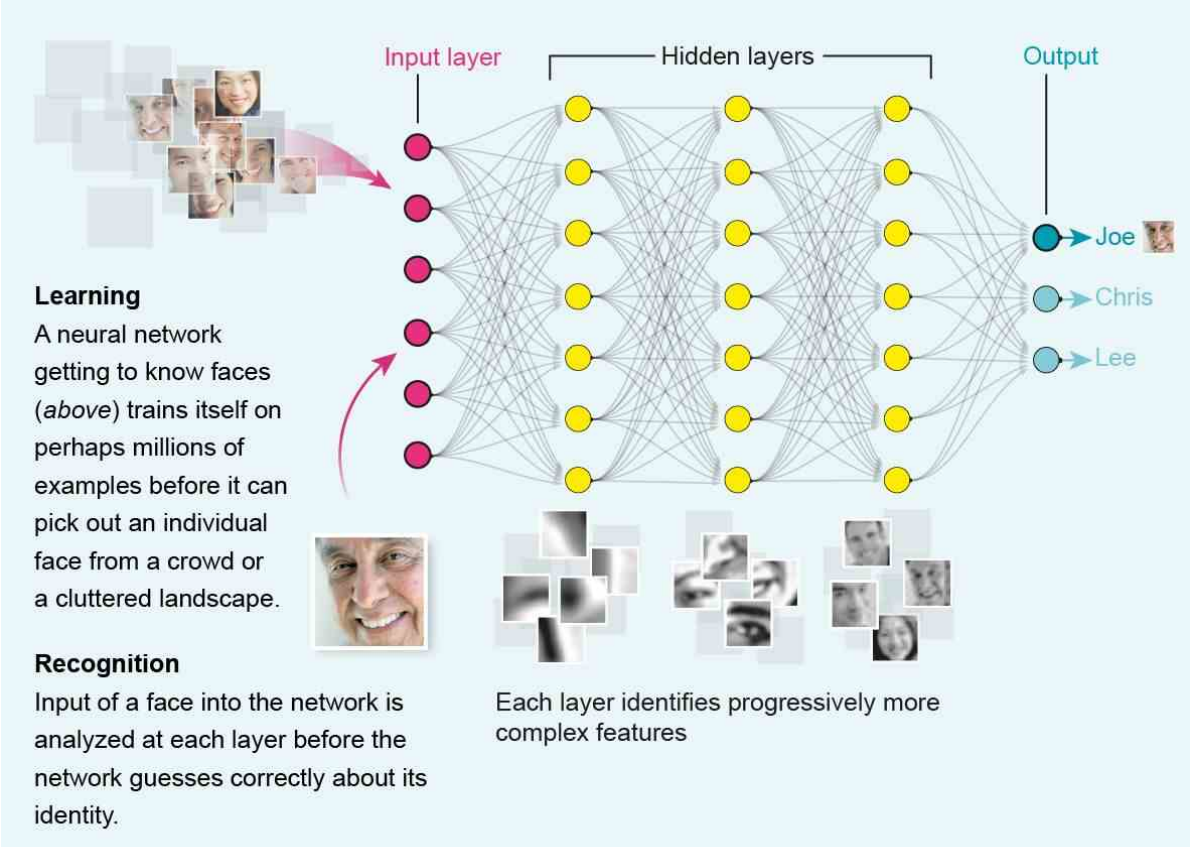
1. The Problem
2. pre-AlexNet approaches
3. How do animals & humans approach this?
4. Artificial Neural Networks
5. Convolutional Neural Networks
6. AlexNet
7. Impact

Artificial Neural Networks



https://www.codingdojo.com/blog/artificial-neural-network-brain-behind-todays-smart-technology

Artificial Neural Networks

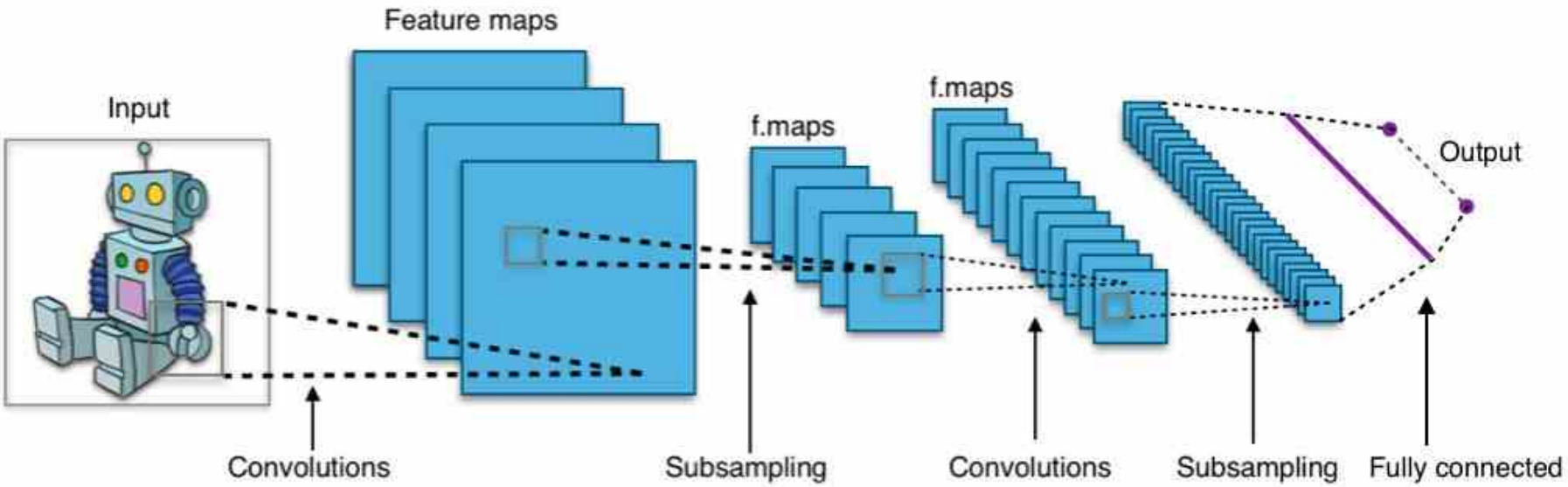


https://www.codingdojo.com/blog/artificial-neural-network-brain-behind-todays-smart-technology

Outline

1. The Problem
2. pre-AlexNet approaches
3. How do animals & humans approach this?
4. Artificial Neural Networks
5. Convolutional Neural Networks
6. AlexNet
7. Impact

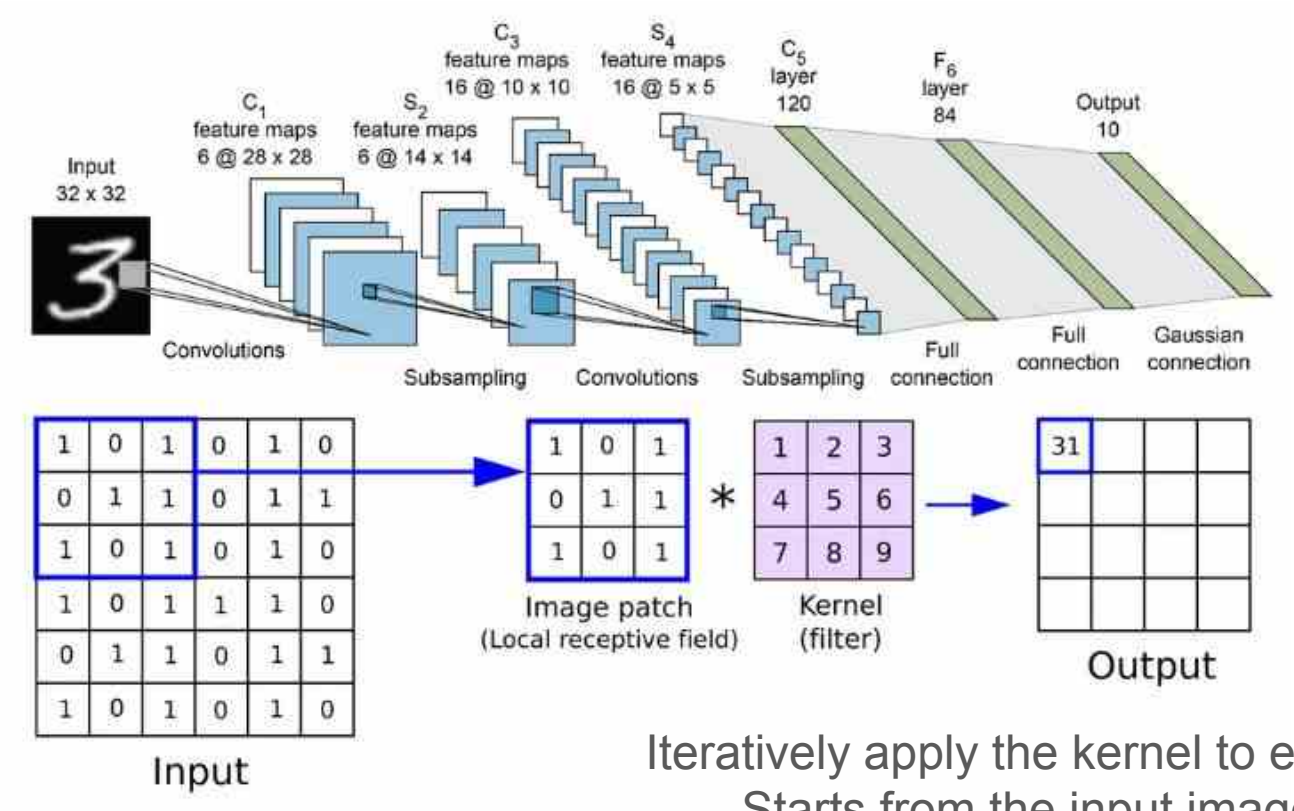
Convolutional Neural Network



Fully connected neural networks are most expressive but are computationally intensive!
CNNs achieve nearly the same power but with much less compute:

- O(Nodes * Depth) rather than O(Nodes ^ Depth)

Convolutional Neural Network



"LeNet" (Lecun et al, 1998)

Iteratively apply the kernel to extract features

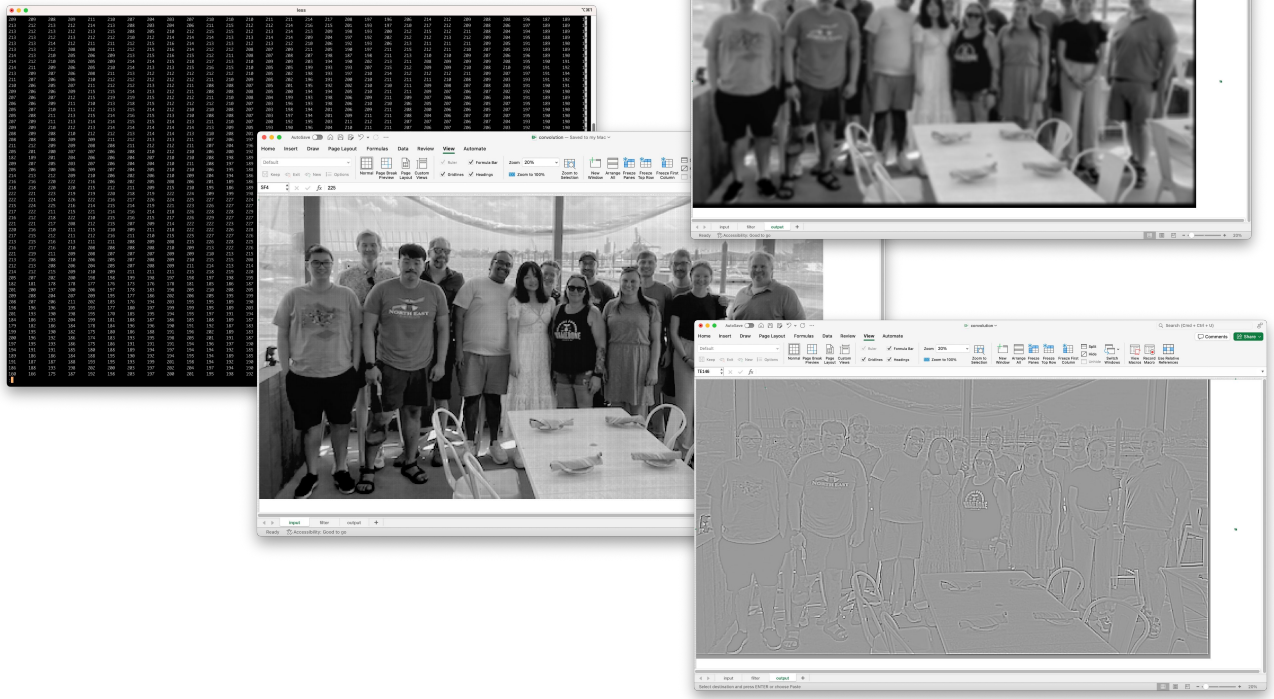
- Starts from the input image, then creates more abstract derived images

Kernels

Operation	Kernel ω	Image result g(x,y)
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Ridge or edge detection	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	
Gaussian blur 3 × 3 (approximation)	$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	
Gaussian blur 5 × 5 (approximation)	$\frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$	
Unsharp masking 5 × 5 Based on Gaussian blur with amount as 1 and threshold as 0 (with no image mask)	$\frac{-1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & -476 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$	

Kernels

Operation	Kernel ω	Image result g(x,y)
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Ridge or **edge detection**	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	
Gaussian blur 3 × 3 (approximation)	$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	
Gaussian blur 5 × 5 (approximation)	$\frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$	
Unsharp masking 5 × 5 Based on Gaussian blur with amount as 1 and threshold as 0 (with no image mask)	$\frac{-1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & -476 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$	



= input!A1 * filter!$A$1 + input!A2 * filter!$A$2 + input!A3 * filter!$A$3 + input!A4 * filter!$A$4 + input!A5 * filter!$A$5 +
input!B1 * filter!$B$1 + input!B2 * filter!$B$2 + input!B3 * filter!$B$3 + input!B4 * filter!$B$4 + input!B5 * filter!$B$5 +
input!C1 * filter!$C$1 + input!C2 * filter!$C$2 + input!C3 * filter!$C$3 + input!C4 * filter!$C$4 + input!C5 * filter!$C$5 +
input!D1 * filter!$D$1 + input!D2 * filter!$D$2 + input!D3 * filter!$D$3 + input!D4 * filter!$D$4 + input!D5 * filter!$D$5 +
input!E1 * filter!$E$1 + input!E2 * filter!$E$2 + input!E3 * filter!$E$3 + input!E4 * filter!$E$4 + input!E5 * filter!$E$5

Outline

1. The Problem
2. pre-AlexNet approaches
3. How do animals & humans approach this?
4. Artificial Neural Networks
5. Convolutional Neural Networks
6. AlexNet
7. Impact

ImageNet Classification with Deep Convolutional Neural Networks

1

2

3

4

5

6

7

8

9

Network Architecture

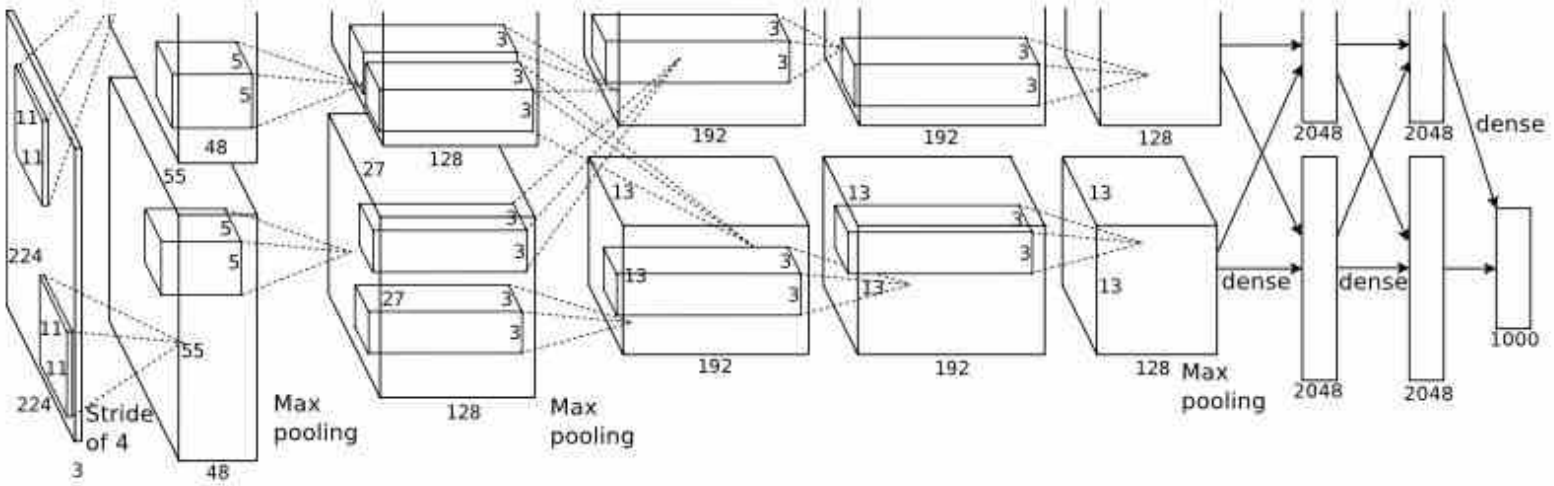
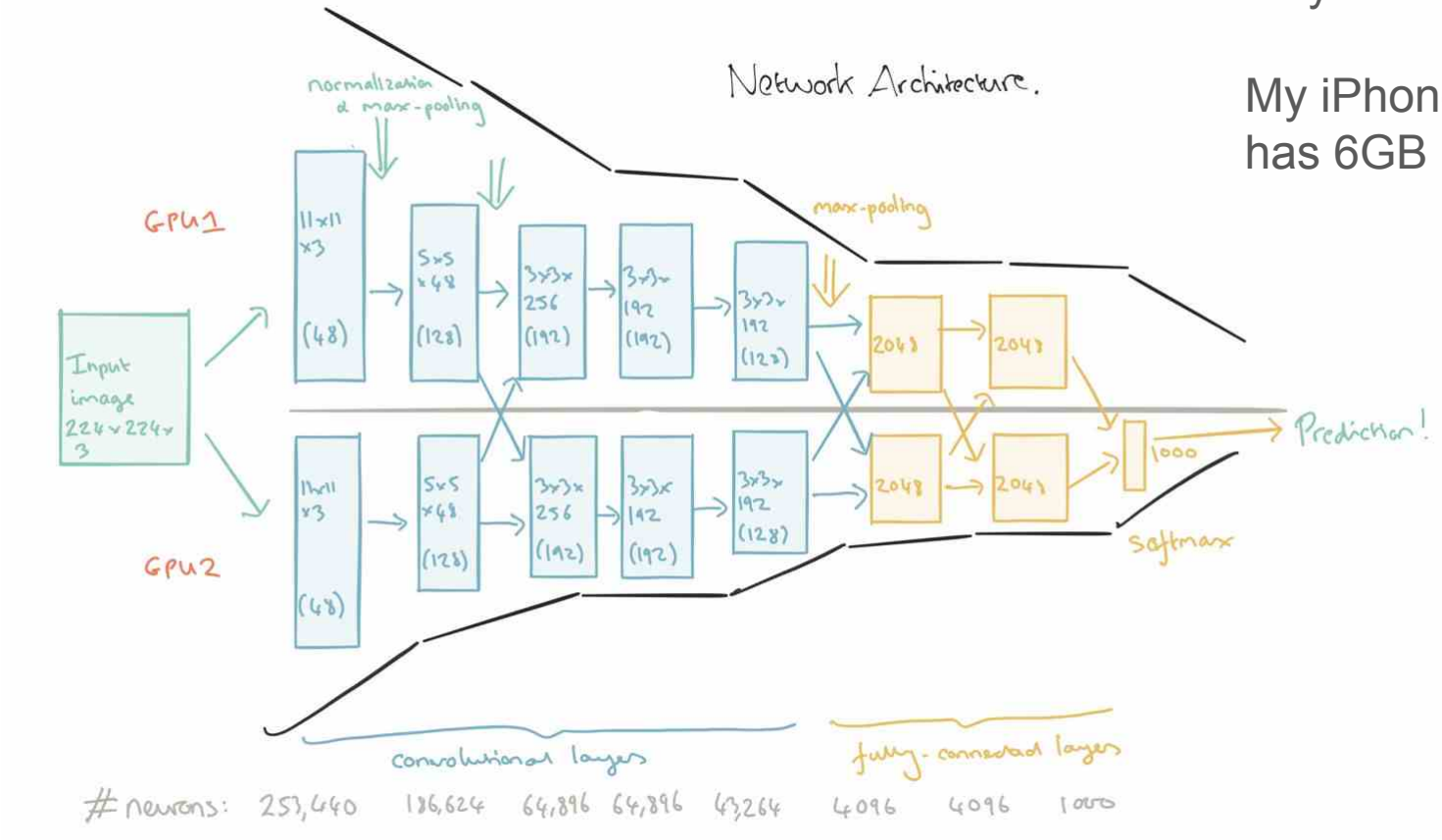


Figure 2: An illustration of the architecture of our CNN, explicitly showing the delineation of responsibilities between the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-parts at the bottom. The GPUs communicate only at certain layers. The network's input is 150,528-dimensional, and the number of neurons in the network's remaining layers is given by 253,440–186,624–64,896–64,896–43,264–4096–4096–1000.

Network Architecture (annotated)

GTX 580 GPU only has 3GB!

My iPhone's GPU has 6GB



https://blog.acolyer.org/2016/04/20/imagenet-classification-with-deep-convolutional-neural-networks/

Training process

Stochastic Gradient Descent

- Training process
  - Minimizing the cross-entropy loss function:

$$L(w) = \sum_{i=1}^{N} \sum_{c=1}^{1000} -y_{ic} \log f_c(x_i) + \epsilon \|w\|_2^2$$

predicted probability of class c for image x

indicator that example i has label c

- SGD with a batch size of 128
- Learning rate initialized at 0.01; divided by 10 if validation error rate stopped improving
- Update rule for weight w:

weight decay

learning rate

$$v_{i+1} := 0.9 * v_i - 0.0005 * \epsilon * w_i - \epsilon * \left\langle \left. \frac{\partial L}{\partial w} \right|_{w_i} \right\rangle_{D_i}$$

$$w_{i+1} := w_i + v_{i+1}$$

momentum

Gradient of Loss

- ~ 90 cycles → five to six days on two NVIDIA GTX 580 3GB GPUs

https://cvml.ista.ac.at/courses/DLWT_W17/material/AlexNet.pdf

Learned Convolutional Kernels

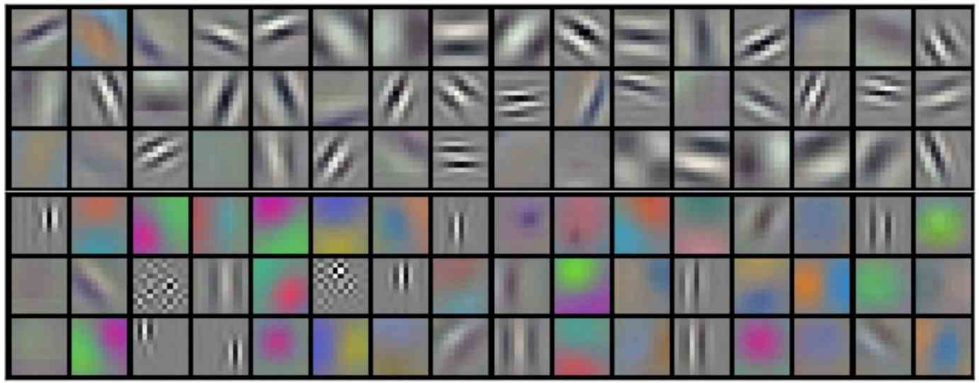
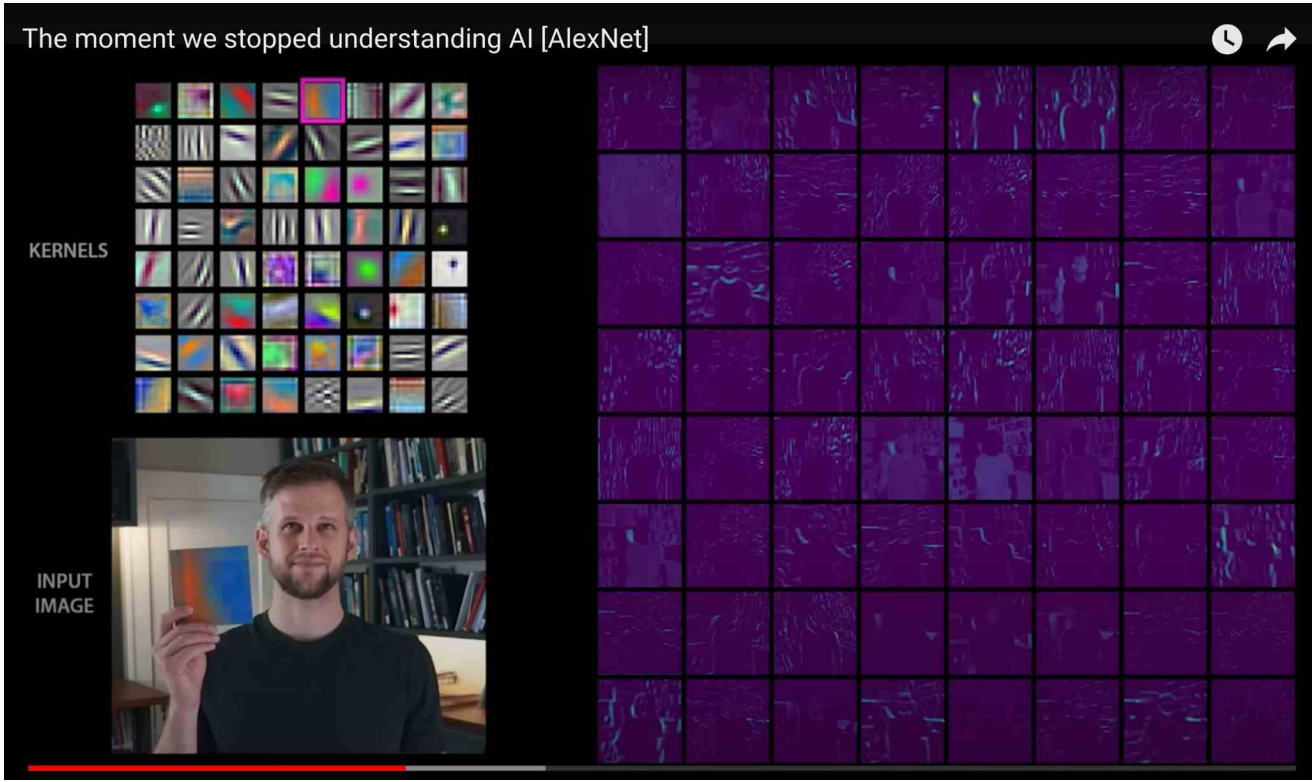


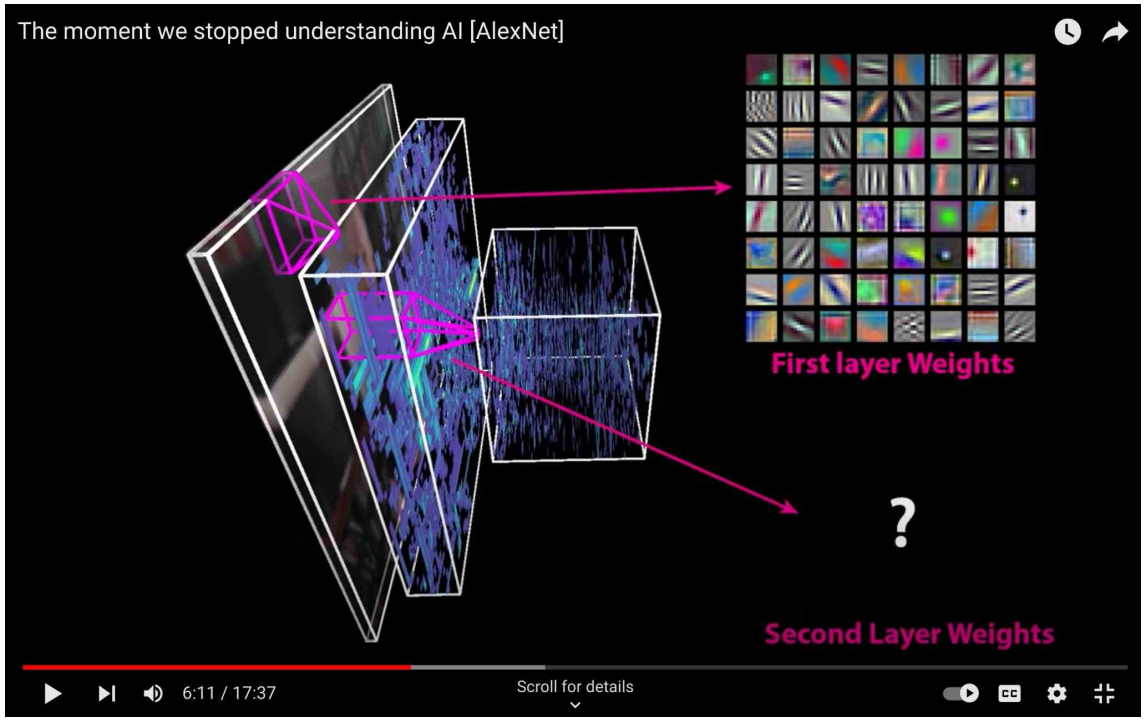
Figure 3: 96 convolutional kernels of size $11 \times 11 \times 3$ learned by the first convolutional layer on the $224 \times 224 \times 3$ input images. The top 48 kernels were learned on GPU 1 while the bottom 48 kernels were learned on GPU 2. See Section 6.1 for details.

From 1 image to 96 images: Activation Maps



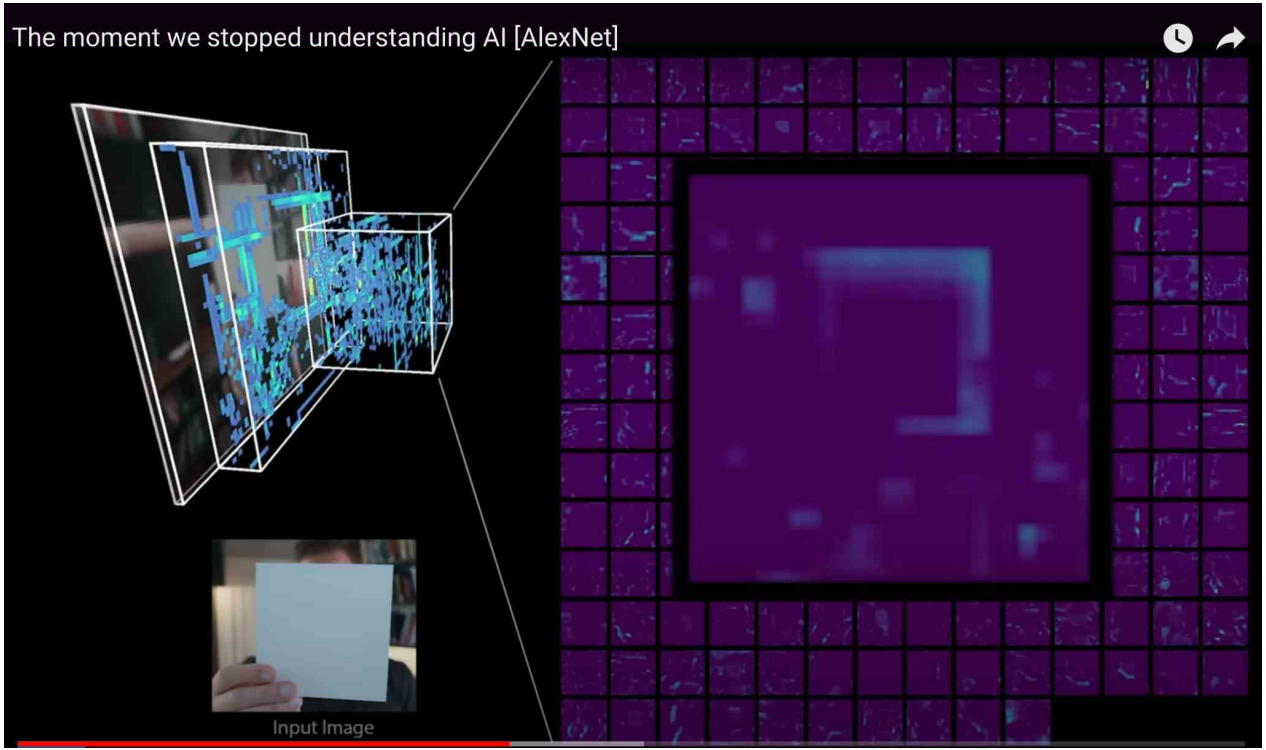
https://www.youtube.com/watch?v=UZDiGooFs54

Repeat convolution for deeper layers
But hard to visualize: 1 image becomes a stack of 96



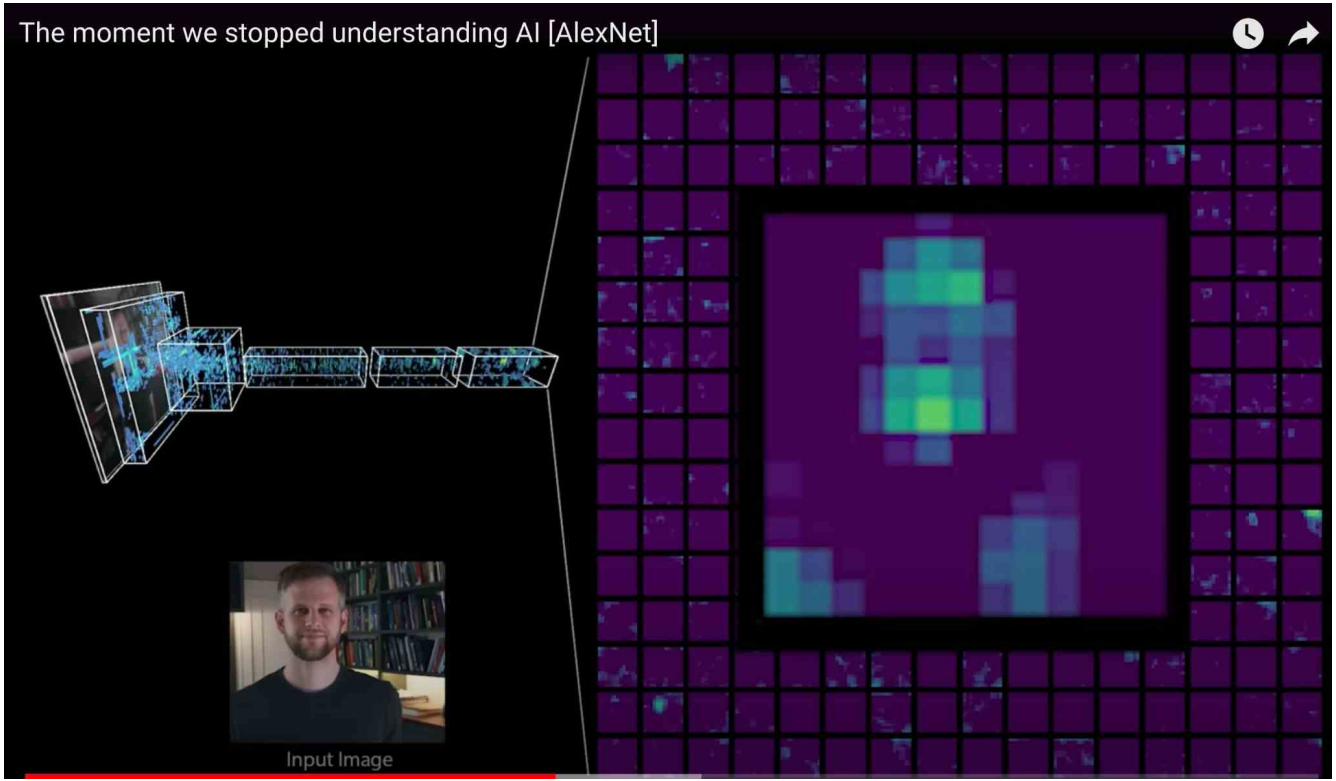
https://www.youtube.com/watch?v=UZDiGooFs54

Testing images shows the deeper layers are learning more complicated features



https://www.youtube.com/watch?v=UZDiGooFs54

Deeper layers recognize very complex features: faces!

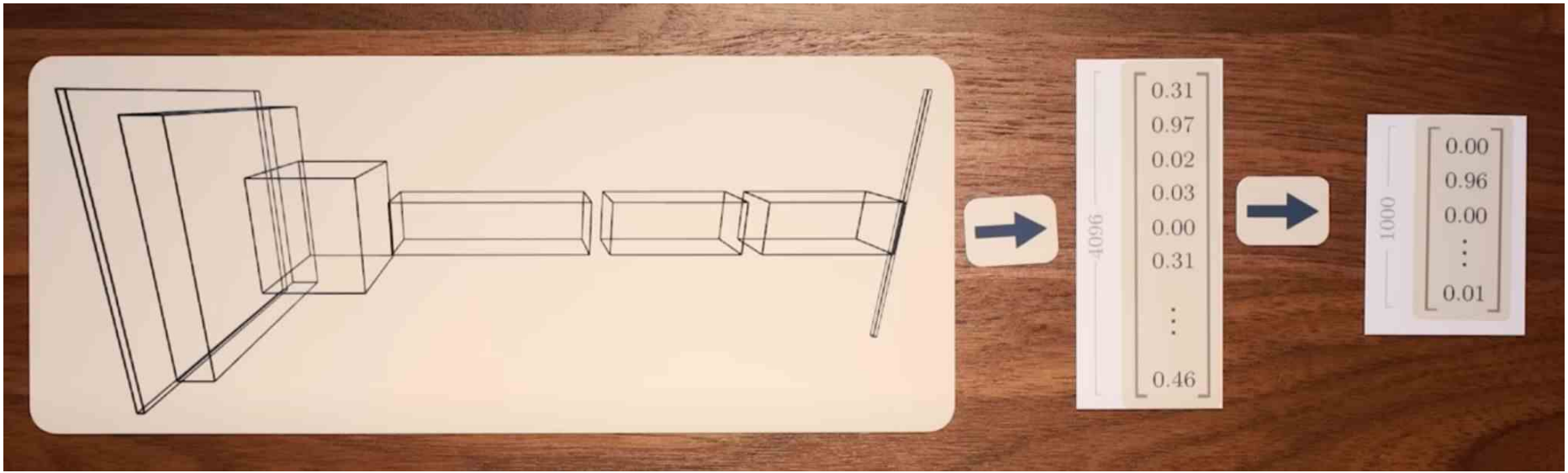


https://www.youtube.com/watch?v=UZDiGooFs54

Final two layers are special:

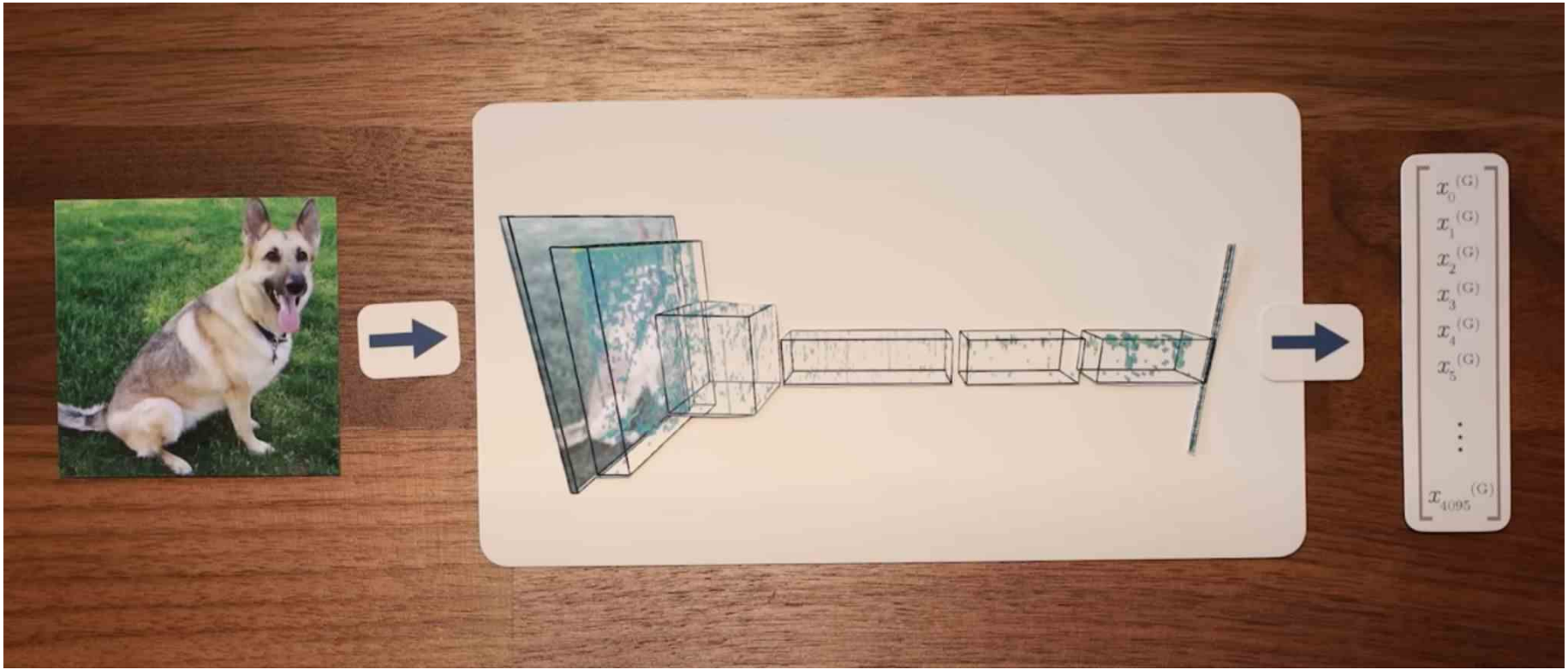
Last layer [1x1000]: Probability of the image in class i

Second last [1x4096]: Projection of image into 4096-dim space



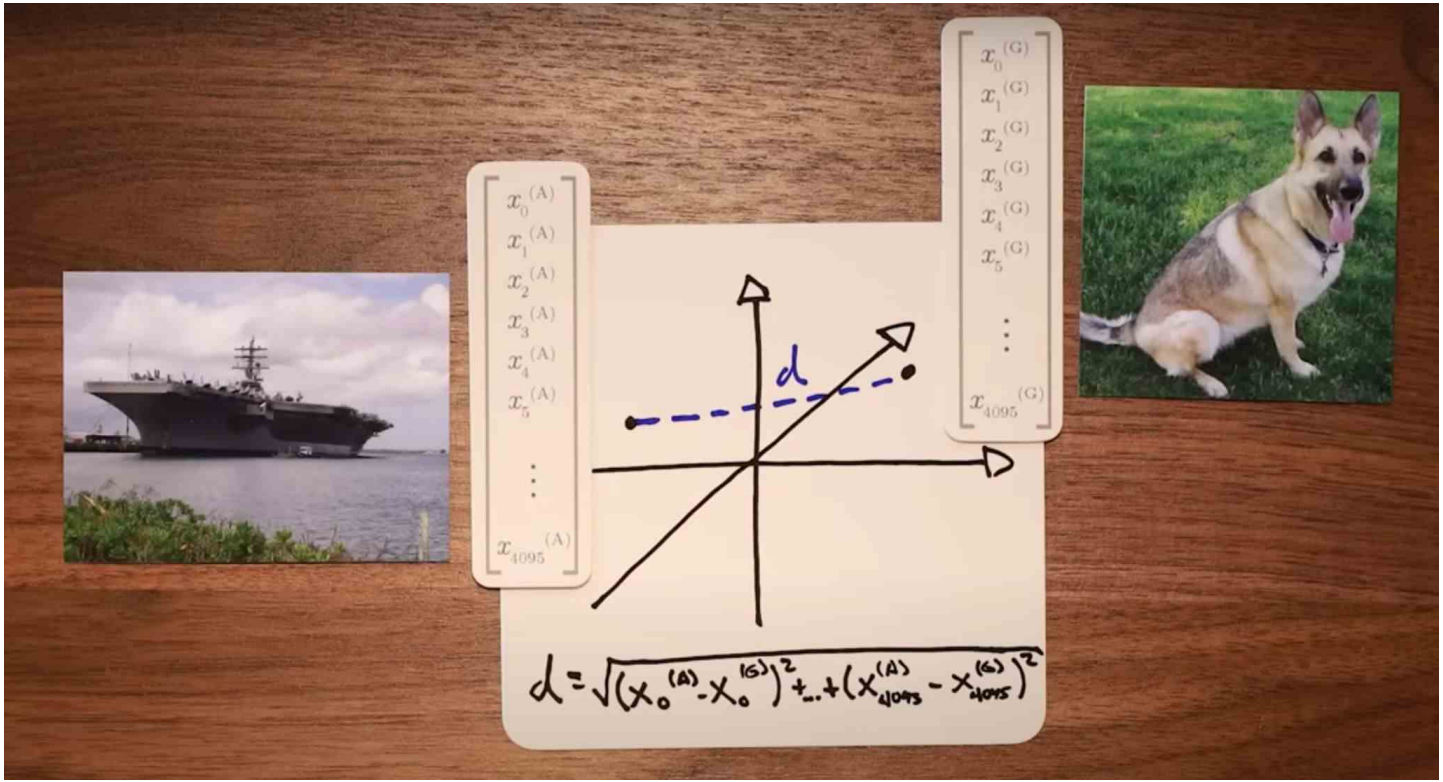
https://www.youtube.com/watch?v=UZDiGooFs54

Second to last forms an embedding of the input image



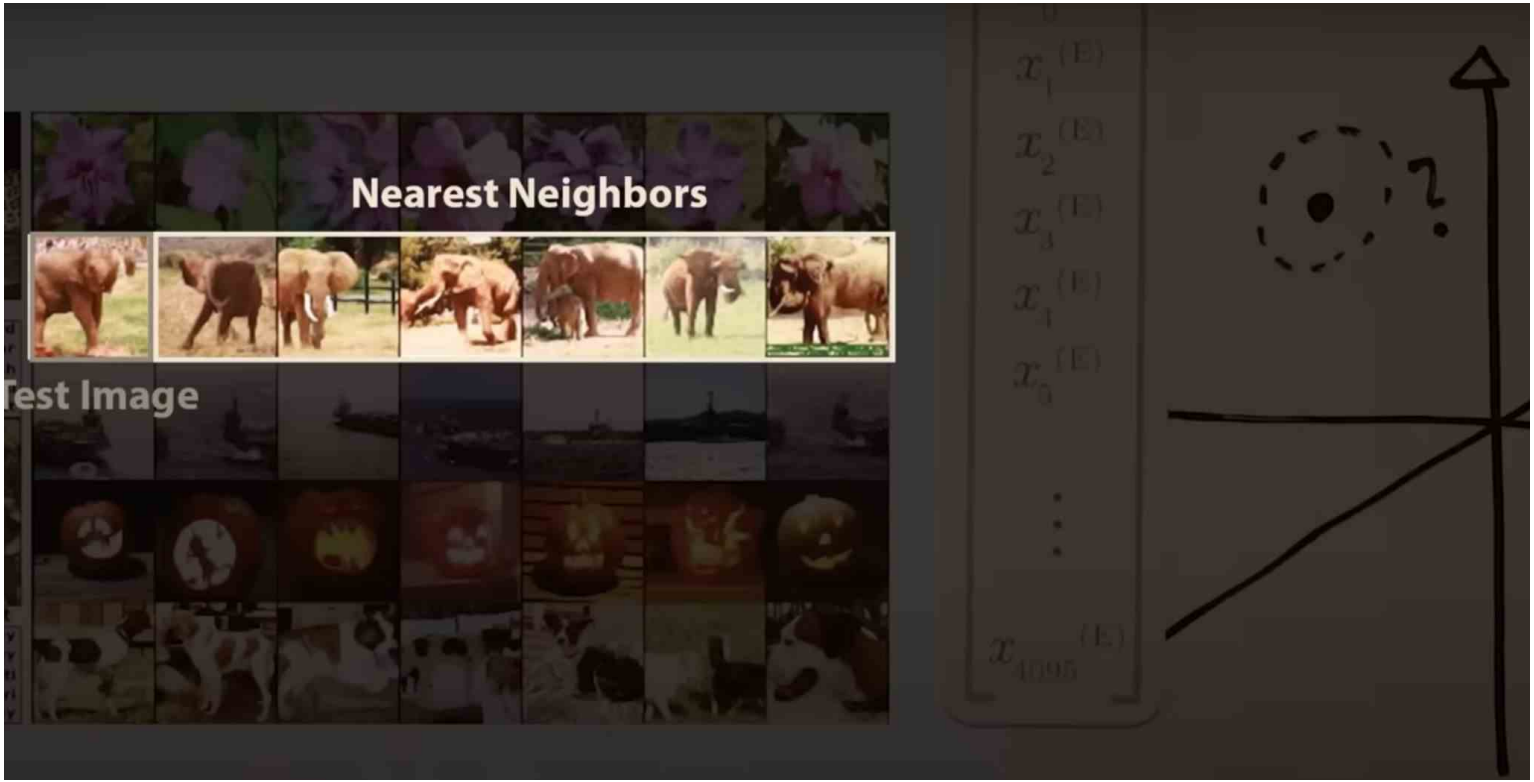
https://www.youtube.com/watch?v=UZDiGooFs54

Each image creates a unique vector
Compute the distance between vectors in 4096-dim space



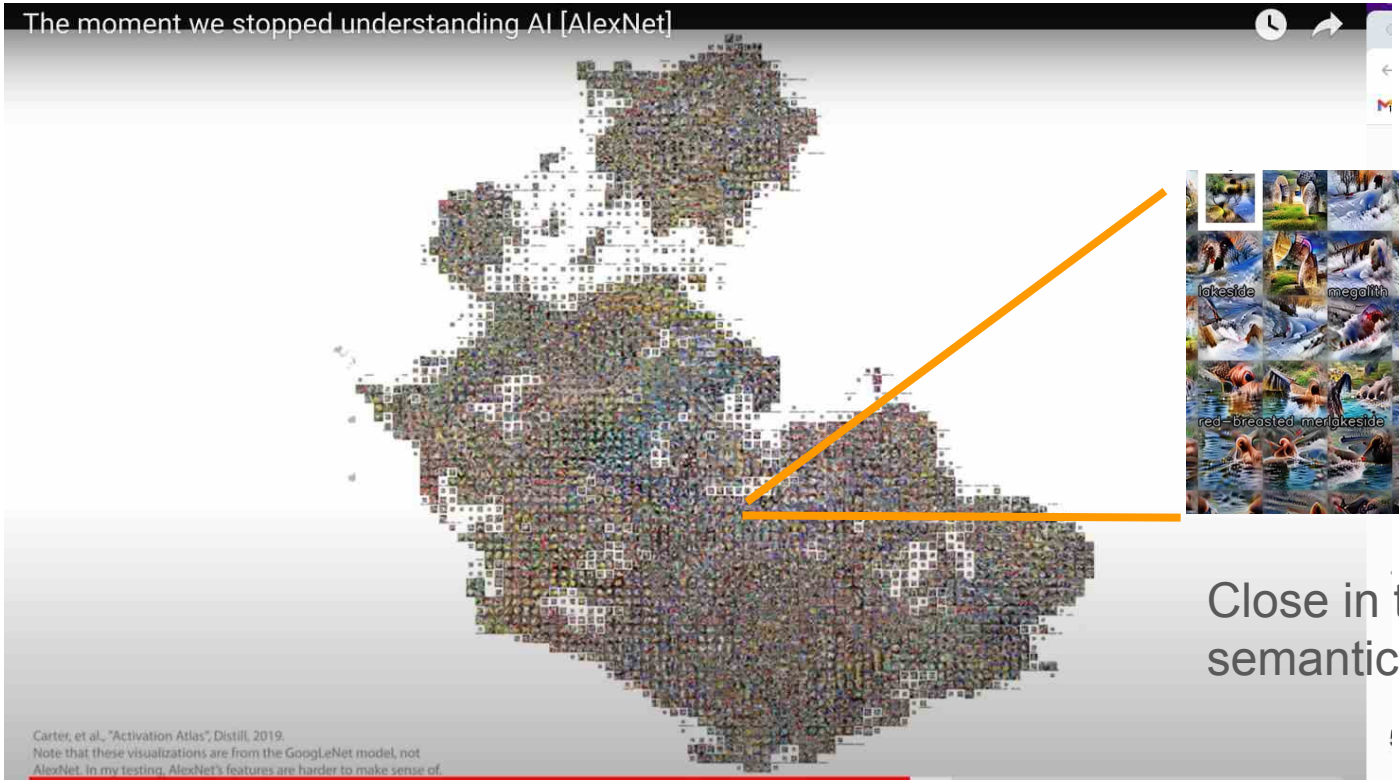
https://www.youtube.com/watch?v=UZDiGooFs54

Nearest neighbors in 4096-dim space are semantically related! Robust to object orientation, size, color, etc



https://www.youtube.com/watch?v=UZDiGooFs54

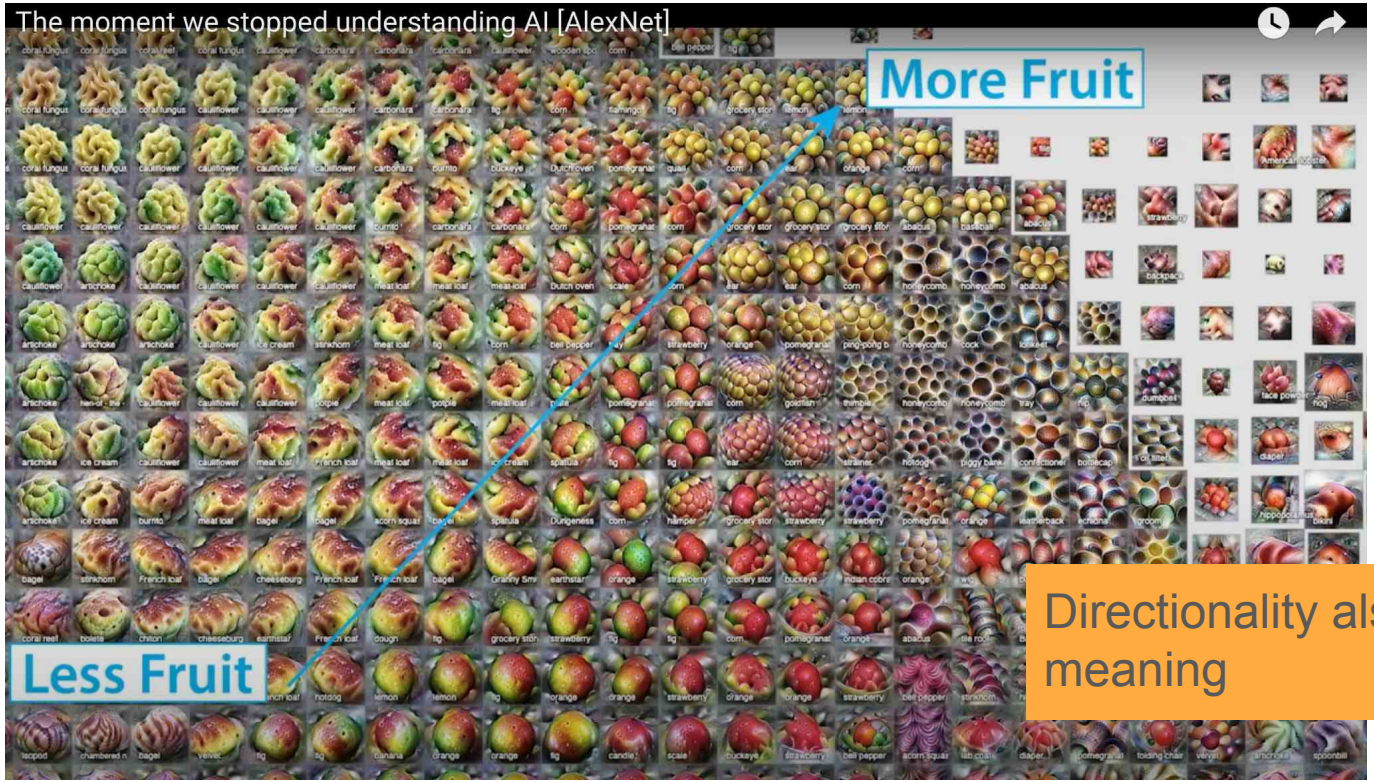
Activation Atlas: tSNE plot of images embeddings



Close in the projection are semantically related

https://www.youtube.com/watch?v=UZDiGooFs54

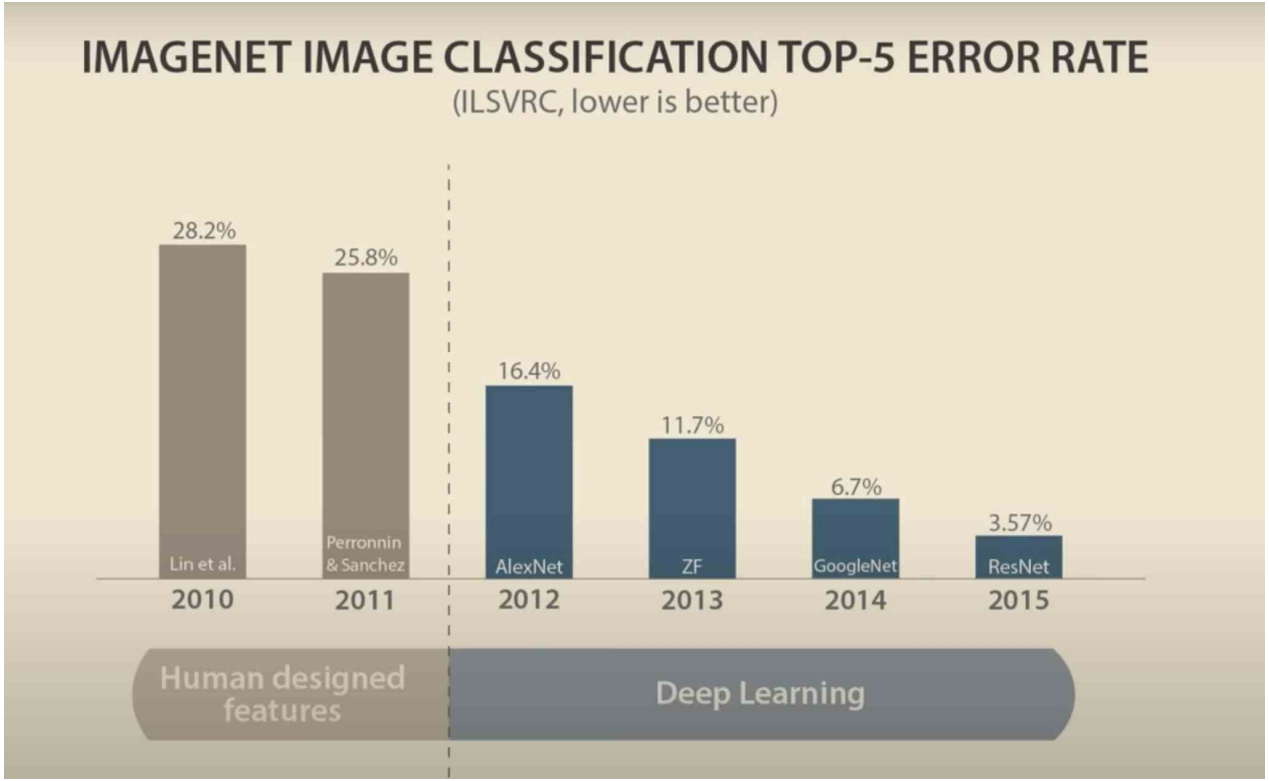
Activation Atlas: tSNE plot of images embeddings



Directionality also caries meaning

https://www.youtube.com/watch?v=UZDiGooFs54

Deep Learning Success!

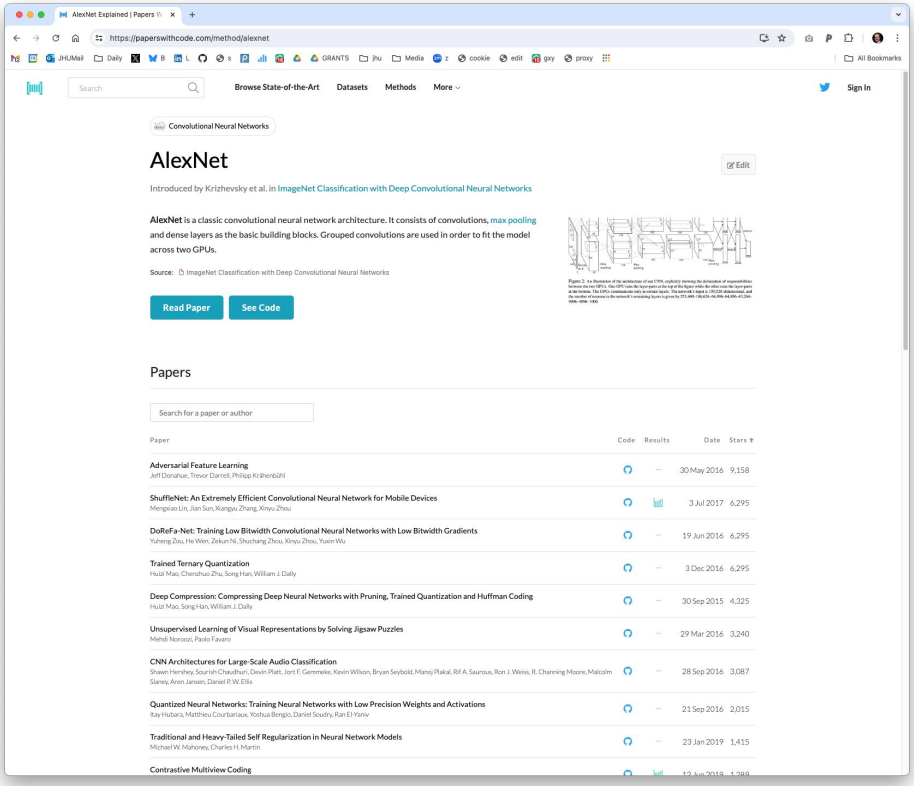


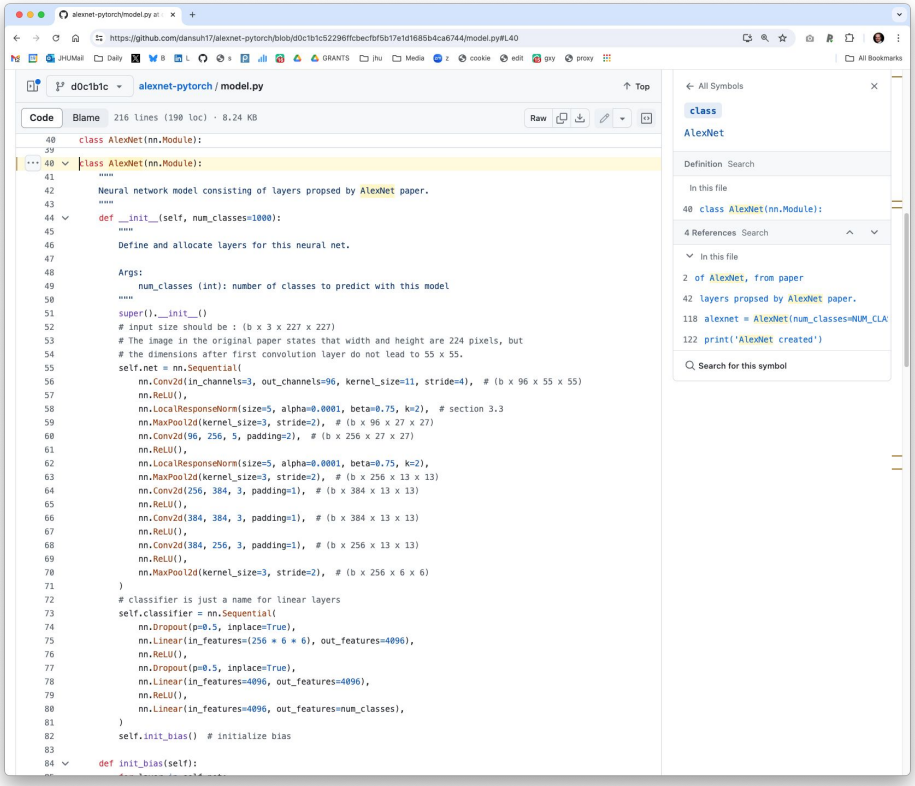
"Our results show that a large, deep convolutional neural network is capable of achieving record-breaking results on a highly challenging dataset using purely supervised learning. It is notable that our network's performance degrades if a single convolutional layer is removed. For example, removing any of the middle layers results in a loss of about 2% for the top-1 performance of the network. ***So the depth really is important for achieving our results.***"

https://www.youtube.com/watch?v=UZDiGooFs54

Outline

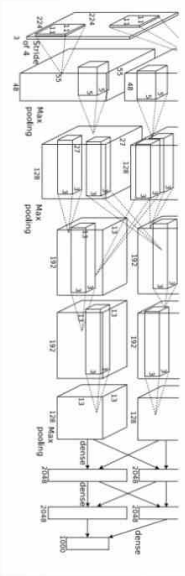
1. The Problem
2. pre-AlexNet approaches
3. How do animals & humans approach this?
4. Artificial Neural Networks
5. Convolutional Neural Networks
6. AlexNet
7. Impact





https://paperswithcode.com/method/alexnet

“AlexNet”



[Krizhevsky et al. NIPS 2012]

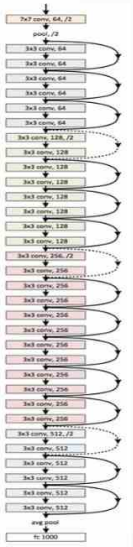
“GoogLeNet”

[Szegedy et al. CVPR 2015]

“VGG Net”

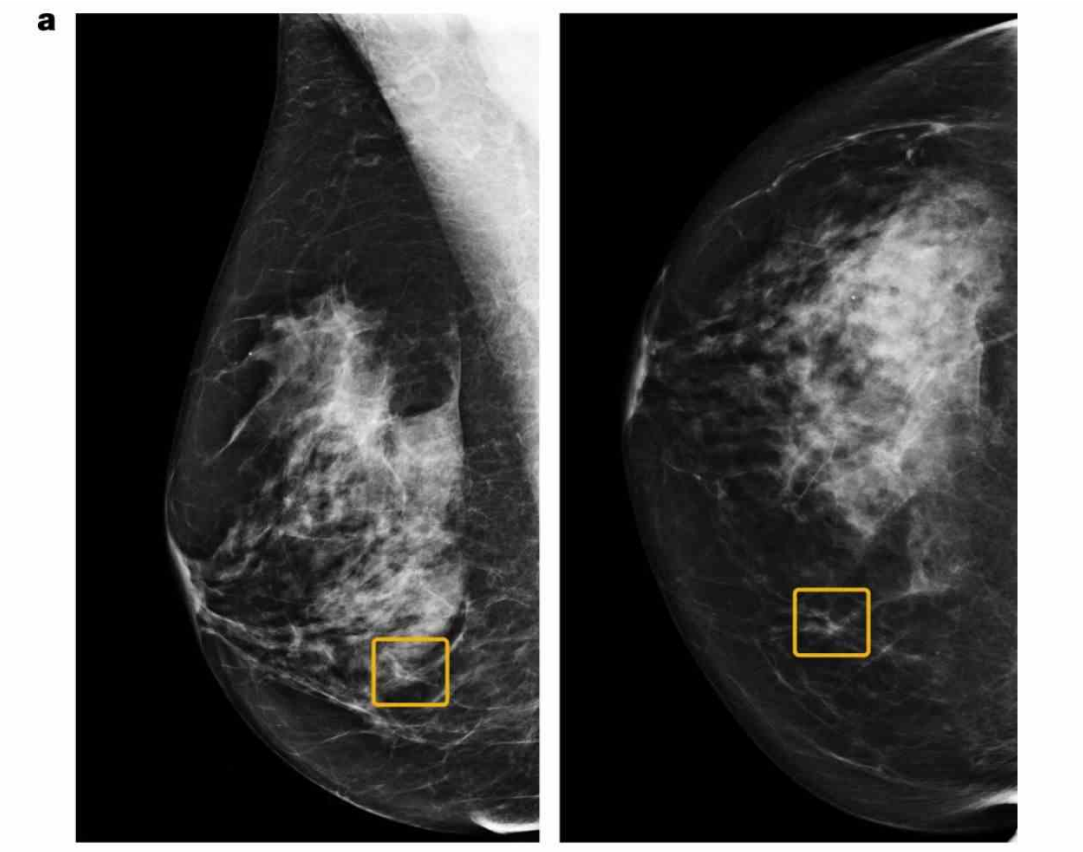
[Simonyan & Zisserman, ICLR 2015]

“ResNet”

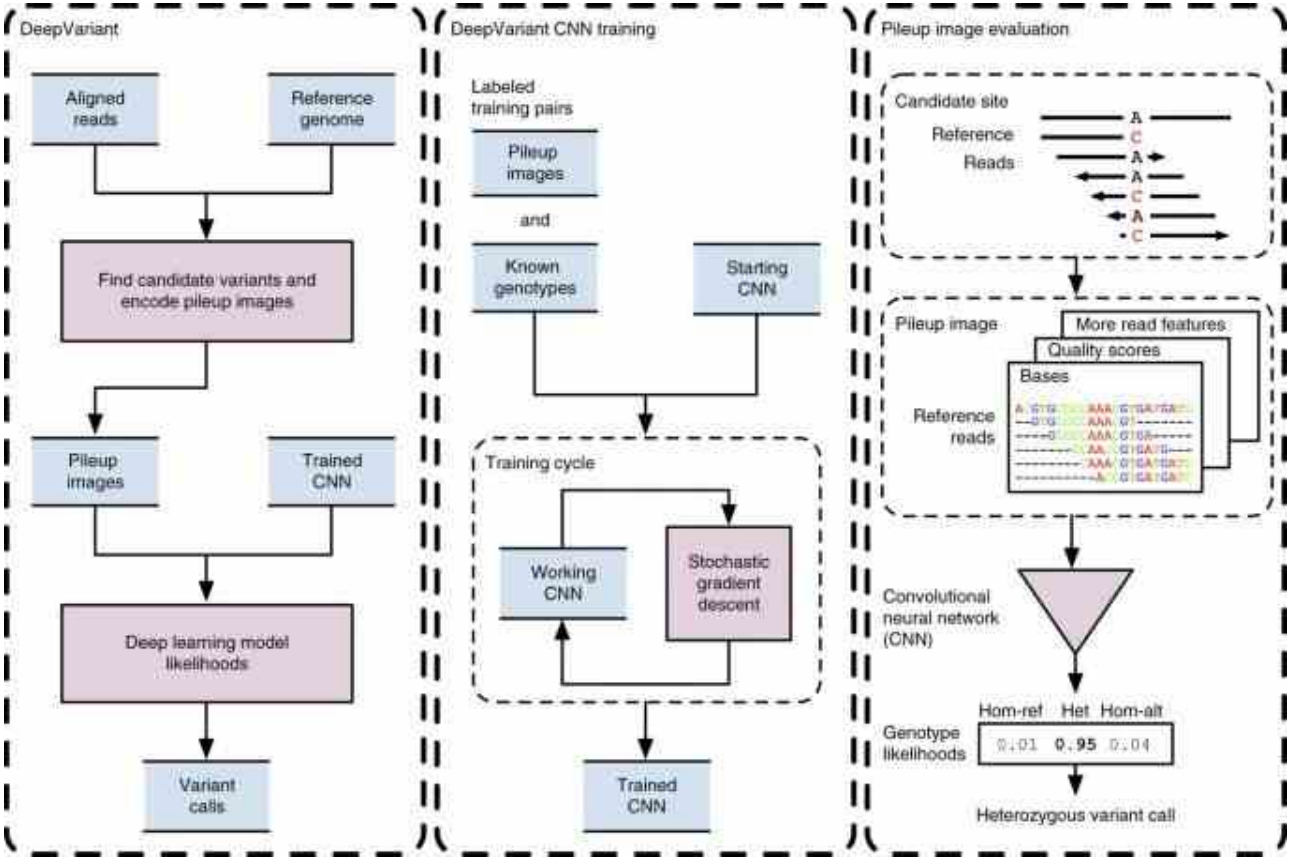


[He et al. CVPR 2016]

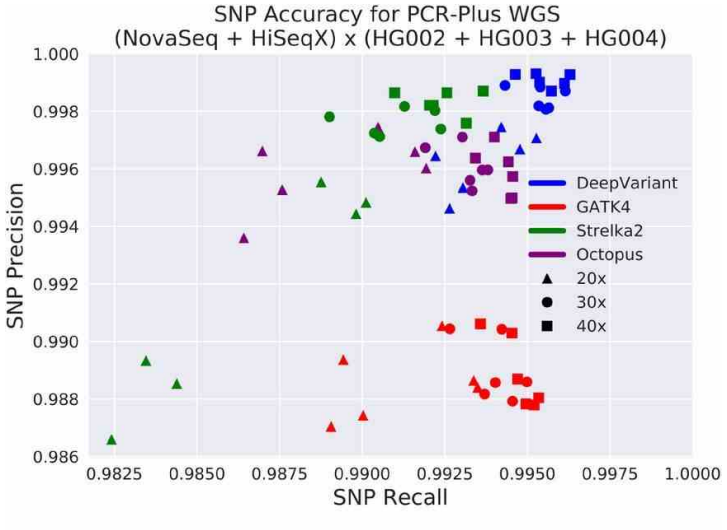
imagenet_ilsvrc2017_v1.0

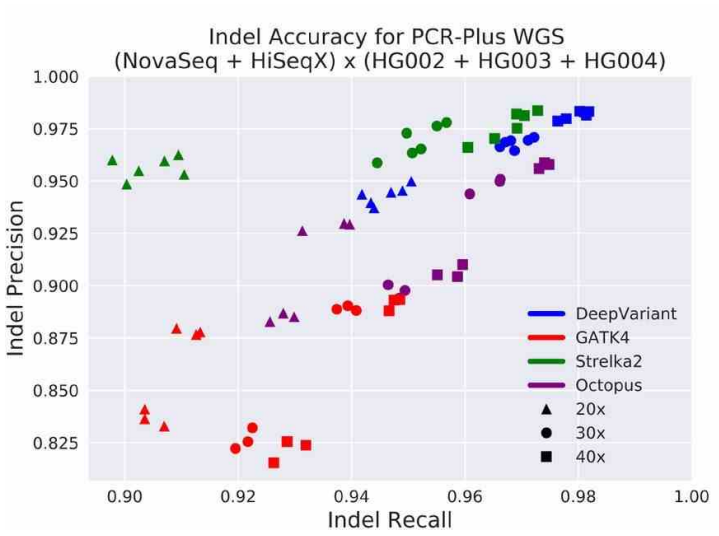


International evaluation of an AI system for breast cancer screening
McKinney et al (2020) Nature. https://doi.org/10.1038/s41586-019-1799-6



A universal SNP and small-indel variant caller using deep neural networks
Poplin et al (2018) Nature Biotechnology. https://doi.org/10.1038/nbt.4235





precisionFDA Challenges

An Extensive Sequence Dataset of Gold-Standard Samples for Benchmarking and Development

Baid et al (2020) bioRxiv

https://www.biorxiv.org/content/10.1101/2020.12.11.422022v1.full