

# Deep learning hands-on

## Lecture 1 Introduction

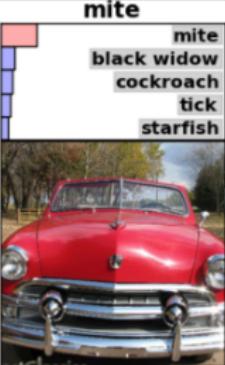
Ole Winther

Dept for Applied Mathematics and Computer Science  
Technical University of Denmark (DTU)



August 17, 2018

# ImageNet - image classification

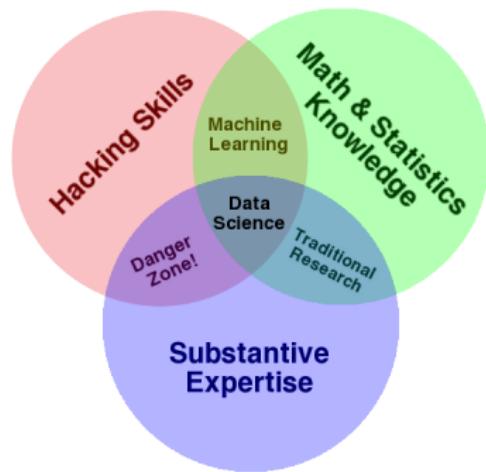
			
<b>mite</b> mite black widow cockroach tick starfish	<b>container ship</b> container ship lifeboat amphibian fireboat drilling platform	<b>motor scooter</b> motor scooter go-kart moped bumper car golfcart	<b>leopard</b> leopard jaguar cheetah snow leopard Egyptian cat
			
<b>grille</b> convertible grille pickup beach wagon fire engine	<b>mushroom</b> agaric mushroom jelly fungus gill fungus dead-man's-fingers	<b>cherry</b> dalmatian grape elderberry ffordshire bulterrier currant	<b>Madagascar cat</b> squirrel monkey spider monkey titi indri howler monkey

# Agenda

- Why the AI and deep learning hype
- Deep learning 101 - feed-forward neural networks
- Convolutional neural networks
- Biological applications of deep learning
- Hands-on labs!

# Agenda

- Why the AI and deep learning hype
- Deep learning 101 - feed-forward neural networks
- Convolutional neural networks
- Biological applications of deep learning
- Hands-on labs!
- A bit about myself



# Agenda (approximate timing)

09:15 Deep learning lecture

10:00 Tensorflow playground + break

10:30 Back-propagation lecture

10:45 Hands-on lab 1

12:00 Lunch

13:00 Convolutional neural networks (CNN) lecture

13:30 Hands-on lab 2

15:00 Break

15:15 Deep learning tricks of the trade

15:50 Ole's closing comments and questions

16:00 Hands-on lab 3.

# Computing - why we are here!

## 1 The accelerating pace of change ...



## 2 ... and exponential growth in computing power ...

Computer technology, shown here climbing dramatically by powers of 10, is now progressing more each hour than it did in its entire first 90 years

### COMPUTER RANKINGS

By calculations per second per \$1,000



#### Colossus

The electronic computer, with 1,500 vacuum tubes, helped the British crack German codes during WW II



#### UNIVAC I

The first commercially marketed computer, used to tabulate the U.S. Census, occupied 943 cu. ft.



#### Apple II

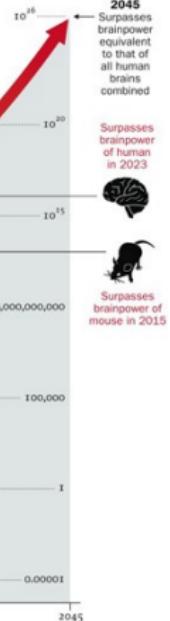
At a price of \$1,298, the compact machine was one of the first massively popular personal computers



#### Power Mac G4

The first personal computer to deliver more than 1 billion floating-point operations per second

## 3 ... will lead to the Singularity

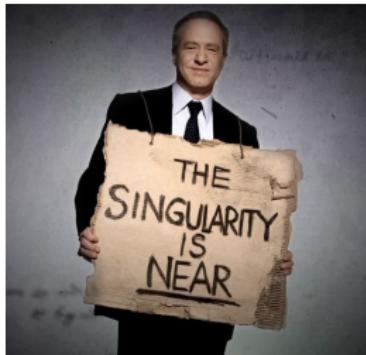


# Data - why we will have self-driving cars



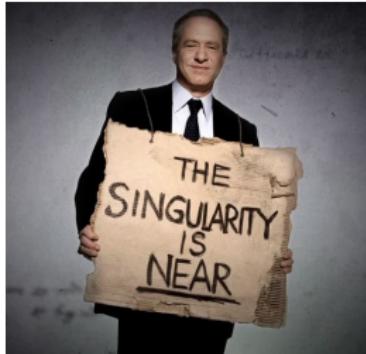
The Cityscapes dataset

# Are we heading towards the singularity?



kurzweilai.net

# Are we heading towards the singularity?



[kurzweilai.net](http://kurzweilai.net)



- Elon Musk at MIT AeroAstro Symp:
- If I were to guess at what our biggest existential threat is, it's probably that...
- With artificial intelligence, we are summoning the demon..
- Inofficial quotes (email to friend):
- The risk of something seriously dangerous happening is in the five year timeframe. 10 years at most,
- Unless you have direct exposure to groups like Deepmind, you have no idea how fast — it is growing at a pace close to exponential.

# Computing - reaching the singularity

## 1 The accelerating pace of change ...



## 2 ... and exponential growth in computing power ...

Computer technology, shown here climbing dramatically by powers of 10, is now progressing more each hour than it did in its entire first 90 years.

### COMPUTER RANKINGS

By calculations per second per \$1,000



#### Colossus

The electronic computer, with 1,500 vacuum tubes, helped the British crack German codes during WW II



#### UNIVAC I

The first commercially marketed computer, used to tabulate the U.S. Census, occupied 943 cu. ft.

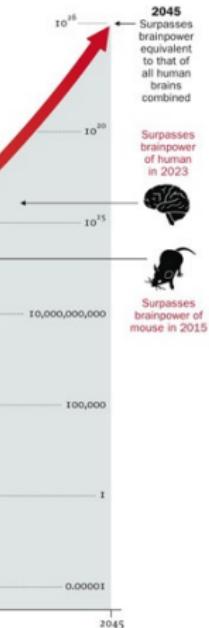


**Apple II**  
At a price of \$1,298, the compact machine was one of the first massively popular personal computers



**Power Mac G4**  
The first personal computer to deliver more than 1 billion floating-point operations per second

## 3 ... will lead to the Singularity



# Part 2: The deep learning revolution

# Achilles' heel of traditional AI: Perception in natural environment



xkcd.com/1425

# Major areas in Artificial intelligence (AI)

- Speech recognition
- Image classification
- Machine translation
- Question-answering
- Self-driving vehicles
- Dialogue systems
- General unsupervised learning

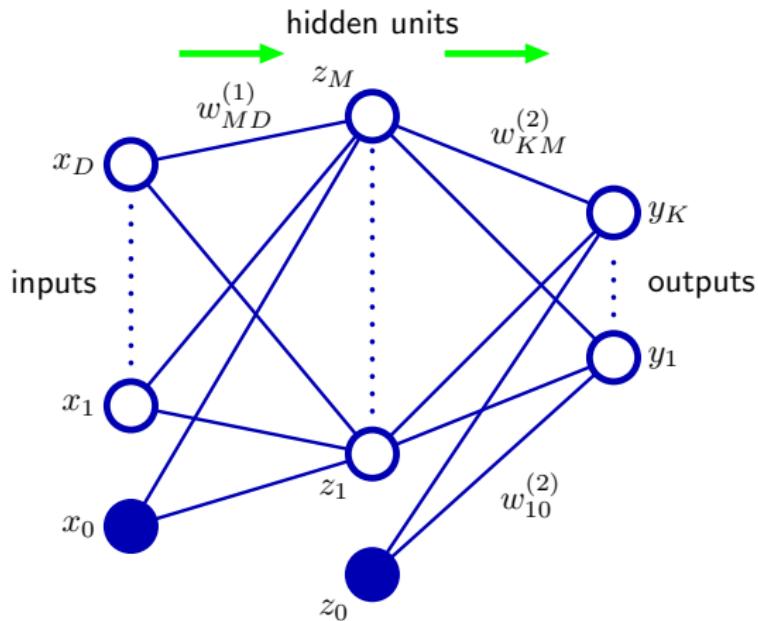


# Major areas in Artificial intelligence (AI)

- Speech recognition
- Image classification
- Machine translation
- Question-answering
- Self-driving vehicles
- Dialogue systems
- General unsupervised learning

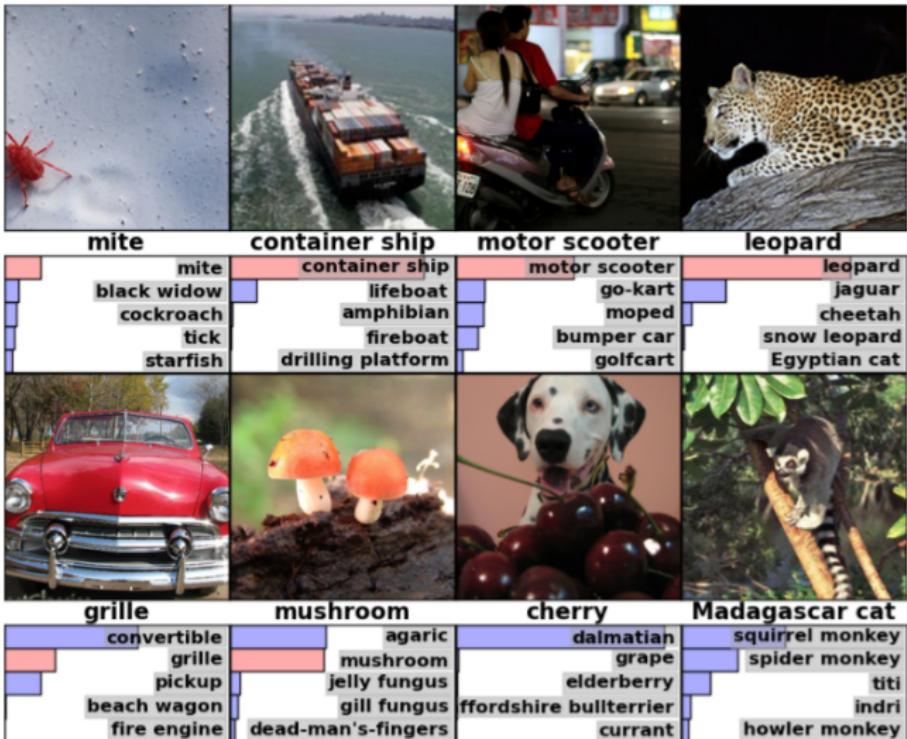


# Feed forward neural networks



$$y_k(\mathbf{x}, \mathbf{w}) = \sigma \left( \sum_{j=0}^M w_{kj}^{(2)} f \left( \underbrace{\sum_{i=0}^D w_{ji}^{(1)} x_i}_{z_j} \right) \right)$$

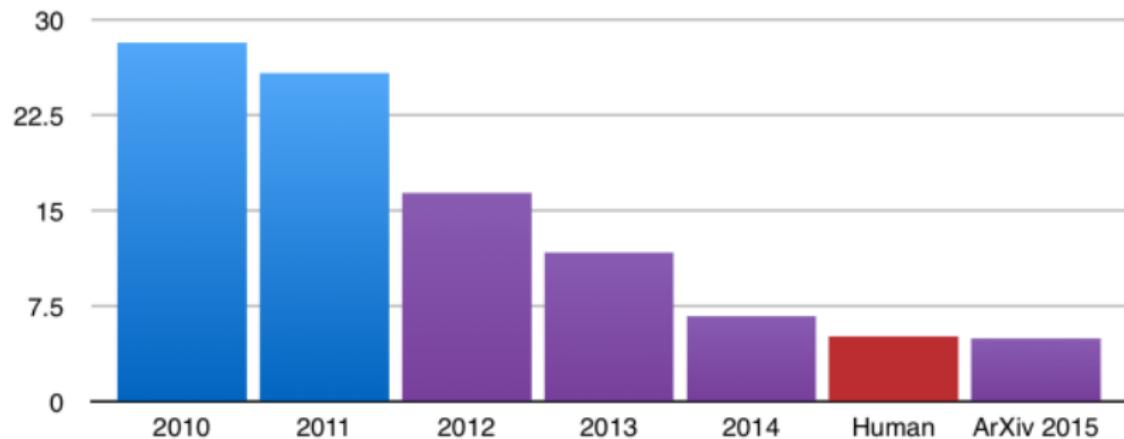
# ImageNet - image classification



- 1.000 different classes - including many types of dogs!
- 1.000.000 training images

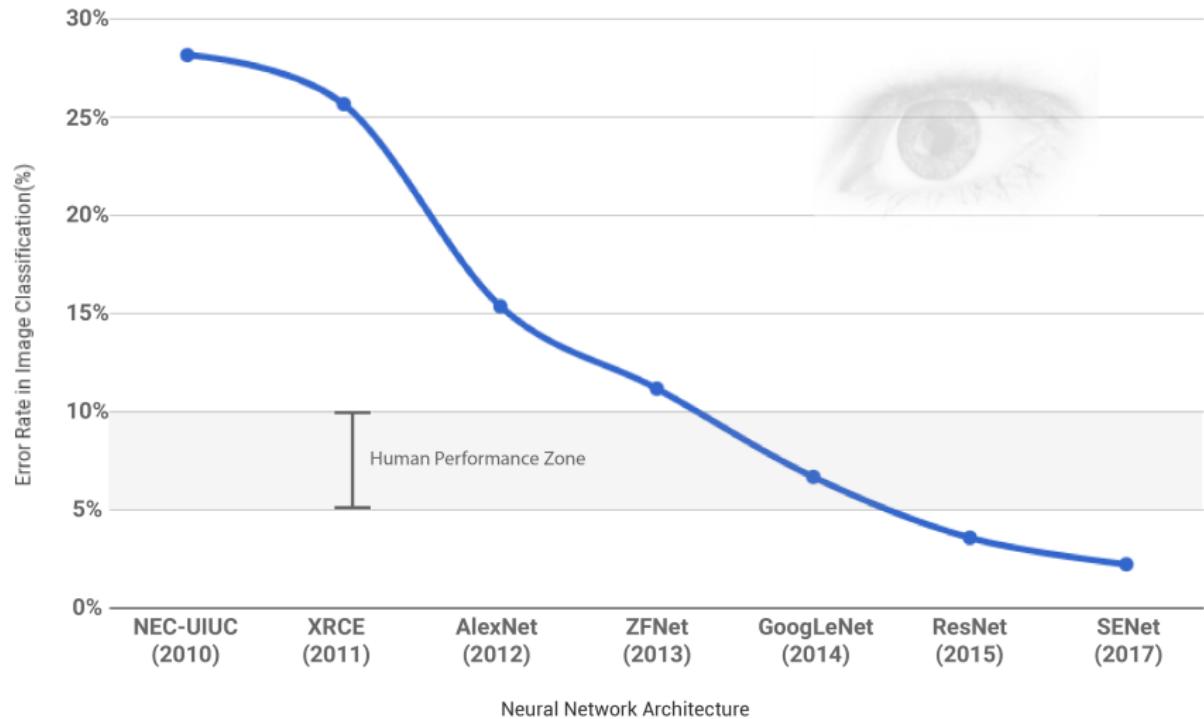
# ImageNet classification challenge

ILSVRC top-5 error on ImageNet



- AlexNet - A Krizhevsky et al. (2012) won with huge margin!
- Soon everyone started using **deep learning** and **GPUs**.

# ImageNet classification challenge 2017 update



# Feature engineering vs engineered models

## ImageNet Classification with Deep Convolutional Neural Networks

Alex Krizhevsky

University of Toronto

kriz@cs.utoronto.ca

Ilya Sutskever

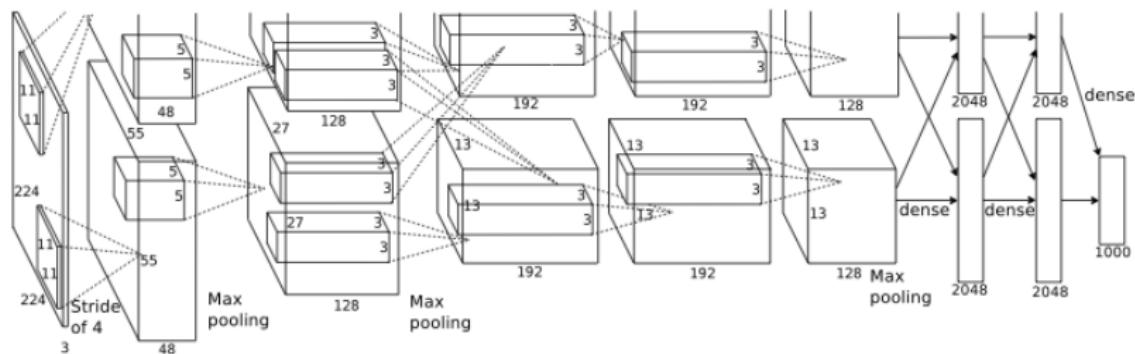
University of Toronto

ilya@cs.utoronto.ca

Geoffrey E. Hinton

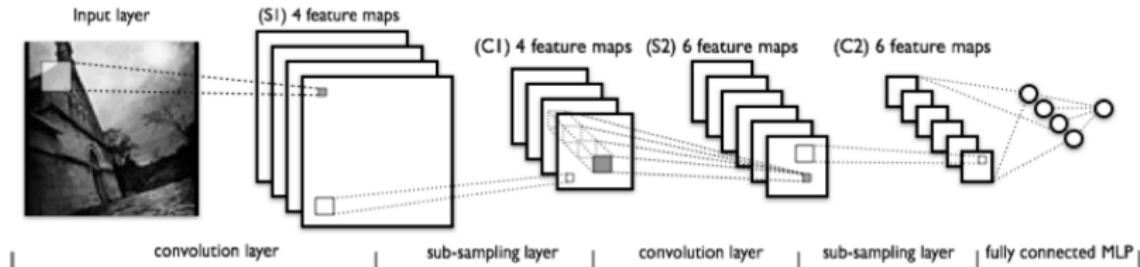
University of Toronto

hinton@cs.utoronto.ca



[www.cs.toronto.edu/~fritz/absps/imagenet.pdf](http://www.cs.toronto.edu/~fritz/absps/imagenet.pdf)

# Convolutional neural networks

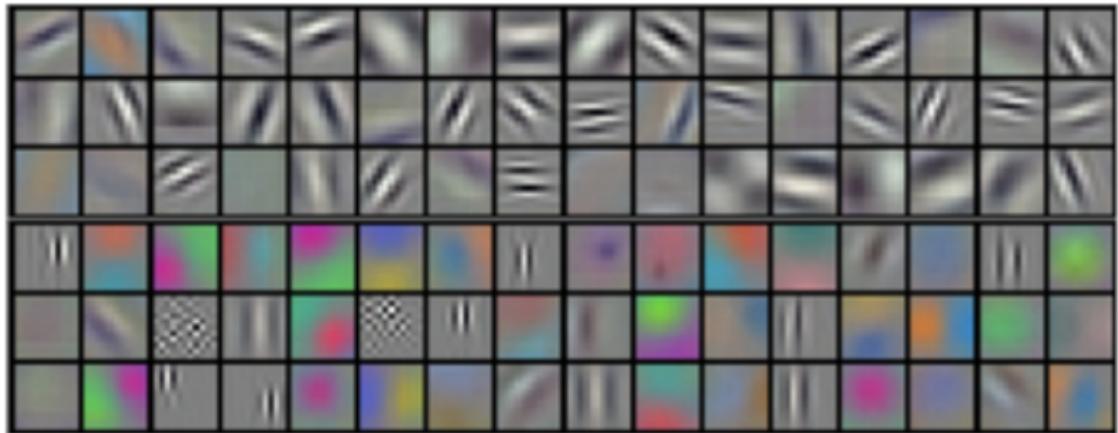


$$\begin{bmatrix} 10 & 0 & -10 \\ 0 & 0 & 0 \\ -10 & 0 & 10 \end{bmatrix}$$



# All filters are learned from training data

- First layer filters



- [www.cs.toronto.edu/~fritz/absps/imagenet.pdf](http://www.cs.toronto.edu/~fritz/absps/imagenet.pdf)

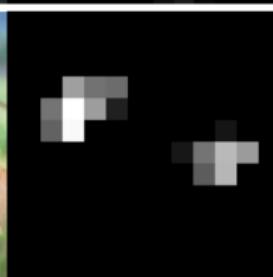
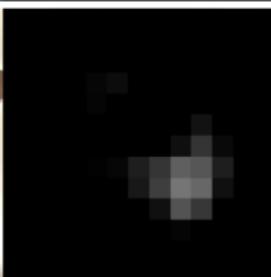
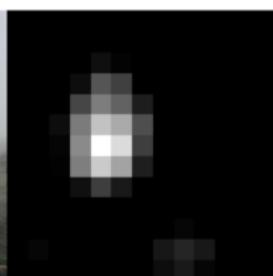
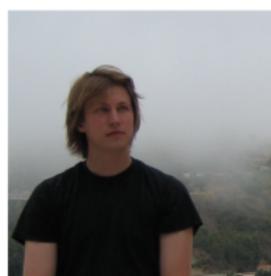
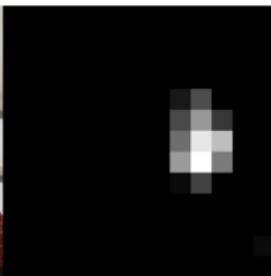
# Emergent higher level abstractions

- Look at output of filter in 5th layer!



# Emergent higher level abstractions

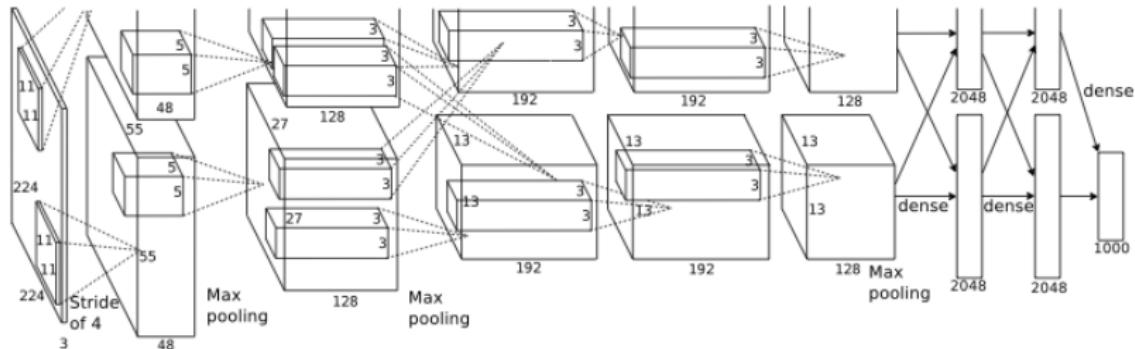
- Look at output of filter in 5th layer!



Yosinski et. al., ICML, google: deepvis

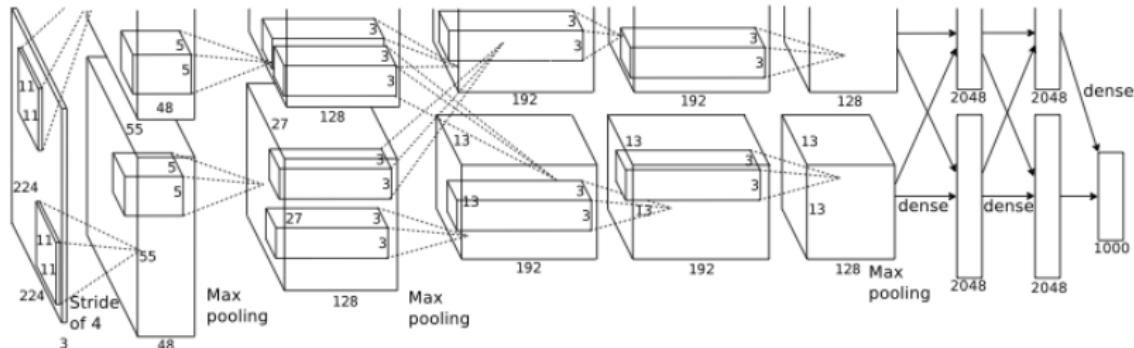
# We need bigger brains

- AlexNet (2012): 16.4% error, 8 layers, 1.4 Gflop

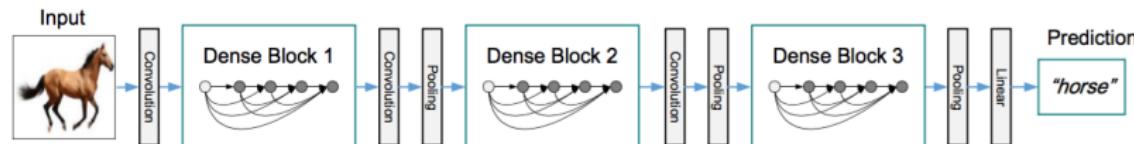


# We need bigger brains

- AlexNet (2012): 16.4% error, 8 layers, 1.4 Gflop

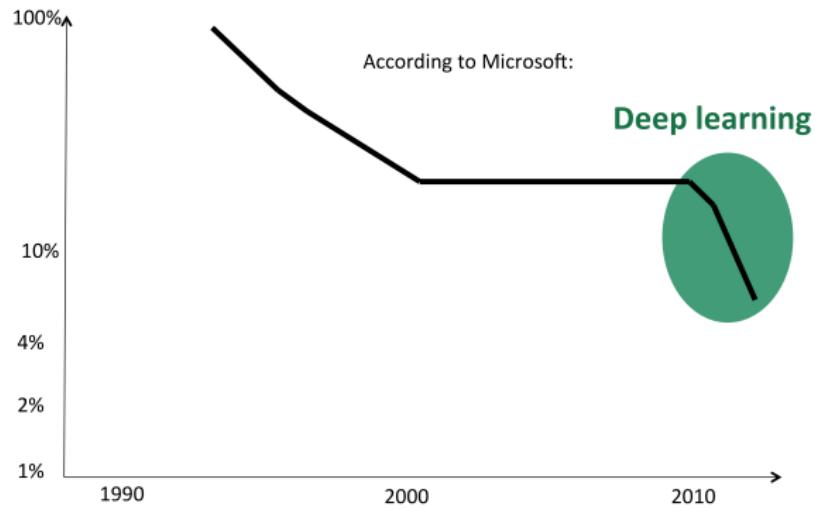


- ResNet (2016): 3.5% error, 152 layers, 22.6 Gflop.



- (This is a so-called DenseNet and not a ResNet.)
- Source: Source Jen-Hsun Huang, CEO NVIDIA, GTC Europe, 2016

# Speech recognition breakthrough



Plot from Yoshua Bengio

# Encoder-decoder - machine translation

---

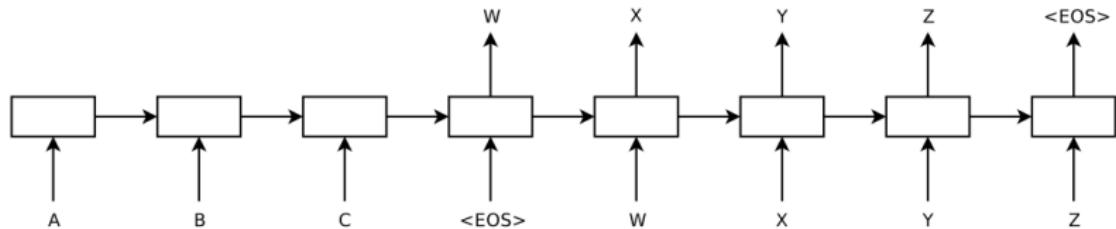
## Sequence to Sequence Learning with Neural Networks

---

Ilya Sutskever  
Google  
ilyasut@google.com

Oriol Vinyals  
Google  
vinyals@google.com

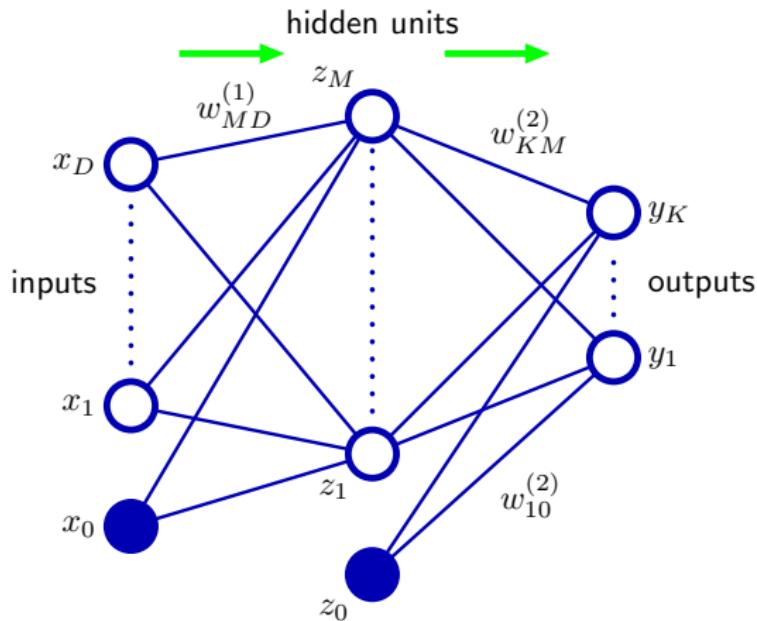
Quoc V. Le  
Google  
qvl@google.com



# Part 3:

# Feed-forward neural networks

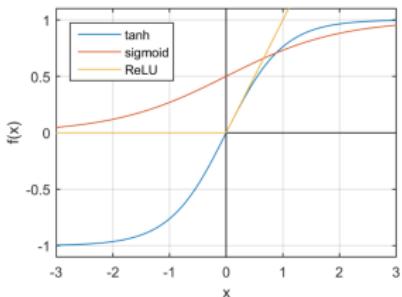
# Feed forward neural networks



$$y_k(\mathbf{x}, \mathbf{w}) = \sigma \left( \sum_{j=0}^M w_{kj}^{(2)} f \left( \underbrace{\sum_{i=0}^D w_{ji}^{(1)} x_i}_{z_j} \right) \right)$$

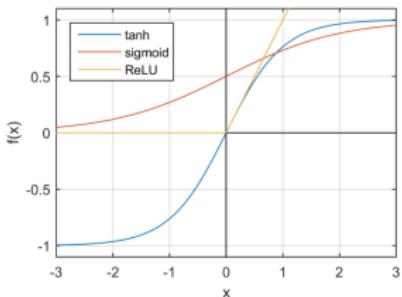
# Non-linearity and training

- Linear activation functions will give a linear network.
- Logistic function  $\sigma(a) = \frac{1}{1+e^{-a}}$
- Hyperbolic tangent  $\tanh(a) = \frac{e^a - e^{-a}}{e^a + e^{-a}}$
- Rectified linear  $\text{relu}(a) = \max(0, a)$



# Non-linearity and training

- Linear activation functions will give a linear network.
- Logistic function  $\sigma(a) = \frac{1}{1+e^{-a}}$
- Hyperbolic tangent  $\tanh(a) = \frac{e^a - e^{-a}}{e^a + e^{-a}}$
- Rectified linear  $\text{relu}(a) = \max(0, a)$



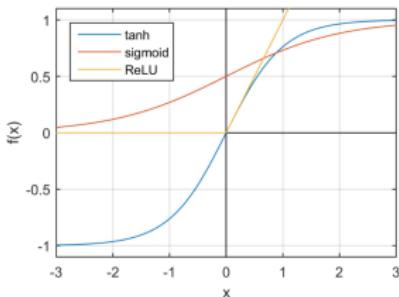
- Supervised learning
- Labeled training set

$$\mathcal{D} = \{(x_i, y_i) | i = 1, \dots, n\} .$$

- Input  $x_i$  and output  $y_i$ .

# Non-linearity and training

- Linear activation functions will give a linear network.
- Logistic function  $\sigma(a) = \frac{1}{1+e^{-a}}$
- Hyperbolic tangent  $\tanh(a) = \frac{e^a - e^{-a}}{e^a + e^{-a}}$
- Rectified linear  $\text{relu}(a) = \max(0, a)$



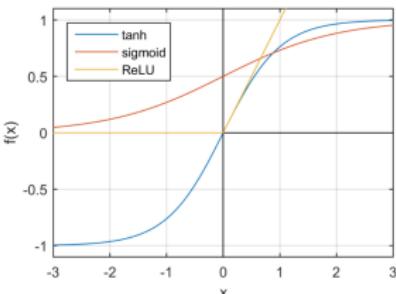
- Supervised learning
- Labeled training set

$$\mathcal{D} = \{(x_i, y_i) | i = 1, \dots, n\} .$$

- Input  $x_i$  and output  $y_i$ .
- Minimize training error by (stochastic) gradient descent

# Non-linearity and training

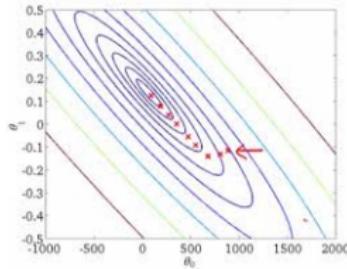
- Linear activation functions will give a linear network.
- Logistic function  $\sigma(a) = \frac{1}{1+e^{-a}}$
- Hyperbolic tangent  $\tanh(a) = \frac{e^a - e^{-a}}{e^a + e^{-a}}$
- Rectified linear  $\text{relu}(a) = \max(0, a)$



- Supervised learning
- Labeled training set

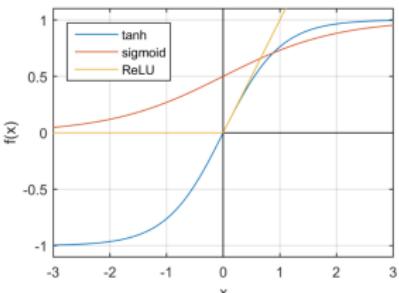
$$\mathcal{D} = \{(x_i, y_i) | i = 1, \dots, n\} .$$

- Input  $x_i$  and output  $y_i$ .
- Minimize training error by (stochastic) gradient descent



# Non-linearity and training

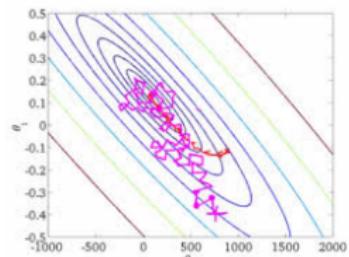
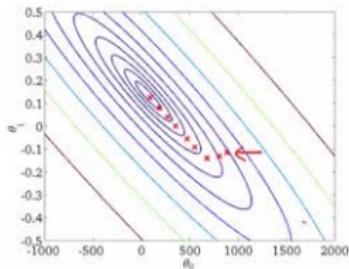
- Linear activation functions will give a linear network.
- Logistic function  $\sigma(a) = \frac{1}{1+e^{-a}}$
- Hyperbolic tangent  $\tanh(a) = \frac{e^a - e^{-a}}{e^a + e^{-a}}$
- Rectified linear  $\text{relu}(a) = \max(0, a)$



- Supervised learning
- Labeled training set

$$\mathcal{D} = \{(x_i, y_i) | i = 1, \dots, n\} .$$

- Input  $x_i$  and output  $y_i$ .
- Minimize training error by (stochastic) gradient descent



# Overfitting!



## Example: MNIST handwritten digits

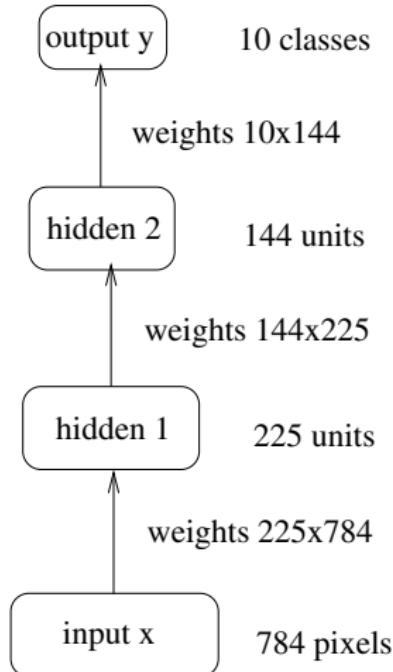
2	6	6	6	4
9	2	6	2	4
7	5	1	4	0
2	1	7	5	3

Train a network to classify  $28 \times 28$  images.

Data: 60000 input images  $\mathbf{x}(n)$  and labels  $y(n)$ .

Example model gives around 1.2% test error.

# Example Network



$$\mathbf{h}^{(3)} = \text{softmax}(\mathbf{W}^{(3)}\mathbf{h}^{(2)} + \mathbf{b}^{(3)})$$

$$\mathbf{h}^{(2)} = \text{relu}(\mathbf{W}^{(2)}\mathbf{h}^{(1)} + \mathbf{b}^{(2)})$$

$$\mathbf{h}^{(1)} = \text{relu}(\mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)})$$

$$\text{softmax}(\mathbf{z})_i = \frac{\exp(z_i)}{\sum_j \exp(z_j)}$$

$$\text{relu}(z) = \max(0, z)$$

# Softmax

- Softmax function

$$\text{softmax}(\mathbf{z})_i = \frac{\exp(z_i)}{\sum_j \exp(z_j)}$$

has two nice properties:

- $\text{softmax}(\mathbf{z})_i \geq 0$
- $\sum_i \text{softmax}(\mathbf{z})_i = 1$

# Softmax

- Softmax function

$$\text{softmax}(\mathbf{z})_i = \frac{\exp(z_i)}{\sum_j \exp(z_j)}$$

has two nice properties:

- $\text{softmax}(\mathbf{z})_i \geq 0$
- $\sum_i \text{softmax}(\mathbf{z})_i = 1$
- MNIST, output labels: 0, 1, ..., 9.
- Output of network

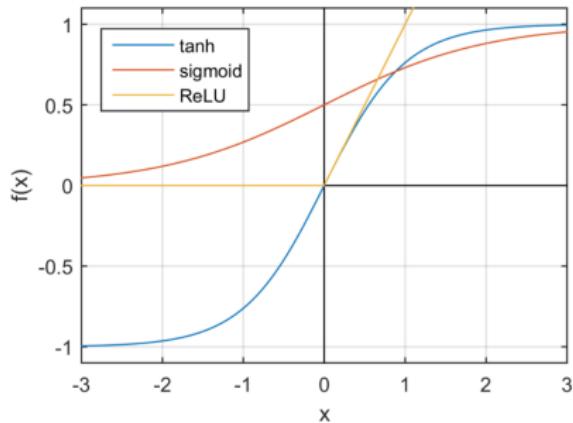
$$\mathbf{h}^{(3)} = \text{softmax}(\mathbf{W}^{(3)}\mathbf{h}^{(2)} + \mathbf{b}^{(3)})$$

interpreted as class(-conditional) probabilities:

- For example:

$$p(\text{digit} = 5 | \mathbf{x}) = \mathbf{h}_6^{(3)}$$

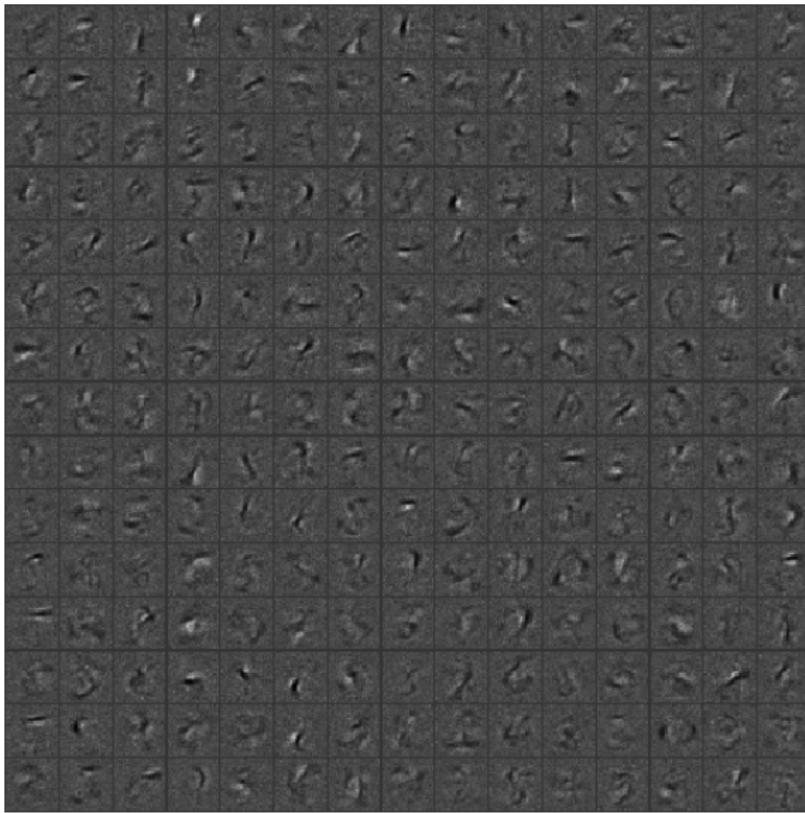
# On activation functions



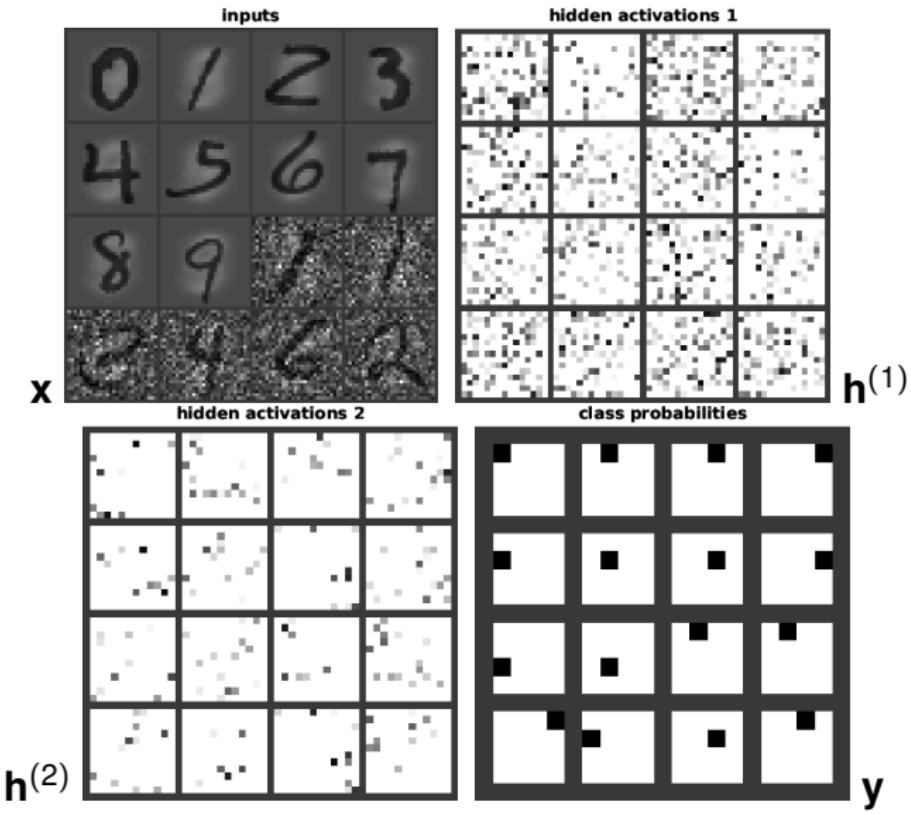
- $\text{relu}(z) = \max(0, z)$  is replacing old sigmoid and tanh.
- Note that identity function would lead into:

$$\begin{aligned}\mathbf{h}^{(2)} &= \mathbf{W}^{(2)}\mathbf{h}^{(1)} + \mathbf{b}^{(2)} \\ &= \mathbf{W}^{(2)}(\mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)}) + \mathbf{b}^{(2)} \\ &= (\mathbf{W}^{(2)}\mathbf{W}^{(1)})\mathbf{x} + (\mathbf{W}^{(2)}\mathbf{b}^{(1)} + \mathbf{b}^{(2)}) \\ &= \mathbf{W}'\mathbf{x} + \mathbf{b}'\end{aligned}$$

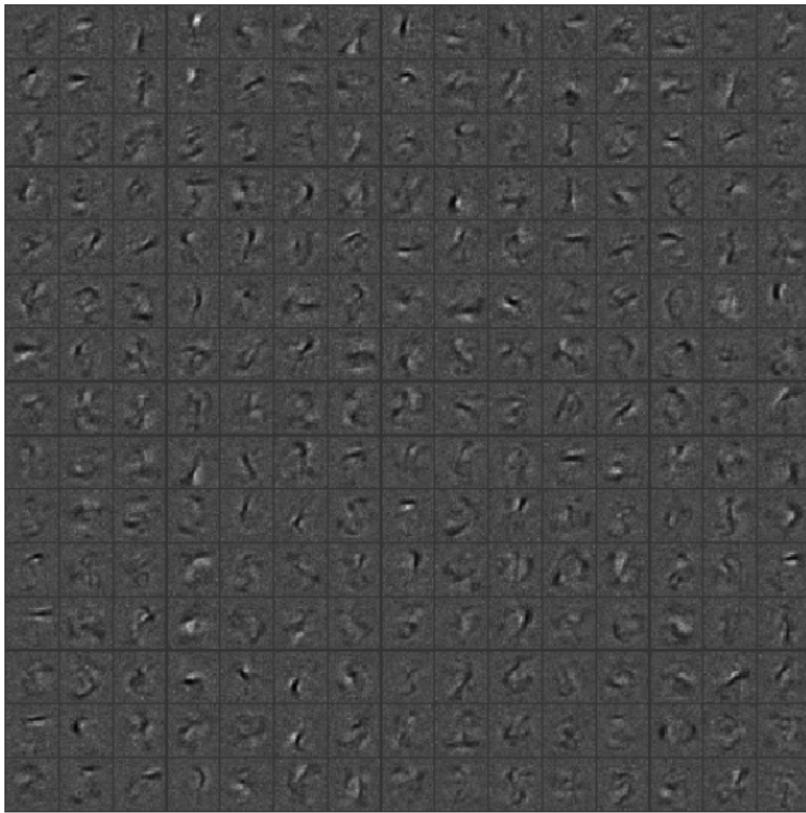
Weight matrix  $\mathbf{W}^{(1)}$  size  $225 \times 784$



Signals  $\mathbf{x} \rightarrow \mathbf{h}^{(1)} \rightarrow \mathbf{h}^{(2)} \rightarrow \mathbf{h}^{(3)}$



Weight matrix  $\mathbf{W}^{(1)}$  size  $225 \times 784$

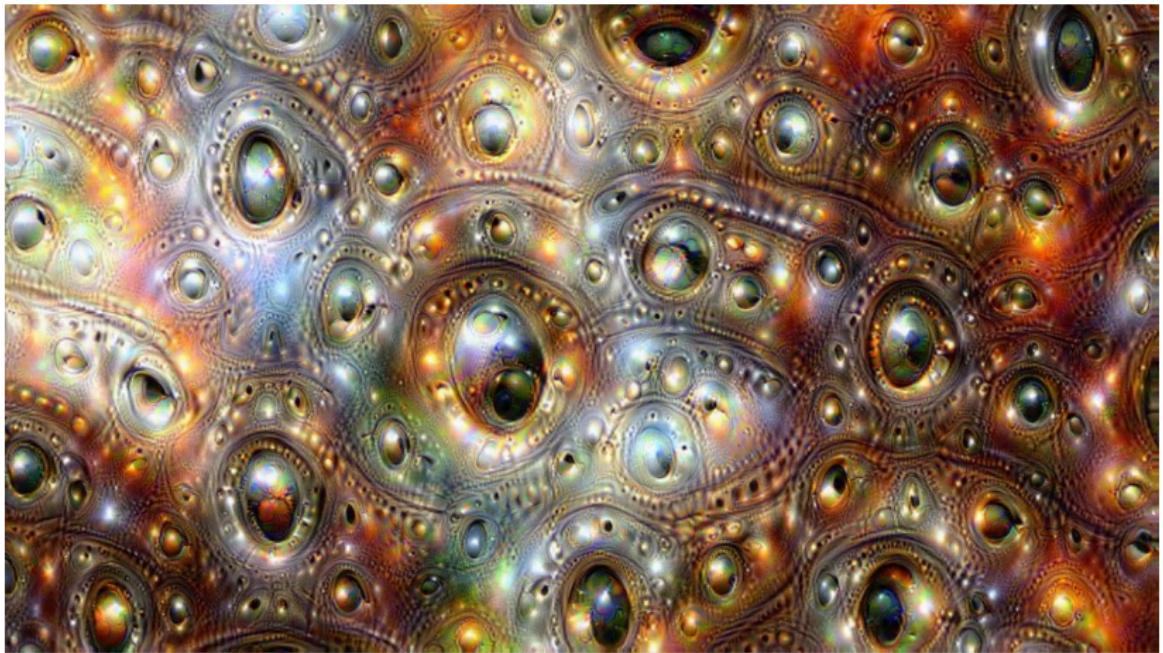


# TensorFlow Playground

[playground.tensorflow.org](http://playground.tensorflow.org)

# References

- Online book: Michael Nielsen, Neural networks and deep learning
- Book also online: Ian Goodfellow and Yoshua Bengio and Aaron Courville, Deep learning
- LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton, Deep learning, Nature 521.7553 (2015): 436-444.
- Mnih, Volodymyr, et al., Human-level control through deep reinforcement learning, Nature 518.7540 (2015): 529-533.
- Alipanahi, Babak, et al., Predicting the sequence specificities of DNA-and RNA-binding proteins by deep learning, Nature biotechnology (2015).
- Silver, David, et al., Mastering the game of Go with deep neural networks and tree search, Nature 529.7587 (2016): 484-489.
- Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhutdinov, R., Zemel, R., & Bengio, Y. (2015), Show, attend and tell: Neural image caption generation with visual attention. arXiv preprint arXiv:1502.03044.
- Mansimov, Elman, et al., Generating Images from Captions with Attention. arXiv preprint arXiv:1511.02793 (2015).
- Larsen, Anders Boesen Lindbo, Soren Kaae Sonderby, and Ole Winther. Autoencoding beyond pixels using a learned similarity metric. arXiv preprint arXiv:1512.09300 (2015).
- Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep Learning. Book in preparation for MIT Press, <http://goodfeli.github.io/dlbook/>, 2016.
- Michael Nielsen, Neural Networks and Deep Learning, <http://neuralnetworksanddeeplearning.com/>
- Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton, ImageNet Classification with Deep Convolutional, NIPS, 2012. Neural Networks,



Thanks!  
Ole Winther