A Gentle Introduction to Machine Learning

Third Lecture

Deep Learning – A Closer Look

Olov Andersson, AIICS

LINKÖPINGS UNIVERSITET

Outline of the Deep Learning Lecture

- What is deep learning
- Some motivation
- Enablers
  - Data
  - Computation
  - Training Algorithms & Tools
  - Network Architectures
- Closing examples

2

AI In The News Lately

- *"The development of full artificial intelligence could spell the end of the human race … it would take off on its own, and re-design itself at an ever increasing rate.* ***Humans****, who are limited by slow biological evolution, couldn't compete, and* ***would be superseded****."* *–* ***Stephen Hawking***
- *"I think we should be very careful about* ***artificial intelligence****. If I had to guess at what our biggest* ***existential threat*** *is, I'd probably say that. So we need to be very careful."* *–* ***Elon Musk***
- *"Artificial intelligence is the future, not only for Russian, but for all of humankind. It comes with colossal opportunities, but also threats that are difficult to predict.* ***Whoever becomes the leader in this sphere will become the ruler of the world****."* *–* ***Vladimir Putin***

There is **a lot of hype** about the capabilities of AI, mainly driven by recent advances in **deep learning**

But Deep Learning Is Not All Hype

No threat to humanity in sight, but **impressive applications**…

- **Google**: "1000 deep learning projects"
  - Extending across search, Android, Gmail, photo, maps, translate, YouTube, and self-driving cars. In 2014 it bought DeepMind, whose deep reinforcement learning project, AlphaGo, defeated the world's Go champion.
- **Microsoft**
  - Speech-recognition products (e.g. Bing voice search, X-Box voice commands), search rankings, photo search, translation systems, and more.
- **Facebook**
  - Uses DL to translate about 2 billion user posts per day in more than 40 languages (About half its community does not speak English.)
- **Baidu** (China's Google)
  - Uses DL for speech recognition, translation, photo search, and a self-driving car project, among others.

Source: Fortune.com

Rapid progress, hardly a day without some new application

The State of Deep Learning

State-of-the-art results in:
- Computer vision (e.g. object detection)
- Natural language processing (e.g. translation)
- Speech recognition/synthesis

Promising results:
- Robotics
- Content generation

Real-world applications are mainly in **supervised learning**, deep reinforcement and unsupervised learning are still less mature

So, what is deep learning?!

General Approach For ”AI scale” Real-world ML

In particular excels at
- Human modalities
- Eg. image, text and speech domains
  - Recognition, segmentation, translation and generation

”Traditional” machine learning

Hand-crafted Feature Extractor → “Simple” Trainable Classifier →

”Deep” machine learning

Trainable Feature Extractor *and* Classifier →

(NVIDIA)

What Learning Algorithm Goes In The Box?

Why did we want feature extraction in the first place?
- Remember the **limitations and pitfalls** of supervised learning
- **Curse of dimensionality:** Input dimension increases data requirements, *worst-case* exponentially
- If we can also **learn** feature extractors, we can get around this

E.g, $y = f_{\text{classifier}}(g_{\text{features}}(D))$, want to learn both $f()$ and $g()$ from raw data D
- Must be a powerful model, ideally able to approximate arbitrary functions f and g...
- Want something that can learn compositions of functions *f(g(...))*, like layers...

7

What Learning Algorithm Goes In The Box?

Why did we want feature extraction in the first place?
- Remember the **limitations and pitfalls** of supervised learning
- **Curse of dimensionality:** Input dimension increases data requirements, *worst-case* exponentially
- If we can also **learn** feature extractors, we can get around this

E.g, $y = f_{\text{classifier}}(g_{\text{features}}(D))$, want to learn both $f()$ and $g()$ from raw data D
- Must be a powerful model, ideally able to approximate arbitrary functions f and g...
- Want something that can learn compositions of functions *f(g(...))*, like layers...

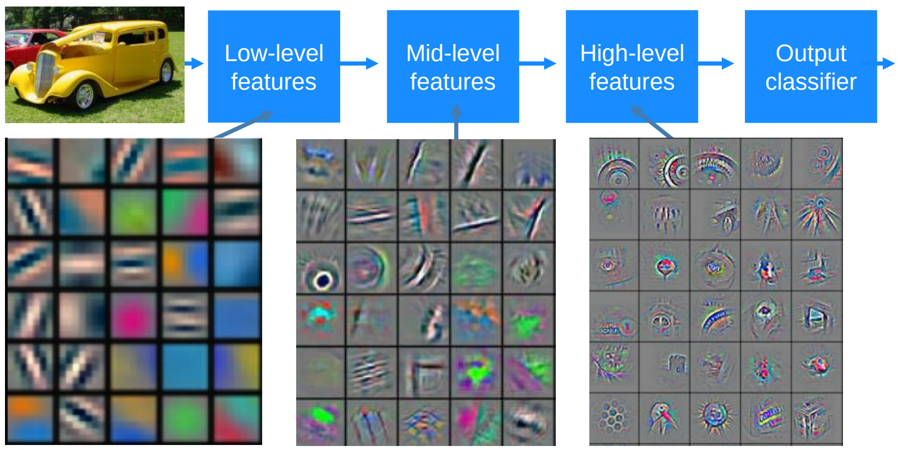
Multi-layer Neural Networks is by far the most common choice

8

Deep learning = Learning Hierarchical Representations

It's deep if it has more than one stage of non-linear feature transformation

- **More than two layers** of abstractions *f(g())* might be even better?



Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

(NVIDIA)

9

Many Problems Appear Naturally Hierarchical

Image recognition

- Pixel → edge → texton → motif → part → object

Text

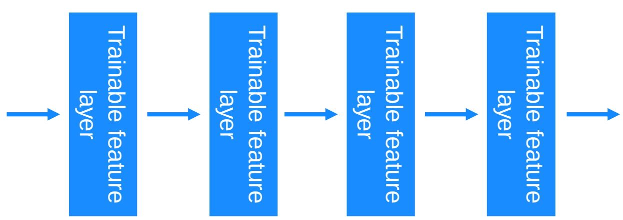
- Character → word → word group → clause → sentence → story

Speech

- Sample → spectral band → sound → … → phone → phoneme → word

Want to capture this **mathematically** via trainable feature hierarchies

- E.g. NN layers can be seen as feature transform with increasing abstraction

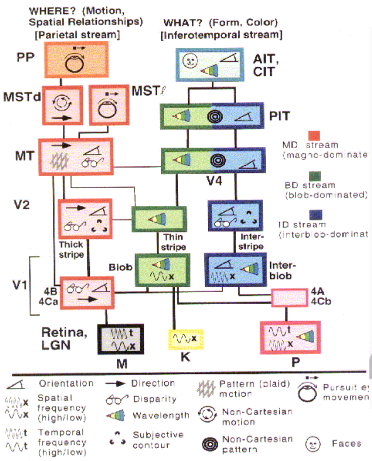


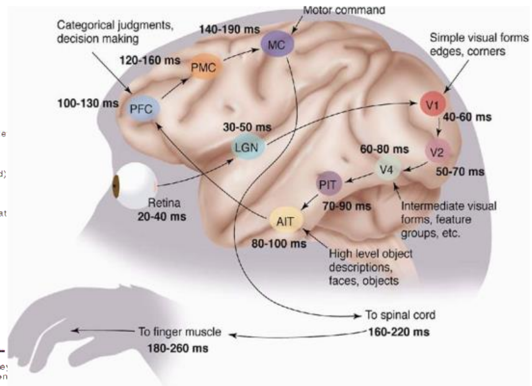
(NVIDIA)

10

Additional Support: The Visual Cortex is Also Hierarchical

- The ventral (recognition) pathway in the visual cortex has multiple stages Retina - LGN - V1 - V2 - V4 - PIT - AIT ....
- Lots of intermediate representations





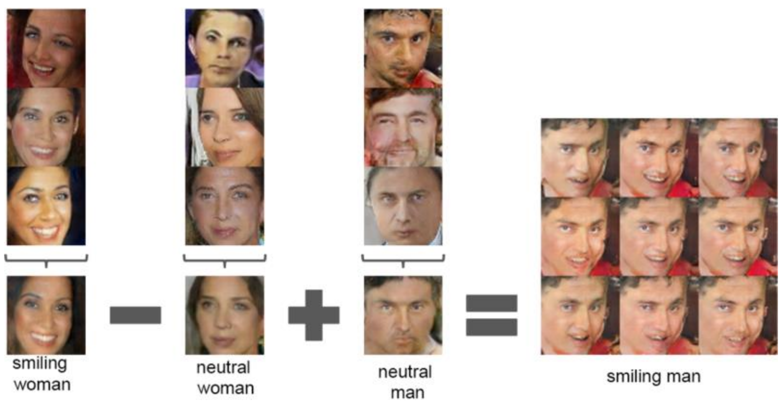
[picture from Simon Thorpe]

[Gallant & Van Essen] (NVIDIA)

11

So, Can It Learn Abstractions?

Generated similar examples using learned high-level features



Input examples -> Arithmetic on high-level features -> Results

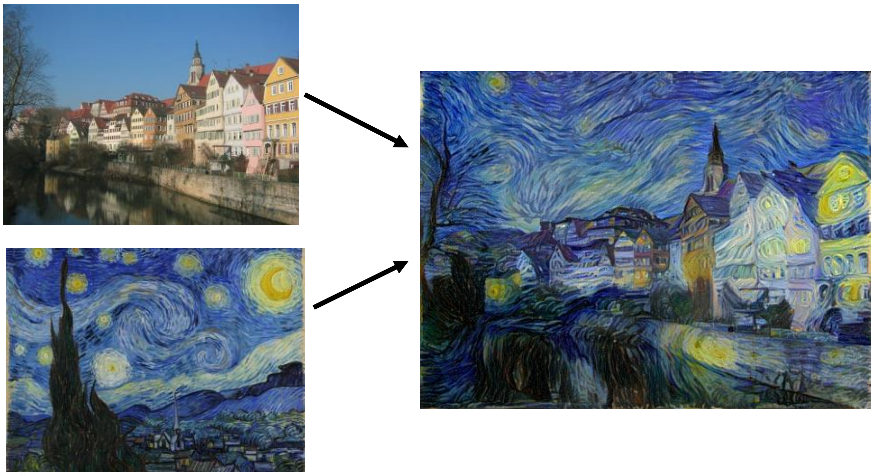
Clearly learning some kind of abstraction

- Such "concept arithmetic" doesn't always work this well...

12

So, Can It Learn Abstractions II

Neural Style Transfer (deepart.io)



So Why Is Deep Learning Taking Off Now?

People have been using Neural Networks for decades

- BUT, it turns out you need **massive scale** to really see the benefits of multiple layers

Learning deep models means more layers = **more parameters**

- More parameters requires **more data** (”identifiability”, overfitting)
- More parameters means **more computation**

Deep Neural Networks (DNNs) may have **millions to billions** parameters, trained on very large data sets

Until recently, this was not feasible

Overview: Deep Learning Driven By…

Larger data sets

- ”Big data” trend, cheap storage, internet collaboration

Faster training

- **Hardware**, algorithms and tools (e.g. Tensorflow)

Network architectures tailored for input type

- E.g. images, sequential data. Can be combined (e.g. video)

Heuristics for **reducing overfitting** during training

15

Overview: Deep Learning Driven By…

Larger data sets

- ”Big data” trend, cheap storage, internet collaboration

Faster training

- **Hardware**, algorithms and tools (e.g. Tensorflow)

Network architectures tailored for input type

- E.g. images, sequential data. Can be combined (e.g. video)

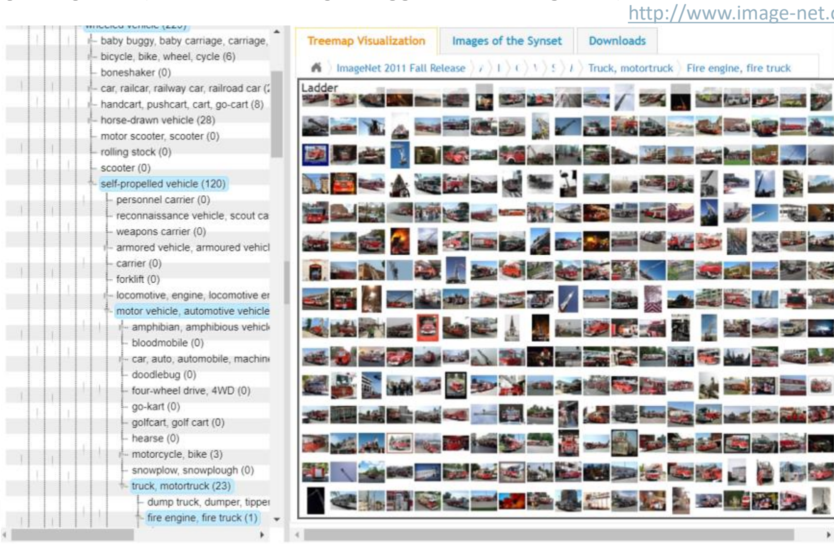
Heuristics for **reducing overfitting** during training

16

Larger Data Sets

E.g. ImageNet (> 14 million images **tagged with categories**)

http://www.image-net.org/



17

Larger Data Sets

E.g. Microsoft COCO (> 1 million images **segmented** into categories)

http://cocodataset.org

18

Companies Collect Large Private Data Sets

Internet companies like Google, Facebook and Microsoft collect plenty of data, only some of it is public (e.g. Youtube data set)

- Can get **much for free** from users
- Data is a **competitive advantage**

All major car manufacturers are researching autonomy, many are betting on deep learning

- E.g. Tesla is betting heavily on **object detection from cameras**
  - Can automatically collect raw images from their autopilot(?)

Supervised (deep) learning is the most mature technology, inputs **x** often collected automatically, but they still need somebody to provide correct outputs **y** (e.g. labels, segmentation etc)

- Such companies can have **large teams just doing labeling**
- Sometimes outsourced to other countries, or Amazon Mechanical Turk

19

Overview: Deep Learning Driven By…

Larger data sets

- ”Big data” trend, cheap storage, internet collaboration

Faster training

- **Hardware**, algorithms and tools (e.g. Tensorflow)

Network architectures tailored for input type

- E.g. images, sequential data. Can be combined (e.g. video)

Heuristics for **reducing overfitting** during training

20

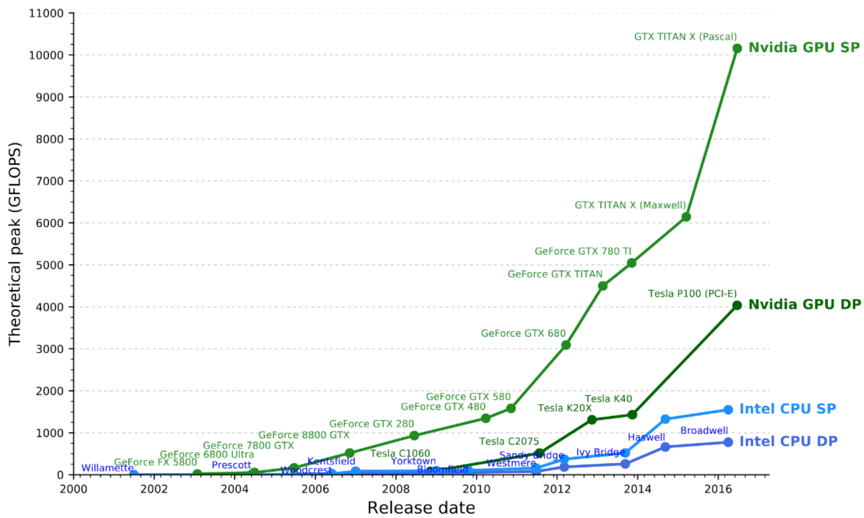
Faster Training

Consumer Desktop CPU as of 2016
- Speed: ~1 TFLOPS (10^12 **F**loating **P**oint **O**perations **P**er **S**econd)

Consumer Graphics Card (GPU)
- ~Speed: **10 TFLOPS** (single precision)
- Cost: ~$1000
- Task must be extremely parallellizable
- Neural networks are, e.g. all neurons in each layer are **independent** given inputs
- GPUs **key enabler** of deep learning

21

Faster Training - GPUs Increasingly Important

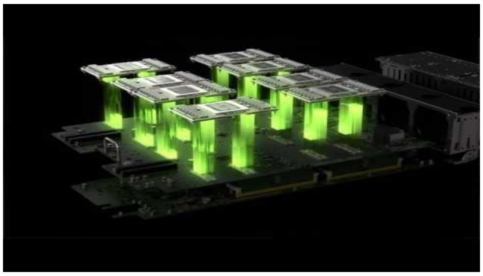


https://github.com/mgalloy/cpu-vs-gpu/

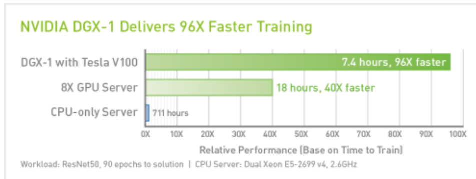
22

Faster Training - Deep Learning "Supercomputer"

- Computation for deep learning is increasingly **big business**
- NVIDIA has recent integrated solutions based on their GPU-technology



- Speed: **80-170** TFLOPS (as of 2017)
- Cost: $150 000 and up...

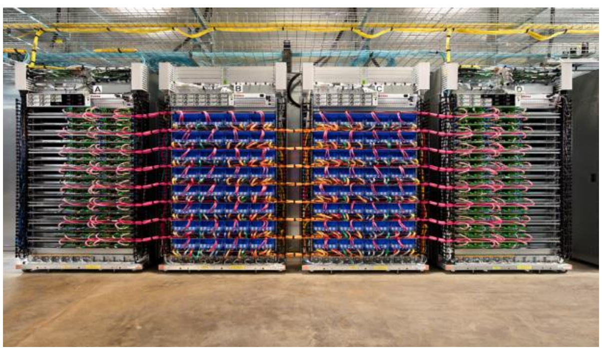


23

Faster Training - Beyond GPU's -> Custom Hardware

Google recently designed custom chips (ASICs) speficially for neural networks

- These "Tensor Processing Units" are organized into "pods"



- Speed: **11 500 TFLOPS per pod** (2017)
- Cost: Trade secret
- The speed of custom hardware usually comes at the cost of flexibility

24

Faster Training - Algorithms

Many algorithms proposed to speed up training

Mainly fall into two categories,

- Approximate gradient calculation of your NN
  - E.g. **stochastic** gradient descent (SGD) variants
- *Modifying* your NN for faster derivatives or converging in fewer iterations
  - E.g. different activation functions or network structure

Faster Training – Stochastic Gradient Descent I

Remember, training objective is a loss function against **all examples** (x_i,y_i) for different weights *w*

$$L(w) = \sum_{i=1}^{n} (NN_w(x_i) - y_i)^2$$

Want to find w that gives low loss against examples

- Gradient descent: Update w by computing gradient, $O(n \cdot w)$ using the backpropagation algorithm (lecture 1)

If we have millions of data points, do we really need all of it for useful gradients?

Faster Training – Stochastic Gradient Descent

Insight: If we just compute gradients for a **randomly selected subset m** of the total n examples, we get a *faster approximation*

Mini-batch SGD:
1. Put data set in **random order**, start at position p = 1
2. Compute gradients of *partial* loss for **m data points at a time**,

$$\hat{L}(w) = \sum_{i=p}^{p+m} (NN_w(x_i) - y_i)^2$$

3. Set p = p + m. When end of data set reached, restart at step 1.

Complexity: O($\boldsymbol{m} \cdot w$), where **m** typically 1-50, much less than n = millions.

- **The approximation over sevaral steps will average out**, and have much **lower computational** cost

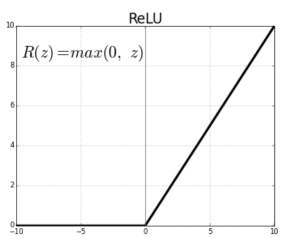
27

Faster Training – Modifying the Network

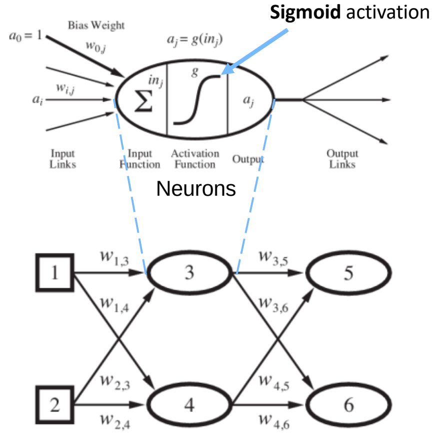
Remember the mathematical structure of neural network models:

To make optimization take fewer steps, we can change **activation function** or **connections**

The **most common** activation function these days is the **RelU,** R(z) = max(0,z)



Simpler gradients and more stable over time!



Network (*fully-connected*)

28

Training Tools – Generalized Backpropagation

Learning these advanced representations raises two issues:

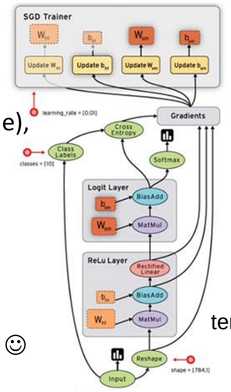
- Deep learning often uses **very large networks** with very large ("Big Data") training set
- **Advanced representations may require modifications to backpropagation**

Reverse-mode Automatic Differentiation is a technique that generalizes backpropagation to differentiate arbitrary scalar (loss) functions

Recent data flow languages like **Tensorflow** (Google), **Torch** and **Theano** let you define **arbitrary models** from primitive mathematical operations and optimize them on or several **GPUs**

Can be orders of magnitude faster!

Will we ever have to manually differentiate again? ☺



tensorflow.org

29

Code Examples: Tensorflow NN Training

- See workbook at: https://goo.gl/UHdwBX
  - Choose File->Save Copy in Drive to run and edit your own version
- Recall, in the ML Lecture 1 code examples we used a grad() function to compute the gradients of the loss function.
- This was from the Autograd package which also uses automatic differentiation like Tensorflow, but directly on python code (slower but easier to use)

30

Overview: Deep Learning Driven By…

Larger data sets

- ”Big data” trend, cheap storage, internet collaboration

Faster training

- **Hardware**, algorithms and tools (e.g. Tensorflow)

Network architectures tailored for input type

- E.g. images, sequential data. Can be combined (e.g. video)

Heuristics for **reducing overfitting** during training

31

Network Architectures Tailored to Input Type

The best results have been achieved mixing in other **network structures** than just fully-connected

Fully connected scales poorly with high-dimensional inputs such as images,

- e.g. RGB HD image is 3x1920x1080 = 6M inputs
- Assume 1000 neurons in first layer
- Then each fully-connected neuron will have a weight for each input, 1000x6M = 6 billion weights just in the first layer

The idea is to reduce the number of connections by capturing the **structure in the problem**

One very successful network structure is **convolutional neural networks**

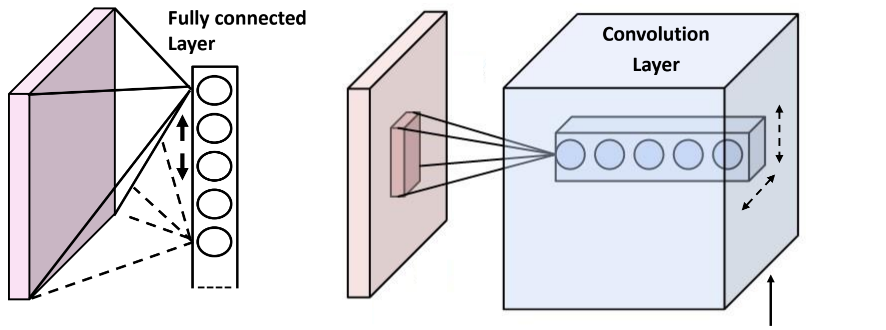
- ”Convolution” layers
- ”Pooling” layers
- Fully-connected output layers

32

Architectures - Convolutional Layers

Insight: Patterns (objects) in an image are translation ”invariant”

In a convolutional layer, **the same neurons** are applied to a sliding **window** over the inputs (e.g. image), **for each input coordinate** (e.g. pixel in image)



Each CNN neuron is a 2D pattern extractor ideal for image data
CNNs retain a lot of information even **with few neurons**

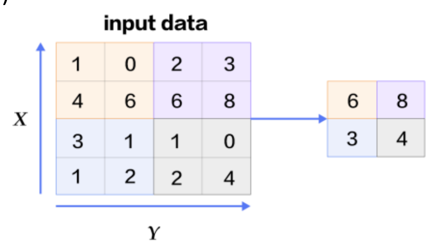
33

Architectures - CNN Pooling Layers

Insight: Patterns exist at different scales, want capture increasingly higher levels of abstraction

Pooling layers force the network to summarize information by ”downsampling”

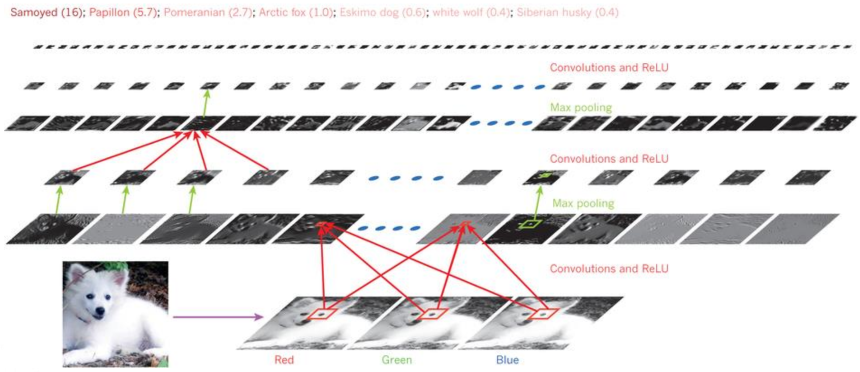
- In an image we **go from higher to lower resolution**
- Typically done by splitting image into regions and taking the **max value**
- E.g, **used after convolutional layers**, selects highest signaling neuron (pattern extractor)



34

Architectures - Convolutional Neural Networks

- **Bringing it together** into one network, in summary
- ”Convolutional” layers are for each pixel only fed inputs only from the local neighborhood to capture **object translation**
- ”Pooling” (downsampling) layers to work at different **scales** (abstraction)
- Typically you **interleave convolutions with RelU and pooling** layers



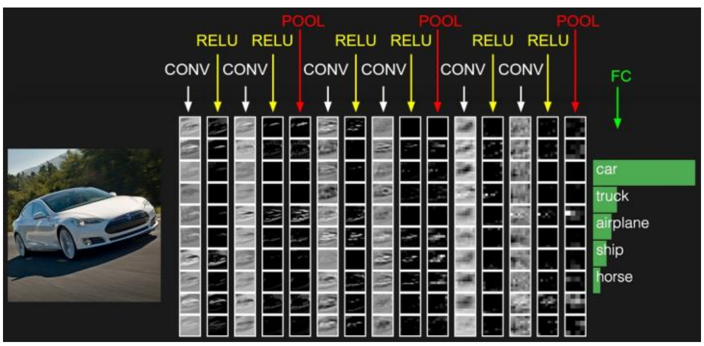
(LeCun, Bengio and Hinton, 2015)

35

Architectures – CNNs for Object Recognition

Object recognition from images is a typical classification task.

You typically add **fully-connected layer**(s) **at the end** to train a traditional classifier and the CNN features, jointly

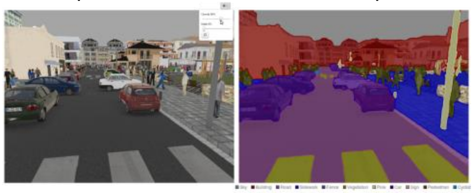


Karpathy, http://cs231n.github.io/convolutional-networks/

36

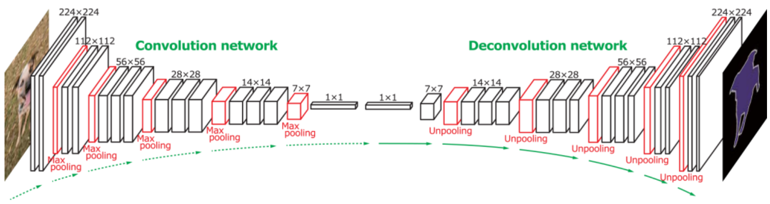
Architectures – CNNs for Segmentation

- Segmentation (typically images) requires us to classifiy **each input** (pixel), e.g. network output is same dimension as input



Classes: Road, car, pedestrian, building, pole, etc.
(NVIDIA)

- Deconvolutional networks do reverse convolution and pooling



37

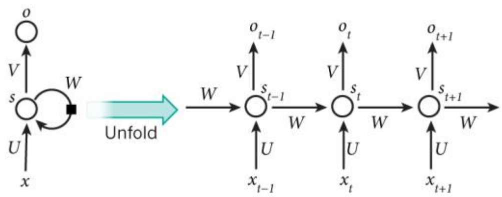
Architectures – CNN Demo

- Online javascript CNN demo: http://cs.stanford.edu/people/karpathy/convnetjs/demo/cifar10.html
- See also the CNN code example in our Tensorflow workbook
  - https://goo.gl/UHdwBX

38

Architectures: Recurrent Neural Networks

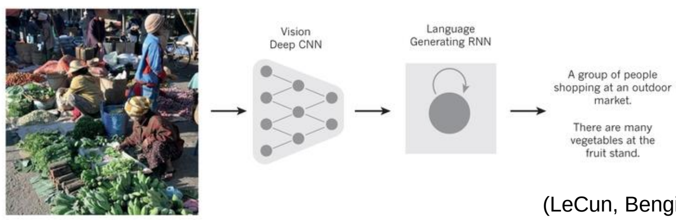
- We used the Bag of Words feature vector in the spam classification example. It's simple but it **discards the sequential structure** in text
- This is flawed since the **meaning** of a text strongly depends on the order of the words!
- **Recurrent Neural Networks (RNN)** depend not only on current input ***x***, but also **remembers** internal state *s* from previous inputs (e.g. words)
- **RNNs can be difficult to train** (variants: "Long Short-Term Memory", "Gated Recurrent Unit")



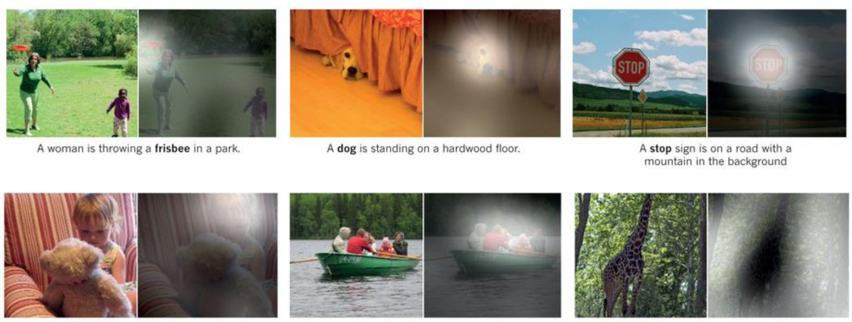
(LeCun, Bengio and Hinton, 2015)

39

Architectures: Combining Network Structures



(LeCun, Bengio and Hinton, 2015)



A woman is throwing a **frisbee** in a park.

A **dog** is standing on a hardwood floor.

A **stop** sign is on a road with a mountain in the background

A little **girl** sitting on a bed with a teddy bear.

A group of **people** sitting on a boat in the water.

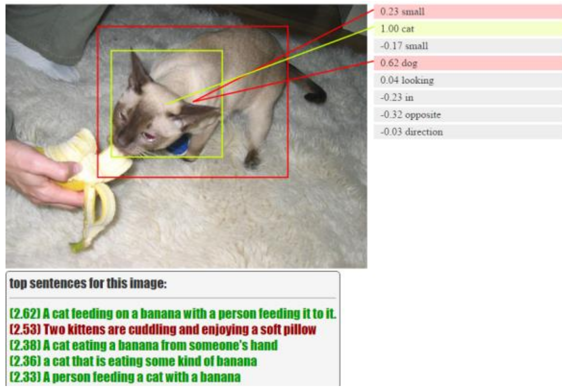
A giraffe standing in a forest with **trees** in the background.

40

Demo – Visual-Semantic Alignment

Paper: http://cs.stanford.edu/people/karpathy/deepimagesent/
Demo: http://cs.stanford.edu/people/karpathy/deepimagesent/rankingdemo/

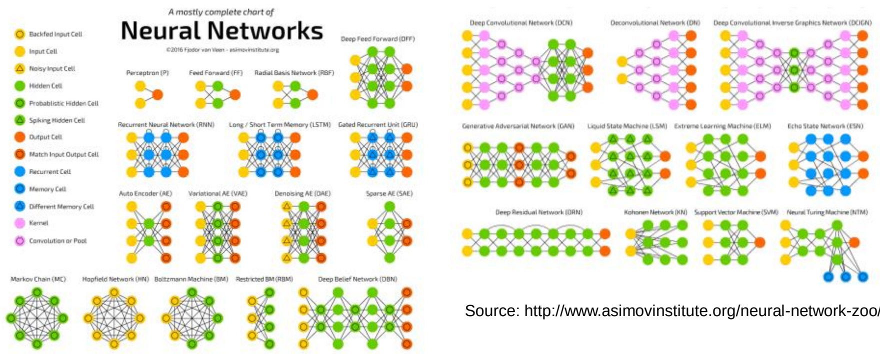
Impressive, but not perfect. Some "weird" mistakes highlight on-going debate on what neural networks have really learned.



41

Architectures: The Neural Network Zoo

Many different types of structure, more or less modular components



Source: http://www.asimovinstitute.org/neural-network-zoo/

- ...and more keeps being invented

42

Applications - What About Reinforcement Learning?

Examples so far have been mostly supervised learning, as it is the most mature and has most real applications

Several impressive research results, e.g:

- The ATARI video-game playing example from the RL lecture used ”fitted” Q-learning, where the **Q-function Q(s,a)** was fed raw pixels as state s, enabled by representing it as a **CNN**

That was two years ago, although not as mature as SL, deep RL is a hot research area

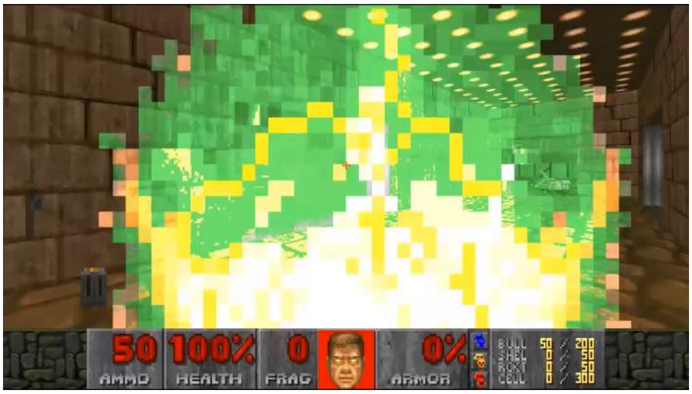
- From 80’s 2D ATARI to 90’s ”3D” Doom in <2 years

Deep Reinforcement Learning - Example

In first-person shooters like Doom the player can only observe part of the map at a time

- The agent needs memory!

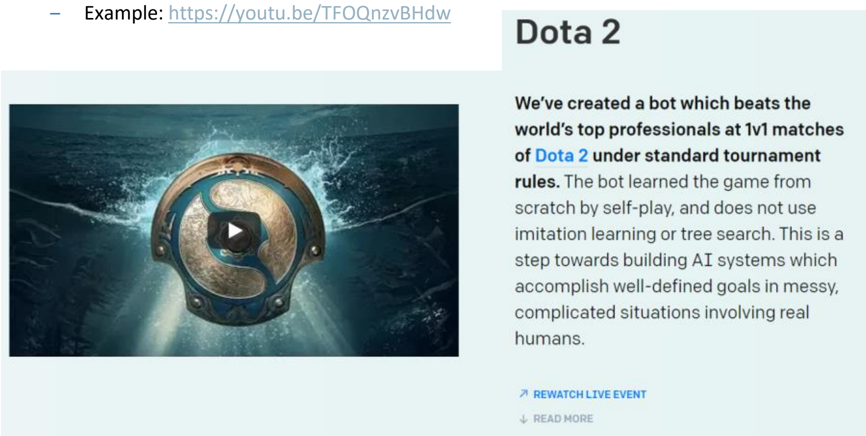
Q-learning Doom directly from **video** by using **CNN + RNN** for Q-function



Deep Reinforcement Learning – Example II

Last year, the OpenAI research institute beat human experts in the popular game of Dota 2, although a simplified scenario

- Towards learning strategies in **team games**
- This year they played human pro's in full scenarios
  - Example: https://youtu.be/TFOQnzvBHdw



45

Overview: Deep Learning Driven By…

Larger data sets

- "Big data" trend, cheap storage, internet collaboration

Faster training

- **Hardware**, algorithms and tools (e.g. Tensorflow)

Network architectures tailored for input type

- E.g. images, sequential data. Can be combined (e.g. video)

Heuristics for **reducing overfitting** during training

- Several: Dropout, BatchNormalization, etc..
- Outside the scope of this course, but lots of DL material out there on the web.

46

If Time Permits - Exjobb Opportunities

- I do research at intersection of AI, machine learning and robotics
  - Remember e.g. deep learning + quadcopter video in previous lecture
- Students interested in an academic exjobb can mail me at olov.a.andersson@liu.se
- Example topics include deep learning applications and optimization-based control
  - Math background corresponding to engineering or statistics degree helps

Closing Remarks

- Deep learning tries to learn multiple levels of abstraction to overcome the curse of dimensionality
- Bottlenecks: requires lots of data (and computation) to train
- Mainly based on multi-layer neural networks of various architectures
- An area with great potential. Lots of hype but also some impressive results
- Quickly evolving, best practices for training DNNs change almost every year

Thank you for listening!