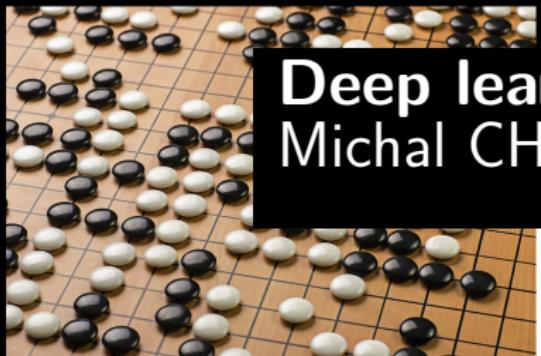


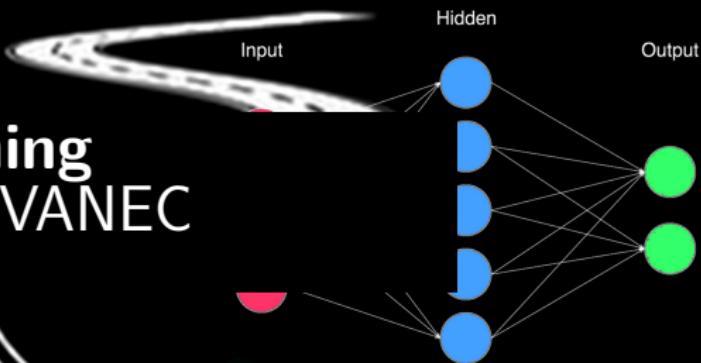
$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha [r_{t+1} + \lambda \max_a Q(s_{t+1}, a) - Q(s_t, a_t)]$$

(The New Action Value = The Old Value) + The Learning Rate  $\times$  (The New Information — the Old Information)



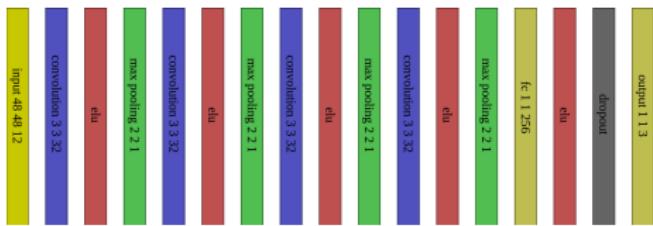
# Deep learning

## Michal CHOVANEC



# Playing on super human level

- playing Atari
  - neurons count **130 000**
  - ant 250 000
  - bee 960 000
- playing Doom
- playing GO
  - supervised training
  - reinforcement learning

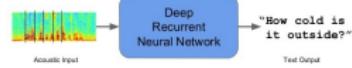


# Deep learning applications

- self driving cars - Tesla model S
- healthcare, biomedical engineering - Cell in fluid
- voice search, control - Google, Amazon Echo
- machine translation - Google translator
- image recognition - Huawei Mate 10
- game bots, robotics - DeepMind, Boston Dynamics



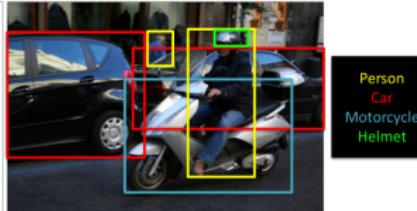
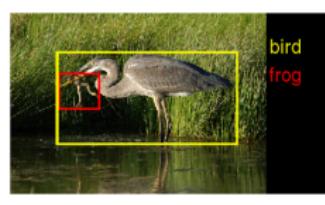
## Speech Recognition



Reduced word errors by more than 30%

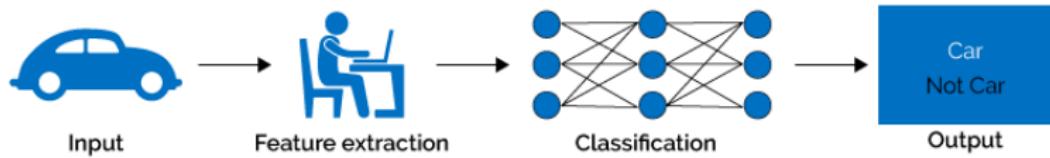
Google Research Blog - August 2012, August 2015

Research at Google

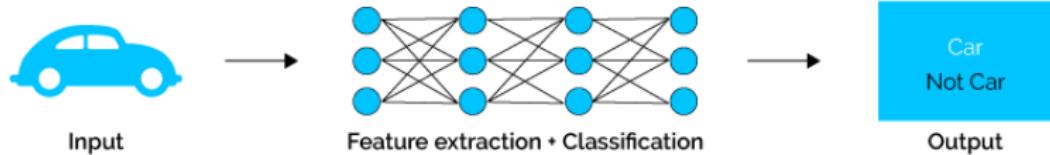


# Deep learning<sup>1</sup>

## Machine Learning



## Deep Learning

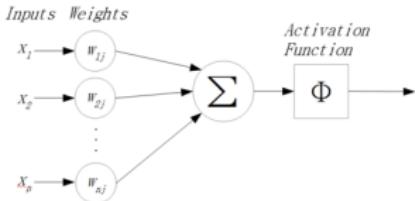


<sup>1</sup><https://medium.com/datamob/clearing-the-buzzwords-in-machine-learning-e395ad73178b>

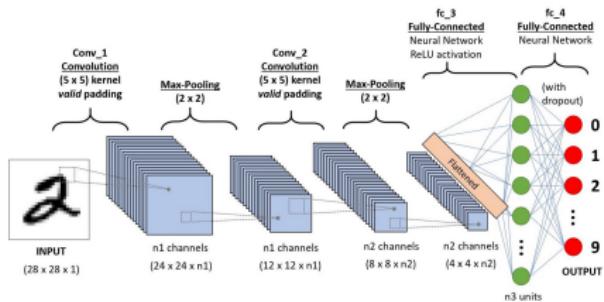
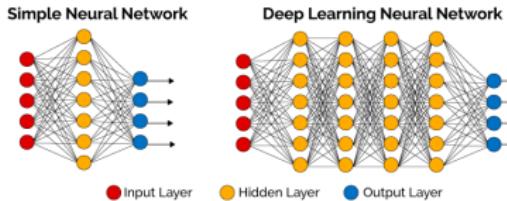
# Technologies

- Python, C++
- Nvidia CUDA
- frameworks
  - C++ : Caffe
  - Python : TensorFlow, Py Torch
  - C++, Python, C# : Rysy

# Neural network

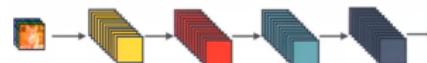


$$y(X) = f\left(\sum_i X_i W_i + b\right)$$

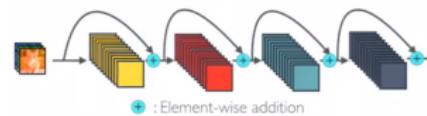


# Image recognition using CNN

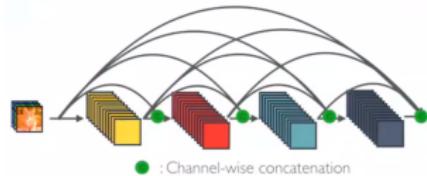
- Old methods (before NN) - 2011, 25.8%
- Convolutional, AlexNet - 2012, 16.4%
- Google inception - 2013, 6.7%
- Microsoft ResNet - 2015, 6.1%
- DenseNet - 2018, 5.17%
- CNN



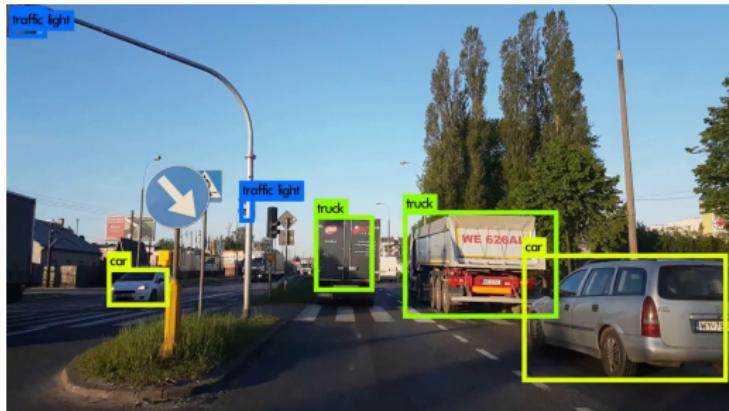
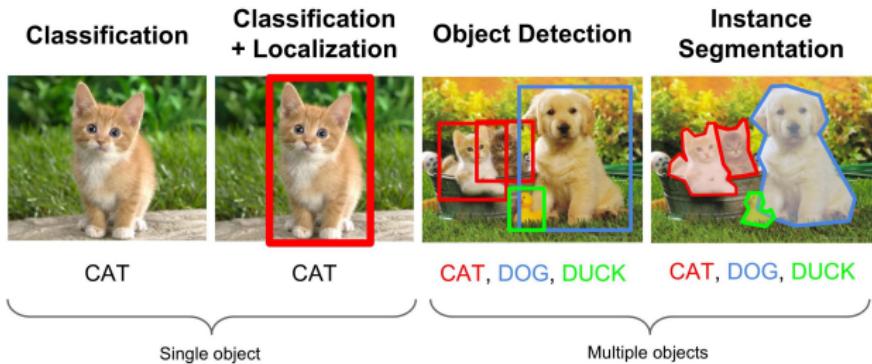
- ResNET

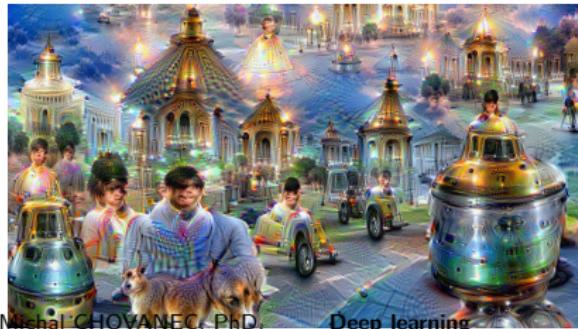


- DenseNet



# Image recognition using CNN

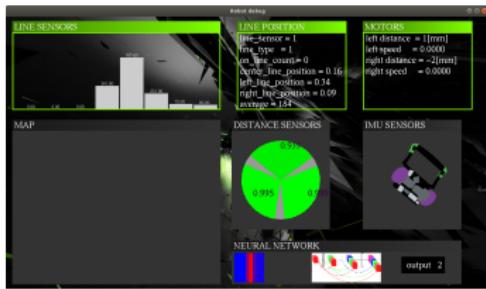




# Robotics - line follower

## Curve shape classification - go faster on straight line

- network architecture  
IN8x8x1 - C3x3x4 - P2x2 -  
DC3x3x4 - DC3x3x4 - FC5
- **first** portable embedded DenseNet implementation
- running **more than 200FPS** on ARM Cortex M4 stm32f303 (72MHz)
- response 4..5ms
- network input : 8 last line sensors results (8x8 matrix)
- **istrobot 2016** L2 first price winner

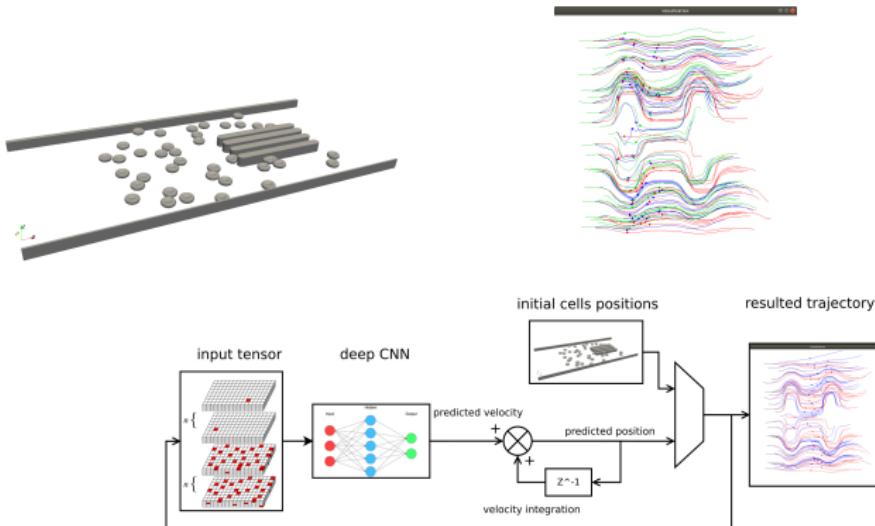


# Red blood cells trajectory prediction

Research group **Cell in fluid**

Mgr. Katarína Jasenčáková, PhD thesis

- train DNN to predict RBC trajectory from past
- 15 conv layers network (6hours training on GTX1080ti)
- input : RBC position + 7 past frames + other cells position
- output: RBC predicted velocity



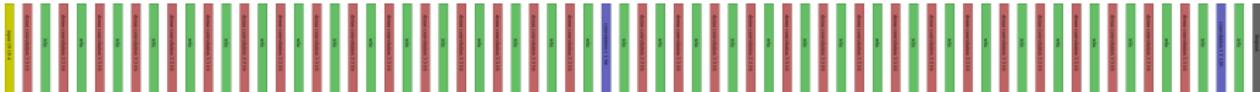
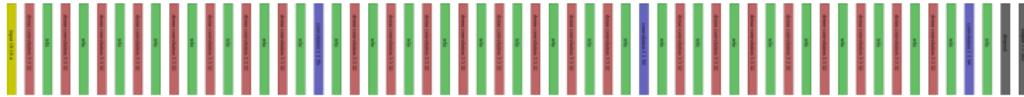
# Networks

layer	net 0	net 1	net 2	net 3	net 4	net 5	net 6	net 7
0	fc 256	conv 3x3x32	dense conv 3x3x8					
1	fc 64	fc 64	dense conv 3x3x8					
2	fc 32	fc 32	dense conv 3x3x8					
3	fc 3	fc 3	dense conv 3x3x8					
4			conv 1x1x32	conv 1x1x16	conv 1x1x32	conv 1x1x16	conv 1x1x16	conv 1x1x32
5			fc 3	dense conv 3x3x8	fc 3	dense conv 3x3x8	dense conv 3x3x8	dense conv 3x3x8
6				dense conv 3x3x8		dense conv 3x3x8	dense conv 3x3x8	dense conv 3x3x8
7				dense conv 3x3x8		dense conv 3x3x8	dense conv 3x3x8	dense conv 3x3x8
8				dense conv 3x3x8		dense conv 3x3x8	dense conv 3x3x8	dense conv 3x3x8
9				conv 1x1x32		conv 1x1x32	conv 1x1x16	conv 1x1x32
10				fc 3		fc 3	dense conv 3x3x8	dense conv 3x3x8
11							dense conv 3x3x8	dense conv 3x3x8
12							dense conv 3x3x8	dense conv 3x3x8
13							dense conv 3x3x8	dense conv 3x3x8
14							conv 1x1x32	conv 1x1x64
15							fc 3	fc 3

## GO Network architecture

we need to go much deeper for GO

- **28, 35 layers**  
dense blocks + feature pooling layer
  - **input**  
4 matrices  $19 \times 19$ : black stones, white stones, empty fields, active player
  - **output**  
recommended moves  $19 \times 19 + 1$  for pass = 362 outputs



# Usefull links

-  CHRISTOPHER J.C.H. WATKINS : Q-learning  
<http://www.gatsby.ucl.ac.uk/~dayan/papers/cjch.pdf>
-  Richard S. Sutton : Reinforcement Learning: An Introduction  
<https://www.amazon.com/Reinforcement-Learning-Introduction-Adaptive-Computation/dp/0262193981>
-  Google DeepMind : Playing Atari with Deep Reinforcement Learning  
<https://arxiv.org/pdf/1312.5602.pdf>
-  Google DeepMind : Dueling Network Architectures for Deep Reinforcement Learning  
<https://arxiv.org/pdf/1511.06581.pdf>
-  Google DeepMind :Mastering the Game of Go without Human Knowledge  
[https://deepmind.com/documents/119/agz\\_unformatted\\_nature.pdf](https://deepmind.com/documents/119/agz_unformatted_nature.pdf)
-  Andrej Karpathy : Pong from pixels  
<http://karpathy.github.io/2016/05/31/r1/>
-  Maxim Lapan : Deep reinforcement learning  
<https://www.amazon.com/Practical-Reinforcement-Learning-Maxim-Lapan/dp/1788834240>
-  Mohit Sewak : Practical Convolutional Neural Networks  
<https://www.amazon.com/Practical-Convolutional-Neural-Networks-Implement/dp/1788392302>
-  Densely Connected Convolutional Networks  
<https://arxiv.org/pdf/1608.06993.pdf>

# Q&A



michal chovanec ([michal.nand@gmail.com](mailto:michal.nand@gmail.com))  
[www.youtube.com/channel/UCzVvP2ou8v3afNiVrPAHQGg](https://www.youtube.com/channel/UCzVvP2ou8v3afNiVrPAHQGg)  
github <https://github.com/michalnand>