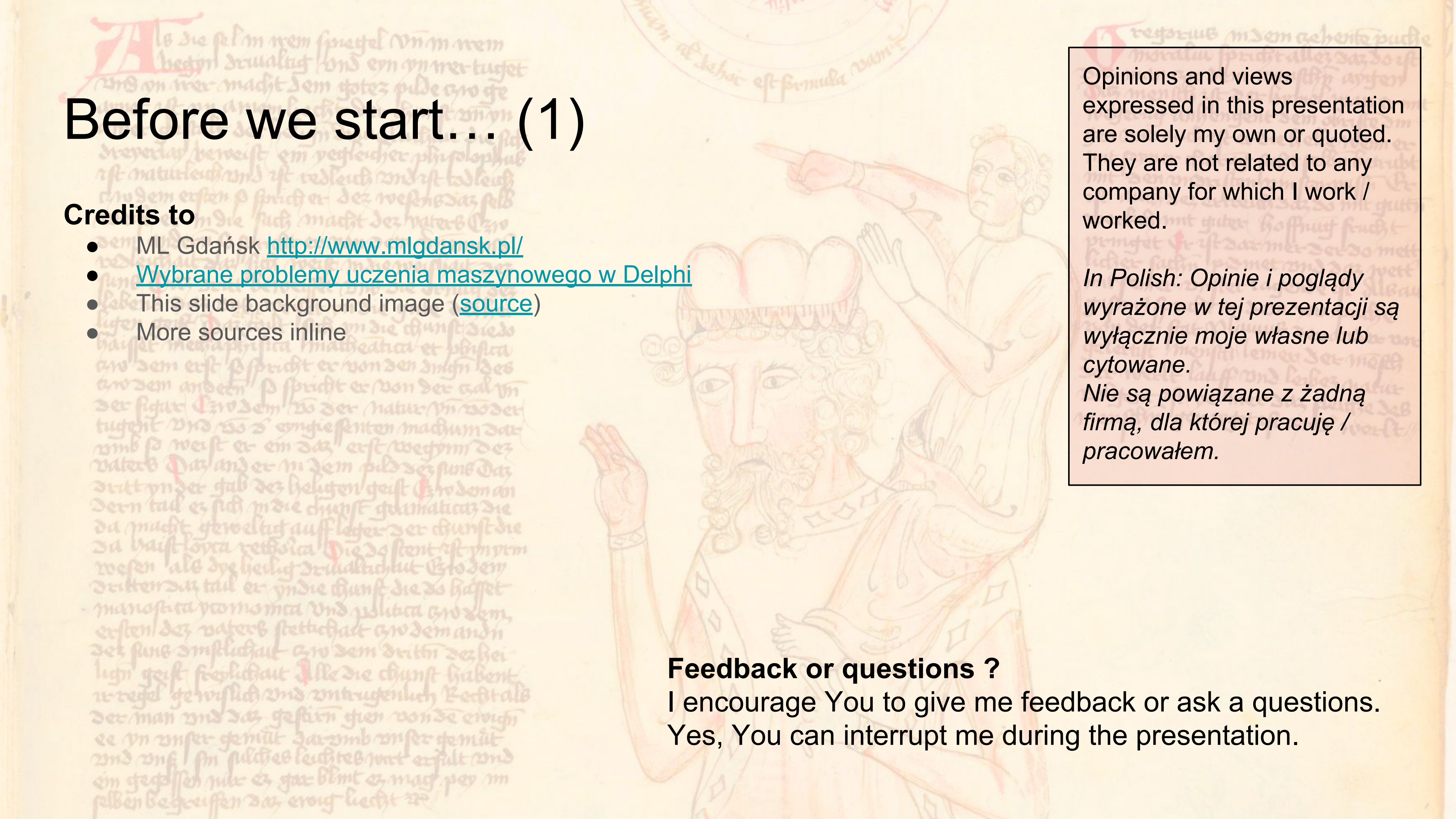
Propagacja wsteczna błędu w sieciach neuronowych

Omówienie algorytmu wstecznej propagacji błędu (ang. backpropagation); przykłady zastosowania, minimum lokalne v.s. globalne, kryteria stopu, oraz inne osobliwości. Trening pojedynczego perceptronu oraz trening sieci neuronowej. Implementacja algorytmu wstecznej propagacji błędu w Delphi.

— Piotr Chlebek
https://www.linkedin.com/in/piotrr/

20 Czerwiec 2017
Zlot Programistów Delphi
http://delphi.pl/zlot/

Version 0.5



Before we start... (1)

Credits to

- ML Gdańsk http://www.mlgdansk.pl/
- Wybrane problemy uczenia maszynowego w Delphi
- This slide background image (source)
- More sources inline

Opinions and views expressed in this presentation are solely my own or quoted. They are not related to any company for which I work / worked.

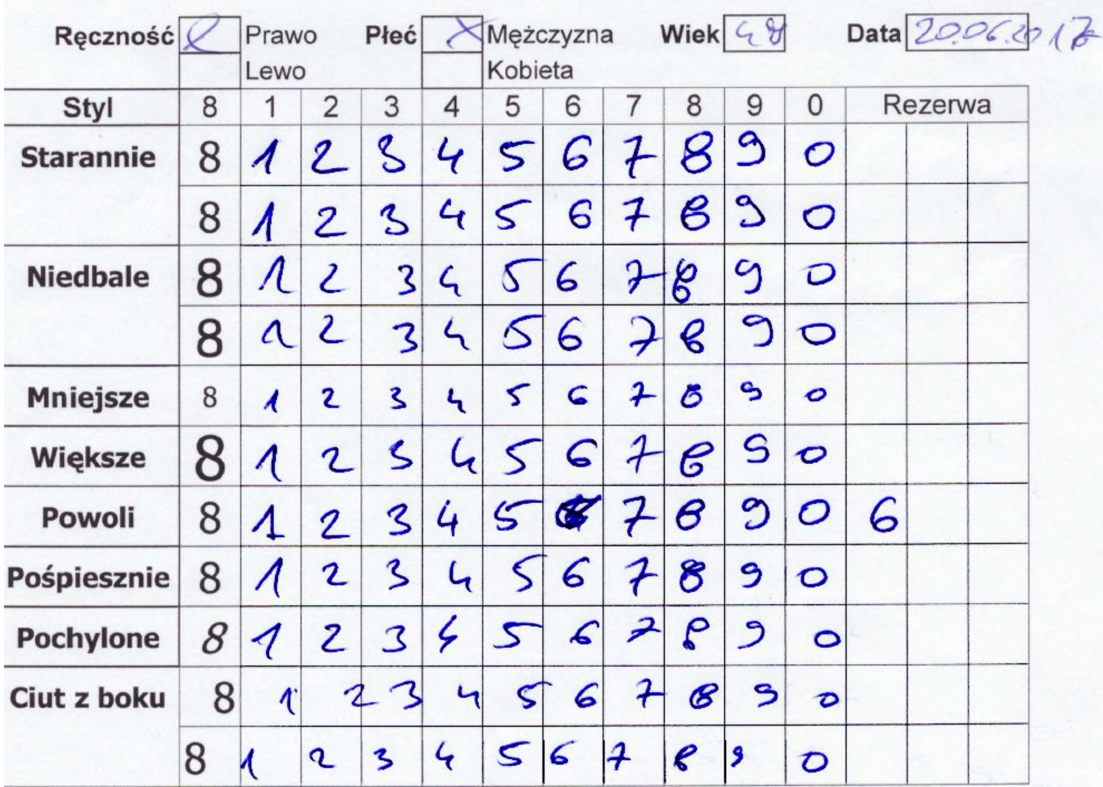
In Polish: Opinie i poglądy wyrażone w tej prezentacji są wyłącznie moje własne lub cytowane.
Nie są powiązane z żadną firmą, dla której pracuję / pracowałem.

Feedback or questions ?
I encourage You to give me feedback or ask a questions.
Yes, You can interrupt me during the presentation.

Before we start… (2)

Voluntary data collection.

Collected data will be published under a **Public Domain** license.



Ręczność ☒ Prawo □ Lewo

Płeć ☒ Mężczyzna □ Kobieta

Wiek 48

Data 20.06.2017

Styl	8	1	2	3	4	5	6	7	8	9	0	Rezerwa		
Starannie	8	1	2	3	4	5	6	7	8	9	0			
	8	1	2	3	4	5	6	7	8	9	0			
Niedbale	8	1	2	3	4	5	6	7	8	9	0			
	8	1	2	3	4	5	6	7	8	9	0			
Mniejsze	8	1	2	3	4	5	6	7	8	9	0			
Większe	8	1	2	3	4	5	6	7	8	9	0			
Powoli	8	1	2	3	4	5	~~6~~	7	8	9	0	6		
Pośpiesznie	8	1	2	3	4	5	6	7	8	9	0			
Pochylone	8	1	2	3	4	5	6	7	8	9	0			
Ciut z boku	8	1	2	3	4	5	6	7	8	9	0			
	8	1	2	3	4	5	6	7	8	9	0			

Piotr Chlebek

- Starting start-up: DataUp.ai
- Last ~8 years: Researcher & Software Engineer @ Intel
  - Speech Recognition & Machine Learning projects, successful products on the market with industry recognition & awards.
- 20 years of experience (in total), R&D a wide range of innovative projects
- Machine Learning passionate & evangelist
  - Machine Learning Gdańsk community public speaker.
  - Sharky Neural Network - software for education
  - Chess programming M.Eng diploma
- I Love Delphi (hobby)

Self driven, software R&D and machine learning passionate. With high math and programming background. Focused on solving challenging problems and delivering end-to-end solutions.
Three patents.

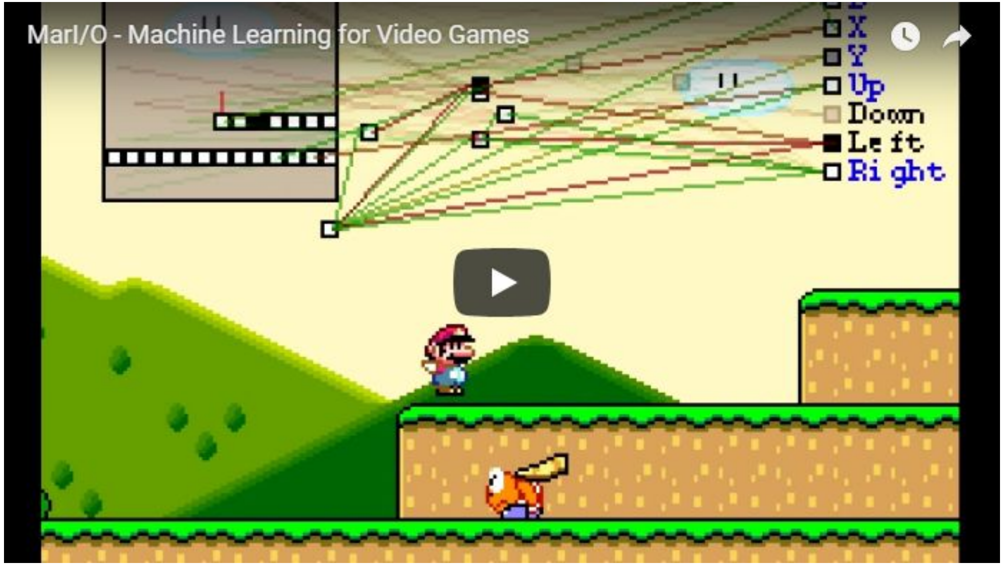
Deep Learning can be **Exciting**

Backpropagation is a method used in artificial neural networks to calculate a gradient that is needed for calculation of the network weights. It is commonly used to train deep neural networks and is essential for a **deep learning**.

Deep learning solves real problems, such as:

Product Recommendation, Clustering / Segmentation, Medical Diagnosis, Speech Recognition / Synthesis, Optical Character Recognition / Mimicking (inc. Handwritten), Object Detection (inc. Localization), Face Detection / Recognition, Spam or Fraud Detection, Natural Language Understanding / Generation, ...

Deep Learning can be **Fun**

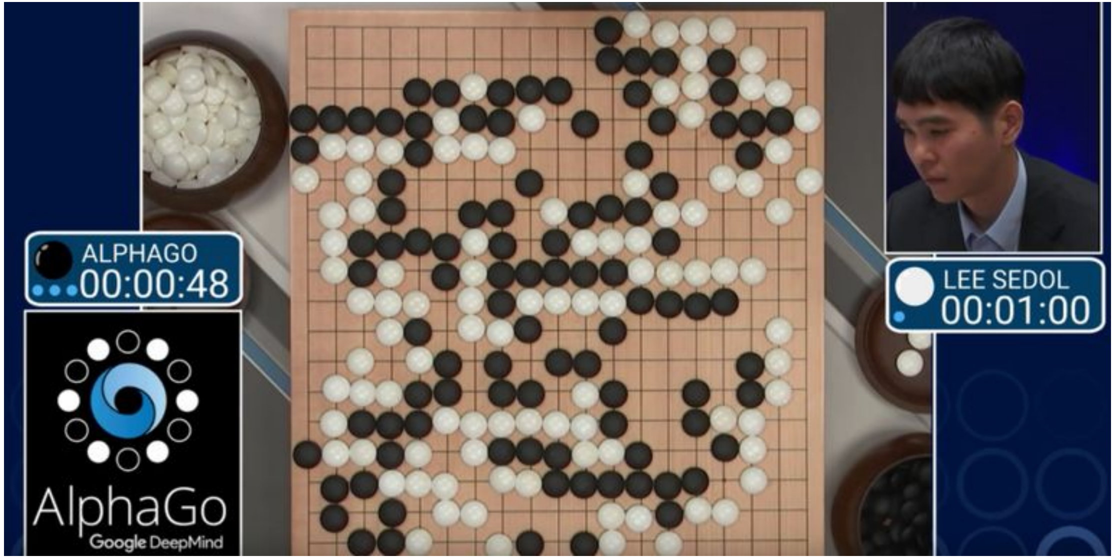


MarI/O - Machine Learning for Video Games

Source: https://www.youtube.com/watch?v=qv6UVOQ0F44

Neural Networks + Genetic Algorithms.

Deep Learning can be **a Champion**

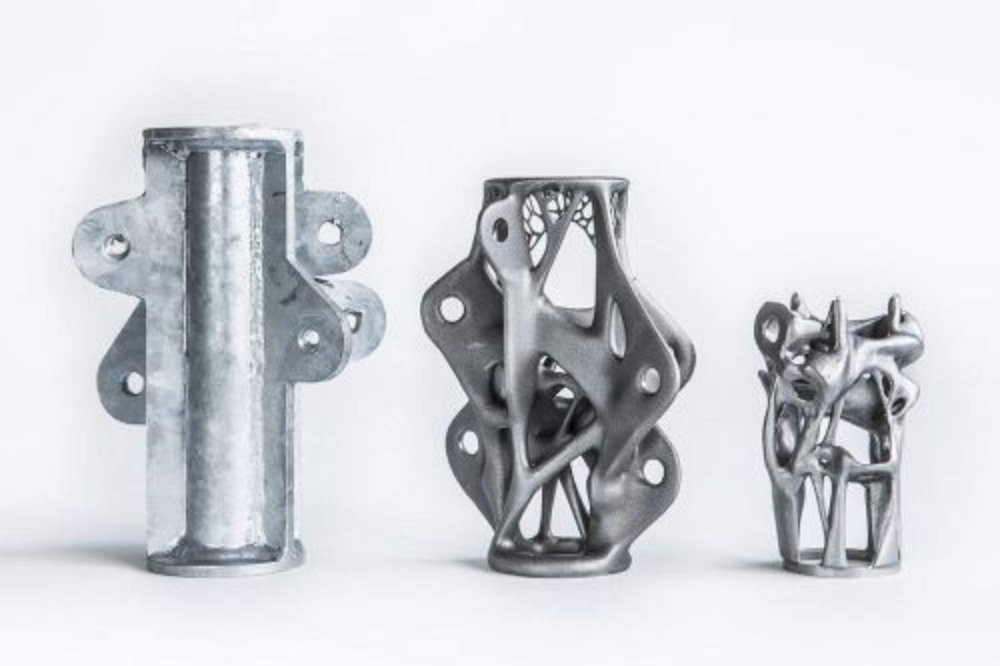


DeepMind's AlphaGo Zero Becomes Go Champion Without Human Input

Img source: https://www.popularmechanics.com/technology/a19863/googles-alphago-ai-wins-second-game-go/

AlphaGo - Monte Carlo Tree Search with Artificial Neural Network (based on human and computer play),
AlphaGo Zero - Reinforcement Learning (no human knowledge).

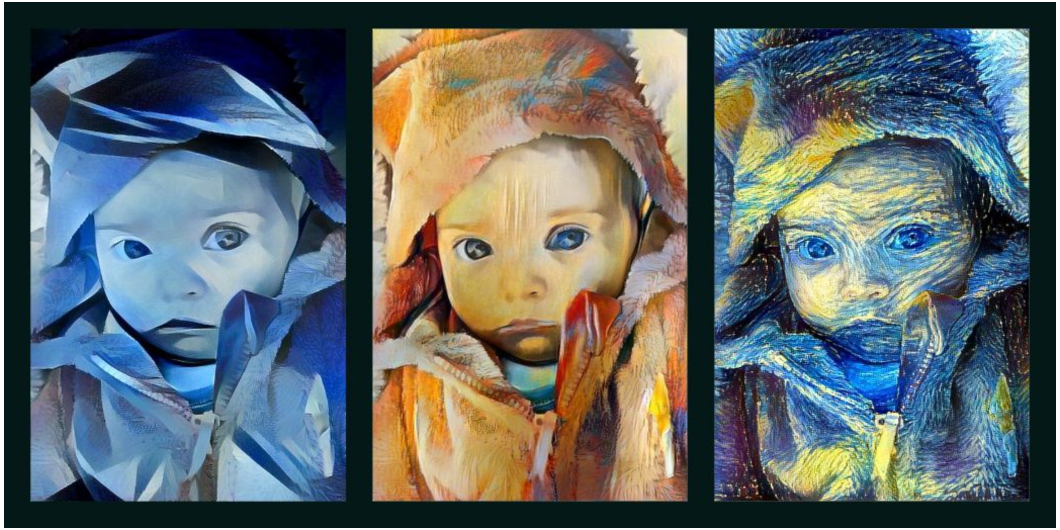
Deep Learning can be **Creative**



The Alien Style of Deep Learning Generative Design

Source: https://medium.com/intuitionmachine/the-alien-look-of-deep-learning-generative-design-5c5f871f7d10

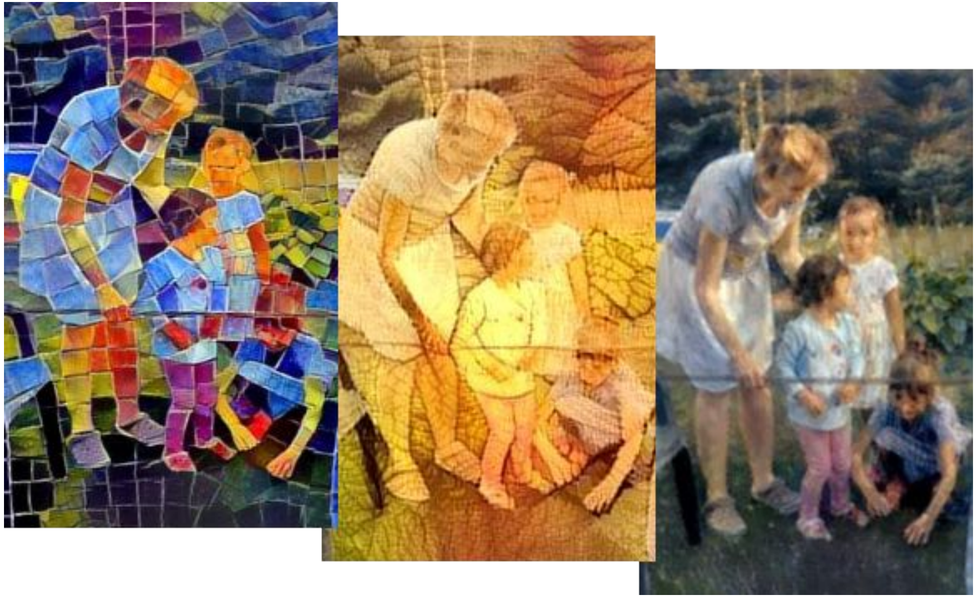
Deep Learning can be **Beautiful**



Three styles of the same portrait

Source: Author's material treated with https://deepdreamgenerator.com/

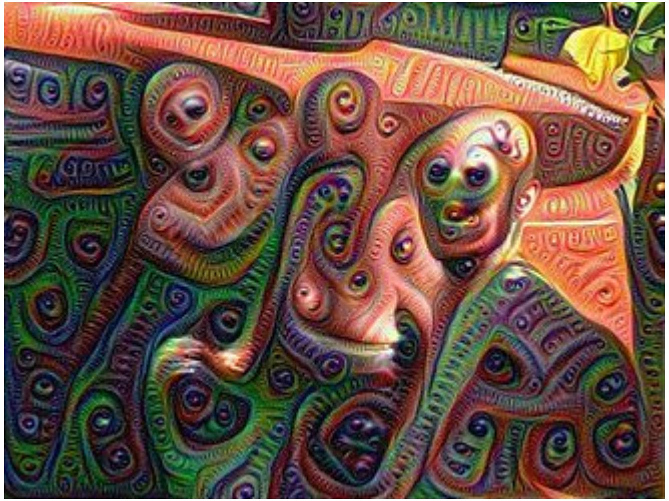
Deep Style v.s. Deep Dream



A Neural Algorithm of Artistic Style

- Leon A. Gatys, Alexander S. Ecker, Matthias Bethge

https://arxiv.org/pdf/1508.06576v2.pdf



DeepDream is a computer vision program created by Google which uses a convolutional neural network to find and enhance patterns in images via algorithmic pareidolia, thus creating a dream-like hallucinogenic appearance in the deliberately over-processed images.

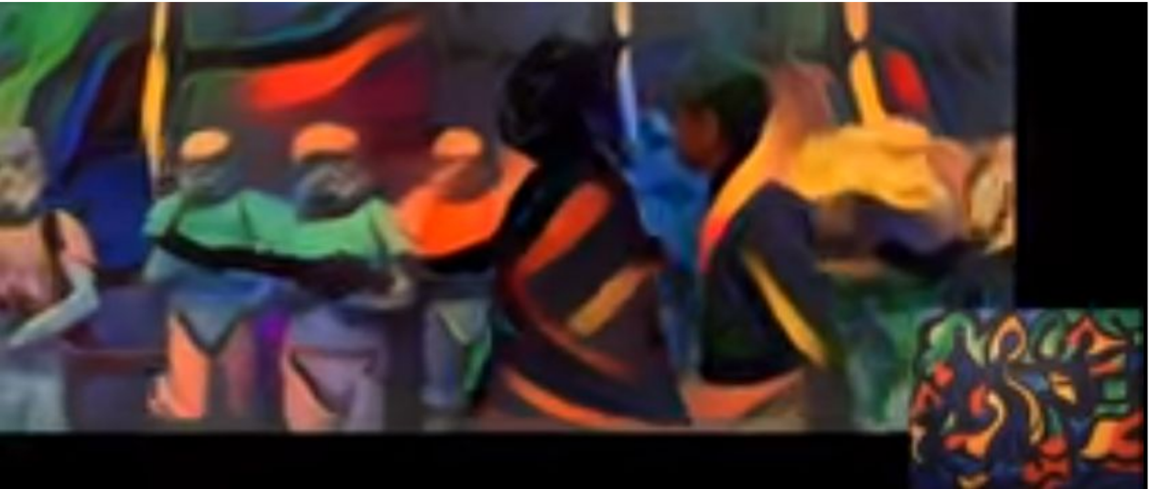
https://en.wikipedia.org/wiki/DeepDream

Deep Style Rules!

Turning our sketches into art with machine learning

- Cambridge Consultants,Sep 2017

https://www.cambridgeconsultants.com/vincent
https://www.youtube.com/watch?v=RXW9Nw-h7QE



Artistic style transfer for videos

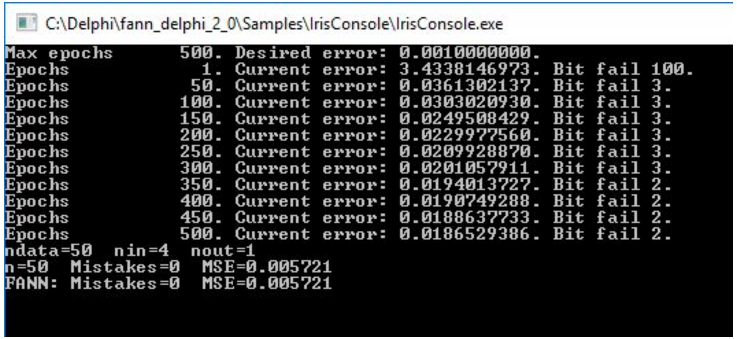
- Manuel Ruder, Alexey Dosovitskiy and Thomas Brox

http://arxiv.org/abs/1604.08610
https://www.youtube.com/watch?v=Khuj4ASldmU

Check My Software

IrisConsole

https://github.com/pcbua/IrisConsole



C:\Delphi\fann_delphi_2_0\Samples\IrisConsole\IrisConsole.exe

```
Max epochs       500. Desired error: 0.0010000000.
Epochs             1. Current error: 3.4338146973. Bit fail 100.
Epochs            50. Current error: 0.0361302137. Bit fail 3.
Epochs           100. Current error: 0.0303020930. Bit fail 3.
Epochs           150. Current error: 0.0249508429. Bit fail 3.
Epochs           200. Current error: 0.0229977560. Bit fail 3.
Epochs           250. Current error: 0.0209928870. Bit fail 3.
Epochs           300. Current error: 0.0201057911. Bit fail 3.
Epochs           350. Current error: 0.0194013727. Bit fail 2.
Epochs           400. Current error: 0.0190749288. Bit fail 2.
Epochs           450. Current error: 0.0188637733. Bit fail 2.
Epochs           500. Current error: 0.0186529386. Bit fail 2.
ndata=50  nin=4  nout=1
n=50  Mistakes=0  MSE=0.005721
FANN: Mistakes=0  MSE=0.005721
```

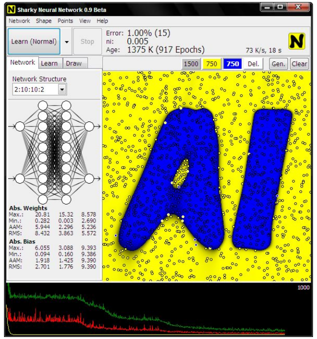
Neural Network example with Delphi + FANN (Fast Artificial Neural Network Library). Solves Iris flowers classification problem.

Credits

- **Ronald Fisher** for Iris flower data set (1936),
- **FANN** Authors.

Sharky Neural Network

http://www.sharktime.com/snn



```
Error:  1.00% (15)
ni:     0.005
Age:    1375 K (917 Epochs)        73 K/s, 18 s

Network Structure
2:10:10:2

Abs. Weights
Max.:   20.81   15.32   8.578
Min.:   0.282   0.003   2.690
AAM:    5.944   2.296   5.236
RMS:    8.432   3.863   5.572
Abs. Bias
Max.:   6.055   3.088   9.393
Min.:   0.094   0.160   9.386
AAM:    1.918   1.425   9.390
RMS:    2.701   1.776   9.390
```

Classification neural network in action.
Free Win32 software for playing with neural networks classification.

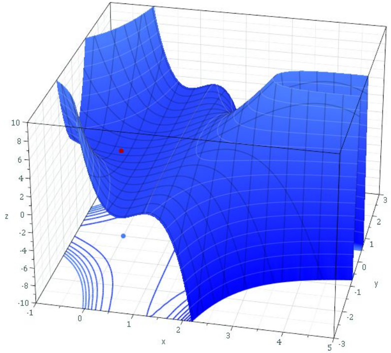
Credits

- **Andrea Fontana** for idea of NN shadow between classes,
- **Piotr Chlebek** for idea of NN live view.

Backpropagation

- Wikipedia: https://pl.wikipedia.org/wiki/Propagacja_wsteczna

  Propagacja wsteczna – podstawowy algorytm uczenia *nadzorowanego* wielowarstwowych, jednokierunkowych sieci neuronowych. Podaje on przepis na zmianę wag dowolnych połączeń elementów przetwarzających rozmieszczonych w sąsiednich warstwach sieci. Oparty jest on na minimalizacji sumy kwadratów błędów (lub innej funkcji błędu) uczenia z wykorzystaniem optymalizacyjnej metody największego spadku. Dzięki zastosowaniu specyficznego sposobu propagowania błędów uczenia sieci powstałych na jej wyjściu, tj. przesyłania ich od warstwy wyjściowej do wejściowej, algorytm propagacji wstecznej stał się jednym z najskuteczniejszych algorytmów uczenia sieci.
- Supervised learning
- Loss function: Mean Square Error, Cross-entropy cost, ..
- Online v.s. Batch v.s Single Batch
- Momentum
- ...

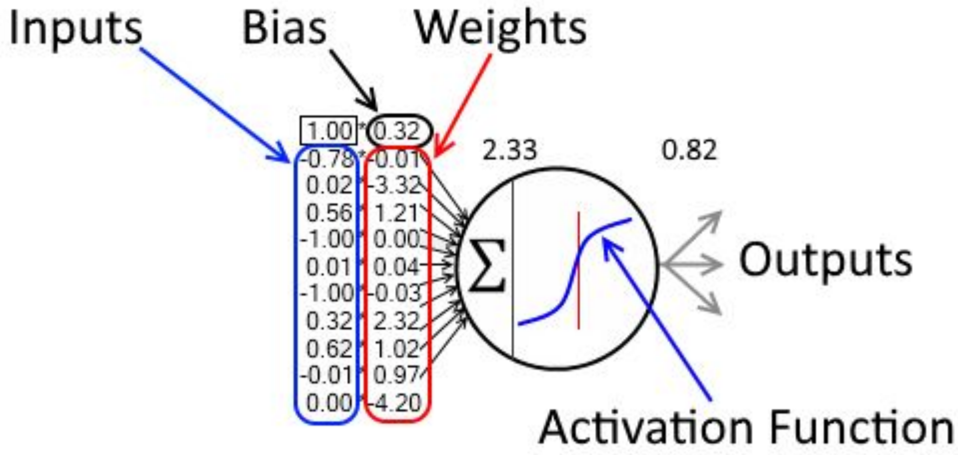


Backpropagation Algorithm

```
initialize network weights (often small random values)
  do
     forEach training example named ex
        prediction = neural-net-output(network, ex)  // forward pass
        actual = teacher-output(ex)
        compute error (prediction - actual) at the output units
        compute dWh for all weights from hidden layer to output layer  // backward pass
        compute dWi for all weights from input layer to hidden layer   // backward pass continued
        update network weights // input layer not modified by error estimate
  until all examples classified correctly or another stopping criterion satisfied
  return the network
```

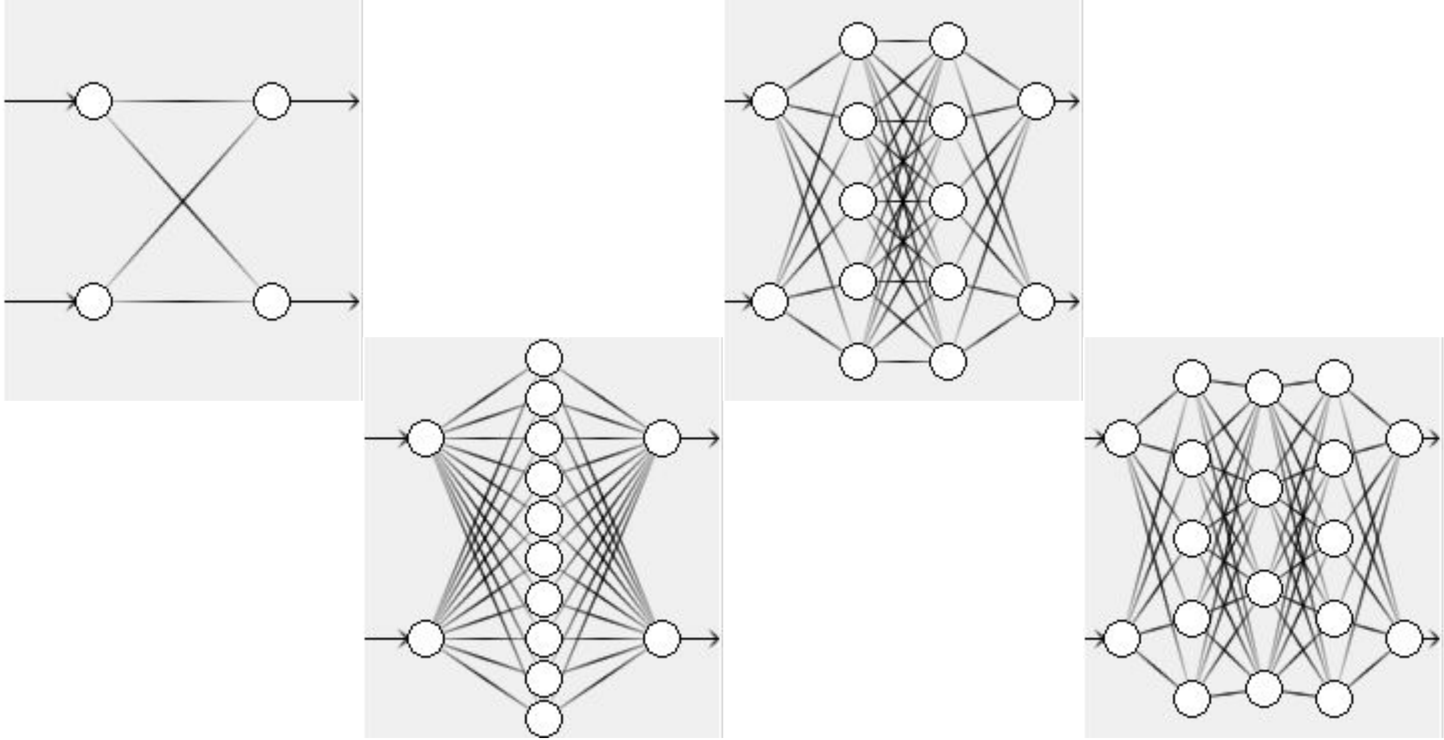
Source: https://en.wikipedia.org/wiki/Backpropagation

The Neuron

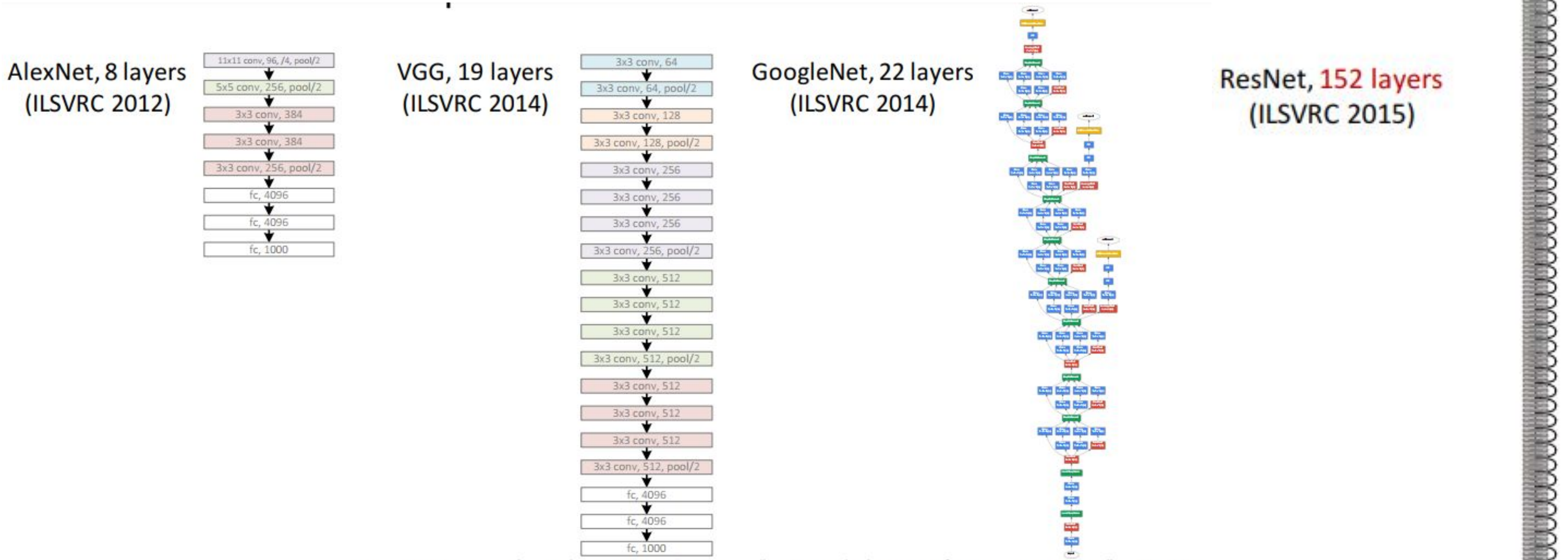


Very simple computing unit. Easy to parallelize in the hardware.

The Network



Deep, deeper...



Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. "Deep Residual Learning for Image Recognition". CVPR 2016.

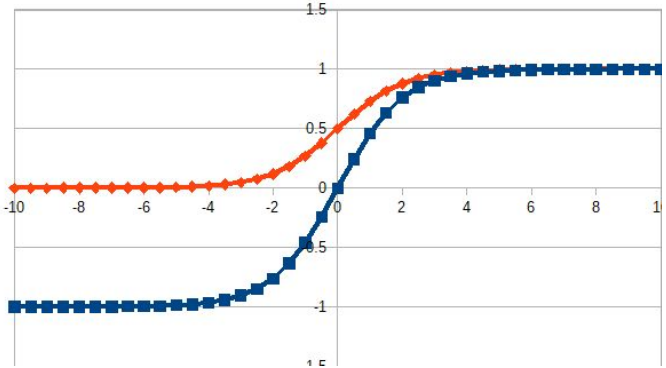
Source: Kaiming He

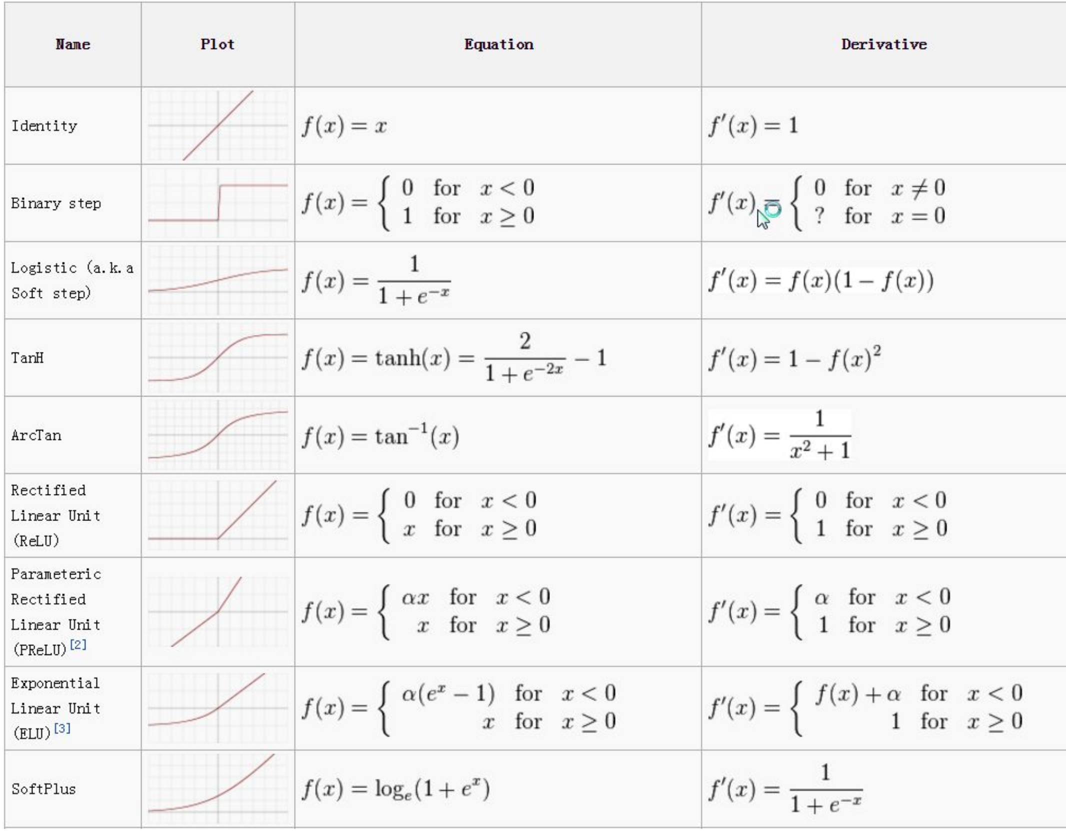
Activation Function

Bipolar sigmoid activation function:

$$f(x)=2/(1+e^{-\beta x})-1$$

Sigmoid (logistic) v.s. bipolar sigmoid:



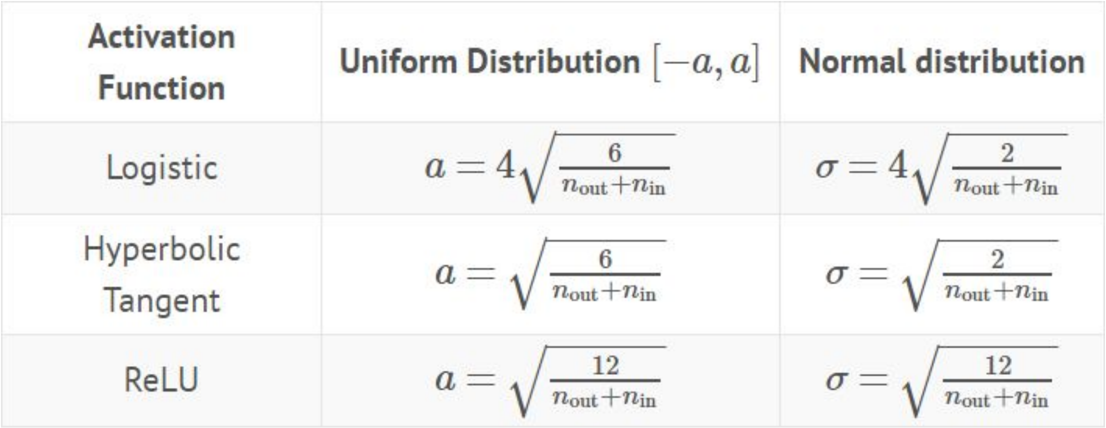


Name	Plot	Equation	Derivative
Identity		$f(x)=x$	$f'(x)=1$
Binary step		$f(x)=\begin{cases}0 & \text{for } x<0\\ 1 & \text{for } x\ge 0\end{cases}$	$f'(x)=\begin{cases}0 & \text{for } x\ne 0\\ ? & \text{for } x=0\end{cases}$
Logistic (a.k.a Soft step)		$f(x)=\frac{1}{1+e^{-x}}$	$f'(x)=f(x)(1-f(x))$
TanH		$f(x)=\tanh(x)=\frac{2}{1+e^{-2x}}-1$	$f'(x)=1-f(x)^2$
ArcTan		$f(x)=\tan^{-1}(x)$	$f'(x)=\frac{1}{x^2+1}$
Rectified Linear Unit (ReLU)		$f(x)=\begin{cases}0 & \text{for } x<0\\ x & \text{for } x\ge 0\end{cases}$	$f'(x)=\begin{cases}0 & \text{for } x<0\\ 1 & \text{for } x\ge 0\end{cases}$
Parameteric Rectified Linear Unit (PReLU)[2]		$f(x)=\begin{cases}\alpha x & \text{for } x<0\\ x & \text{for } x\ge 0\end{cases}$	$f'(x)=\begin{cases}\alpha & \text{for } x<0\\ 1 & \text{for } x\ge 0\end{cases}$
Exponential Linear Unit (ELU)[3]		$f(x)=\begin{cases}\alpha(e^x-1) & \text{for } x<0\\ x & \text{for } x\ge 0\end{cases}$	$f'(x)=\begin{cases}f(x)+\alpha & \text{for } x<0\\ 1 & \text{for } x\ge 0\end{cases}$
SoftPlus		$f(x)=\log_e(1+e^x)$	$f'(x)=\frac{1}{1+e^{-x}}$

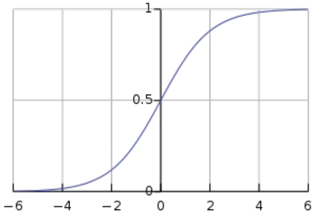
Img source: https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6

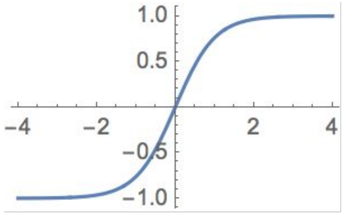
Initialization In Deep Neural Networks

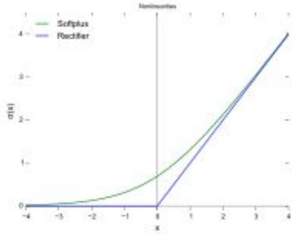
Based on Xavier, Glorot and Bengio (2010).



Activation Function	Uniform Distribution $[-a, a]$	Normal distribution
Logistic	$a = 4\sqrt{\frac{6}{n_{\text{out}}+n_{\text{in}}}}$	$\sigma = 4\sqrt{\frac{2}{n_{\text{out}}+n_{\text{in}}}}$
Hyperbolic Tangent	$a = \sqrt{\frac{6}{n_{\text{out}}+n_{\text{in}}}}$	$\sigma = \sqrt{\frac{2}{n_{\text{out}}+n_{\text{in}}}}$
ReLU	$a = \sqrt{\frac{12}{n_{\text{out}}+n_{\text{in}}}}$	$\sigma = \sqrt{\frac{12}{n_{\text{out}}+n_{\text{in}}}}$

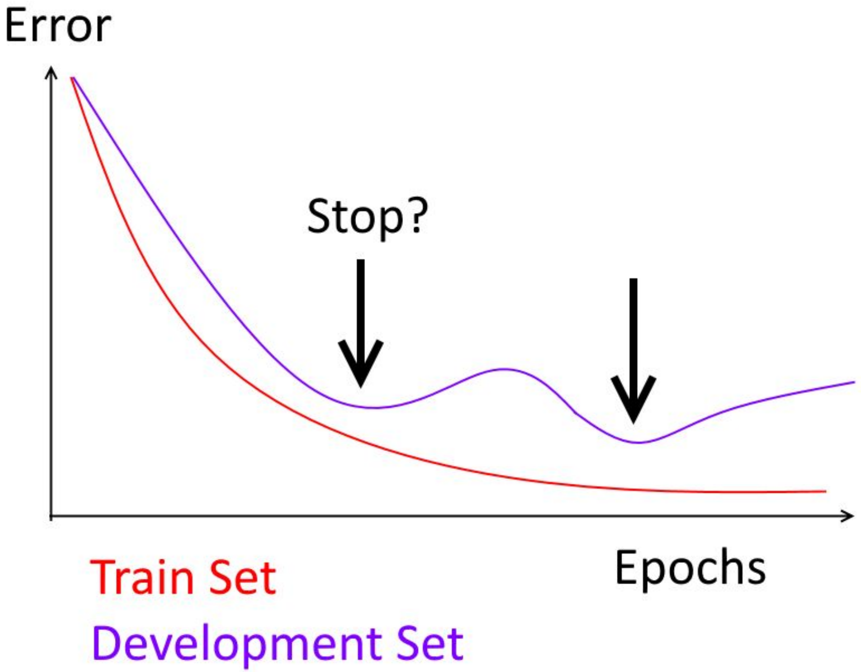




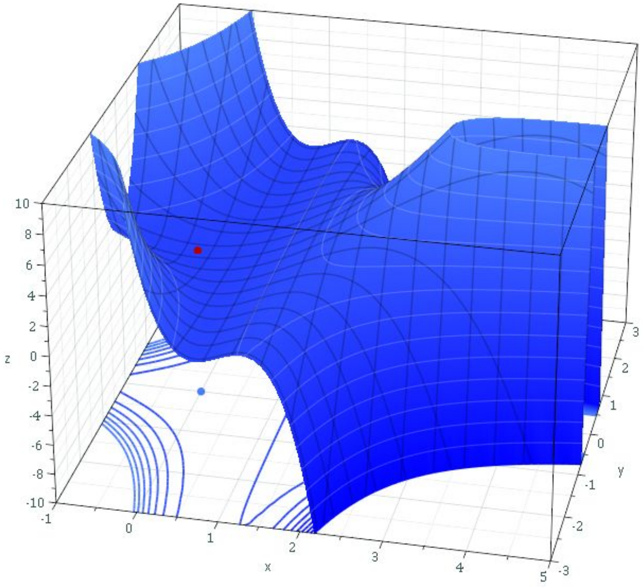


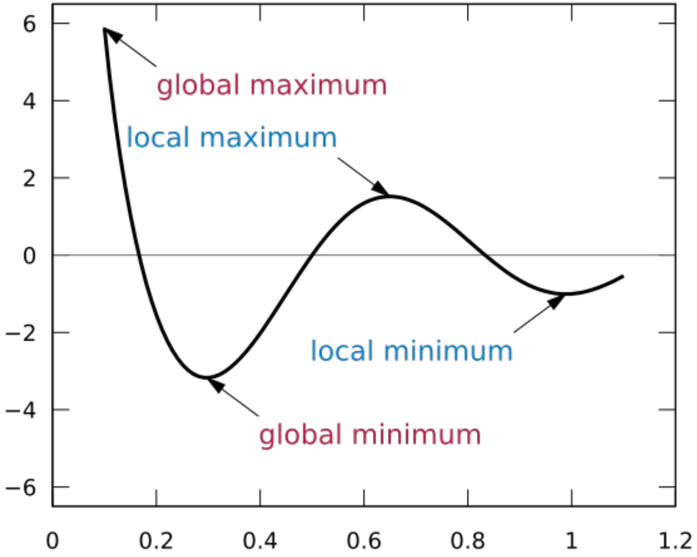
Images source: https://mnsgrg.com/2017/12/21/xavier-initialization/
https://en.wikipedia.org/wiki/Logistic_function
http://reference.wolfram.com/language/ref/Tanh.html
https://en.wikipedia.org/wiki/Rectifier_(neural_networks)

Stop Criterion



Local Minimum v.s. Global Minimum

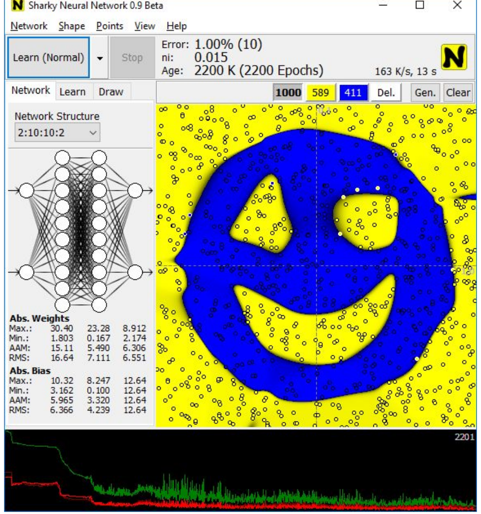




Img source: https://en.wikipedia.org/wiki/Maxima_and_minima

Overfitting

No overlapping data (artificial) ->

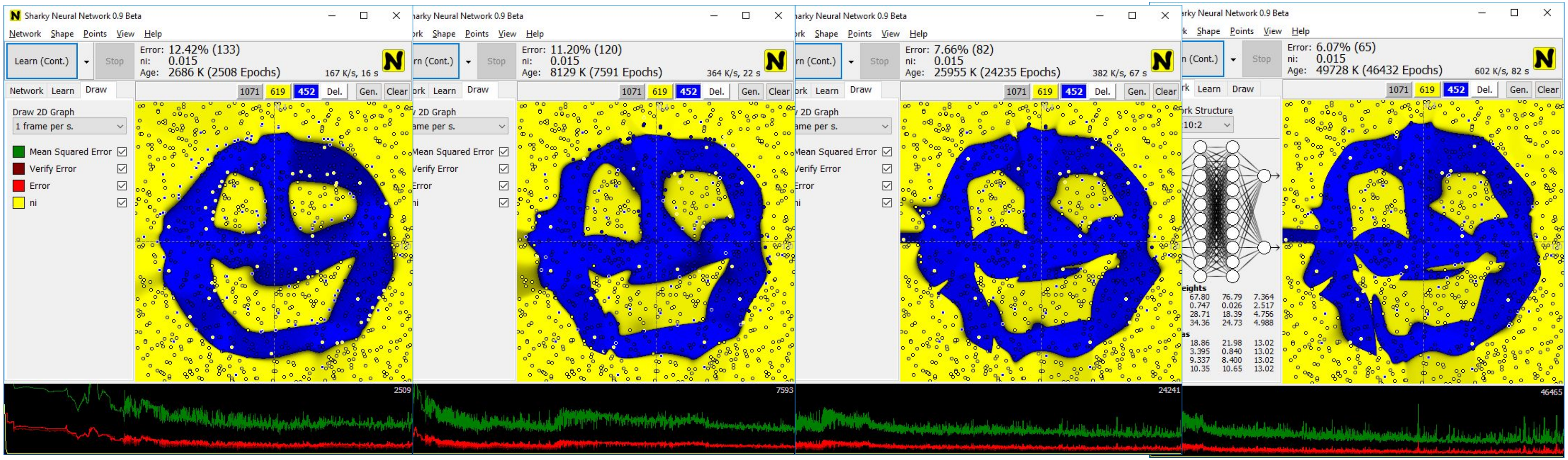


Error: 1.00% (10)
ni: 0.015
Age: 2200 K (2200 Epochs)
163 K/s, 13 s

Network Structure
2:10:10:2

```
Abs. Weights
Max.:   30.40   23.28   8.912
Min.:   1.803   0.167   2.174
AAM:    15.11   5.490   6.306
RMS:    16.64   7.111   6.551
Abs. Bias
Max.:   10.32   8.247   12.64
Min.:   3.162   0.100   12.64
AAM:    5.965   3.320   12.64
RMS:    6.366   4.239   12.64
```

Longer & longer train on overlapping data:



Error: 12.42% (133)
ni: 0.015
Age: 2686 K (2508 Epochs)
167 K/s, 16 s

Draw 2D Graph
1 frame per s.

🟩 Mean Squared Error ☑
🟥 Verify Error ☑
🟥 Error ☑
🟨 ni ☑

Error: 11.20% (120)
ni: 0.015
Age: 8129 K (7591 Epochs)
364 K/s, 22 s

Error: 7.66% (82)
ni: 0.015
Age: 25955 K (24235 Epochs)
382 K/s, 67 s

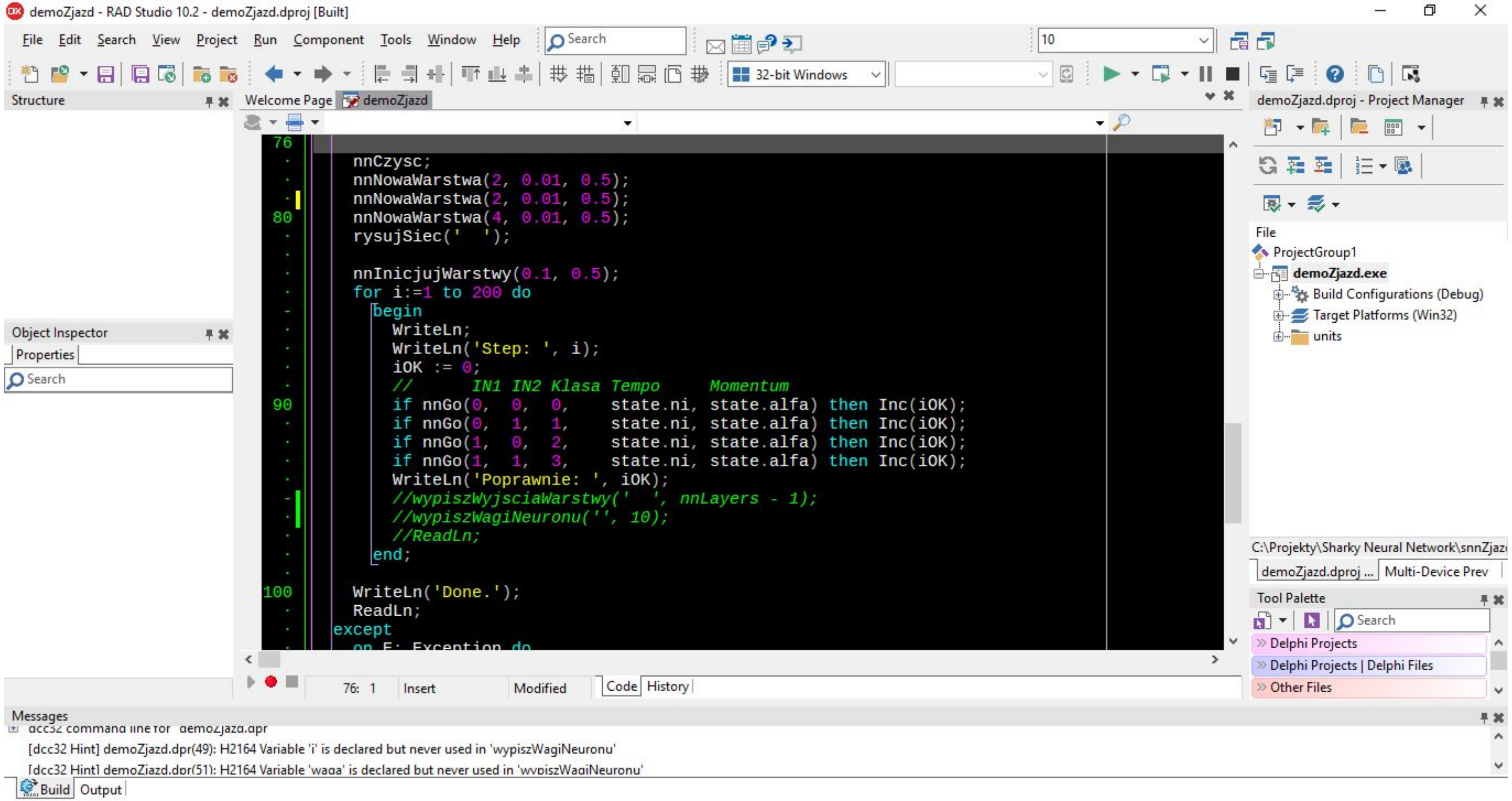
Error: 6.07% (65)
ni: 0.015
Age: 49728 K (46432 Epochs)
602 K/s, 82 s

```
eights
67.80   76.79   7.364
0.747   0.026   2.517
28.71   18.39   4.756
34.36   24.73   4.988
as
18.86   21.98   13.02
3.395   0.840   13.02
9.337   8.400   13.02
10.35   10.65   13.02
```

The Data is The Key

- Split for Test & Train
- Do Augmentation
- Correct/Remove Outliers and Missing Values
- Normalize
- Balance
- Features Engineering
- Monitor Sources, Quality, Variances, ...
- ...

Demo + Source Code



```
    nnCzysc;
    nnNowaWarstwa(2, 0.01, 0.5);
    nnNowaWarstwa(2, 0.01, 0.5);
    nnNowaWarstwa(4, 0.01, 0.5);
    rysujSiec('  ');

    nnInicjujWarstwy(0.1, 0.5);
    for i:=1 to 200 do
      begin
        WriteLn;
        WriteLn('Step: ', i);
        iOK := 0;
        //       IN1 IN2 Klasa Tempo     Momentum
        if nnGo(0,  0,  0,    state.ni, state.alfa) then Inc(iOK);
        if nnGo(0,  1,  1,    state.ni, state.alfa) then Inc(iOK);
        if nnGo(1,  0,  2,    state.ni, state.alfa) then Inc(iOK);
        if nnGo(1,  1,  3,    state.ni, state.alfa) then Inc(iOK);
        WriteLn('Poprawnie: ', iOK);
        //wypiszWyjsciaWarstwy('  ', nnLayers - 1);
        //wypiszWagiNeuronu('', 10);
        //ReadLn;
      end;

    WriteLn('Done.');
    ReadLn;
  except
    on E: Exception do
```

```
dcc32 command line for "demoZjazd.dpr"
[dcc32 Hint] demoZjazd.dpr(49): H2164 Variable 'i' is declared but never used in 'wypiszWagiNeuronu'
[dcc32 Hint] demoZjazd.dpr(51): H2164 Variable 'waga' is declared but never used in 'wypiszWagiNeuronu'
```