04

Problem Solving
&
Machine & Deep Learning

WORLD NEWS **Click**

HOGESCHOOL ROTTERDAM

Home | Episodes | Clips | Gadgets | Spencer Kelly | About Click

The Secrets Behind AI

We look at the impact of emerging technology in Kenya, Nairobi - and analyse the effect of what artificial intelligence will have on the world around us.

30 minutes

Show more

Last on

BBC WORLD NEWS

Fri 23 Nov 2018

02:30

Local time

BBC WORLD NEWS LATIN AMERICA & NORTH AMERICA ONLY

More episodes

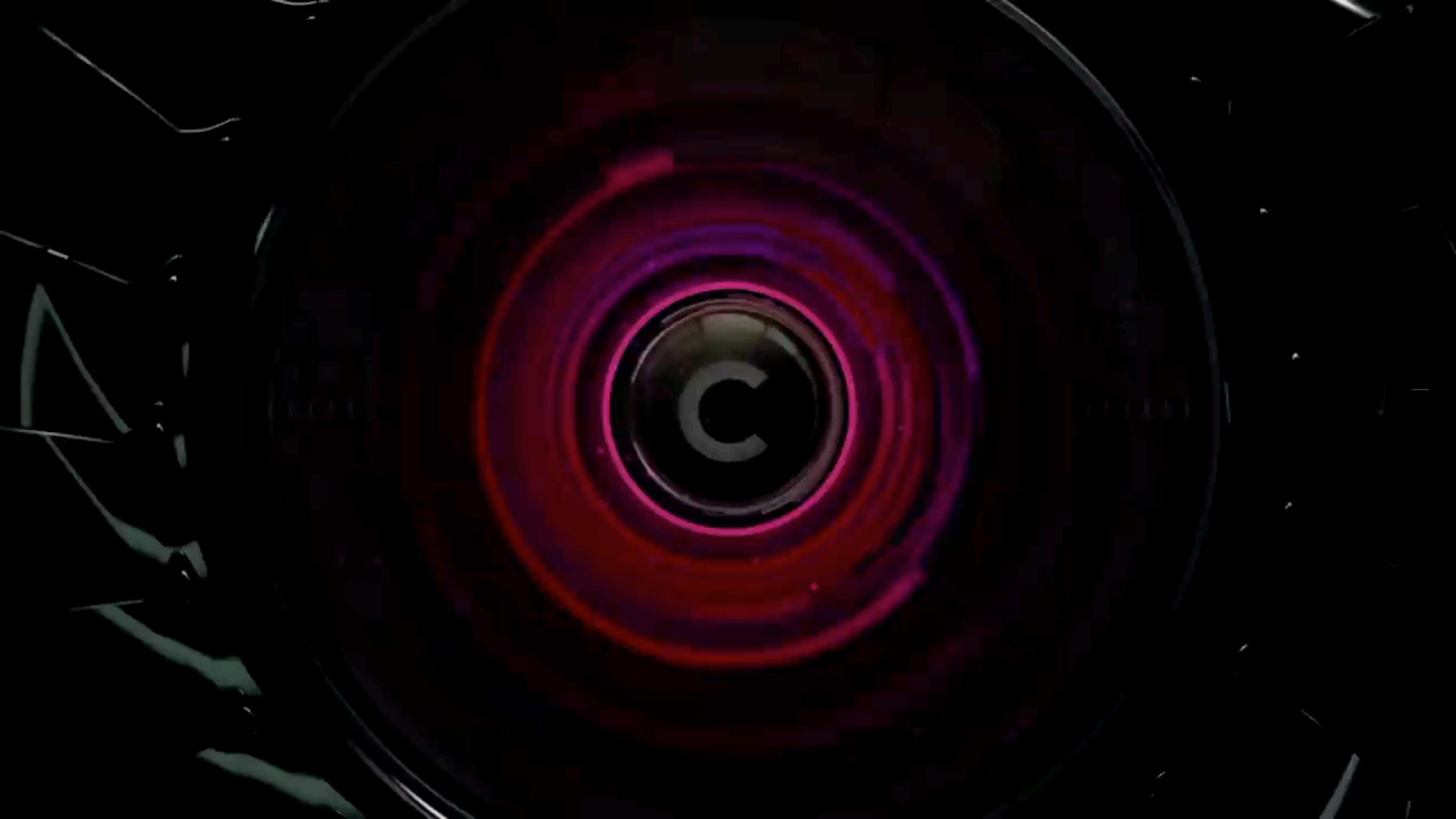
PREVIOUS

Domo Arigato Mr Roboto

NEXT

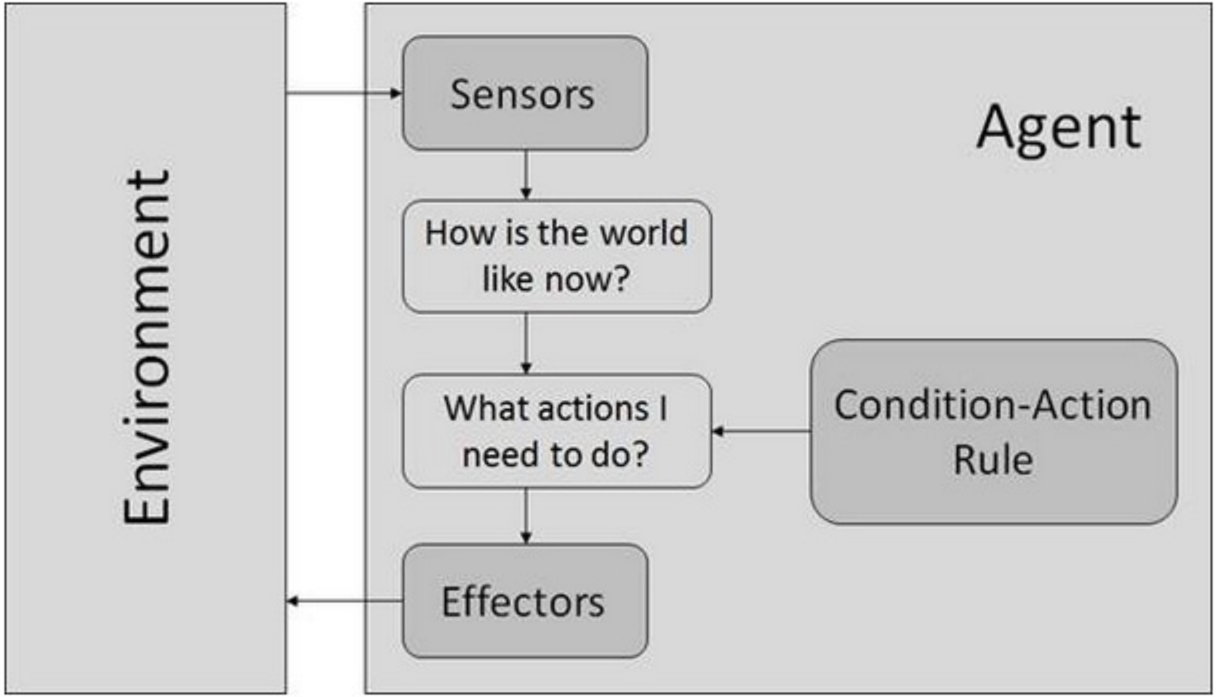
Top Quality Fake News

See all episodes from Click



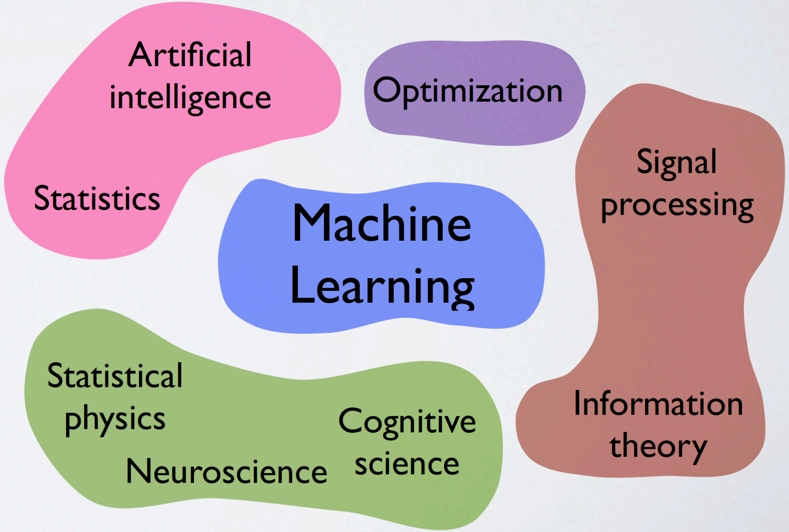
WHAT IS MACHINE LEARNING?

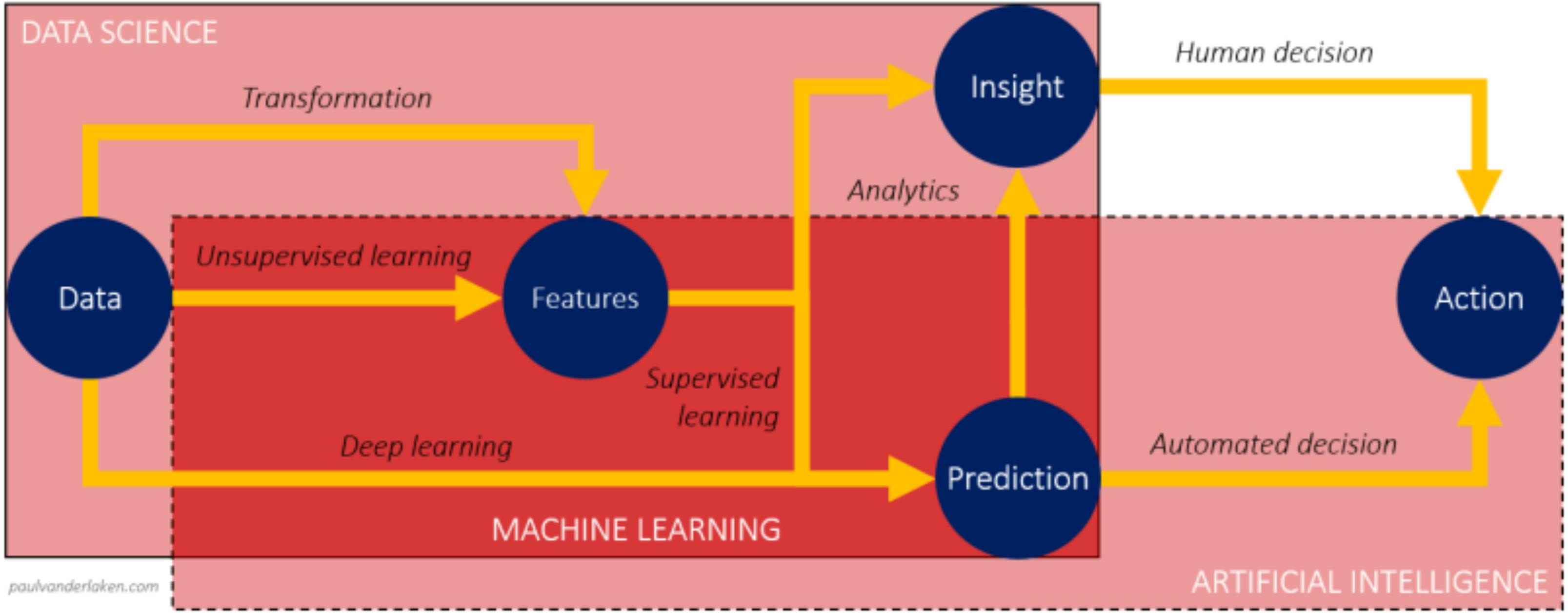
- "The science of getting computers to act without being explicitly programmed" - Andrew Ng (Stanford/Coursera)

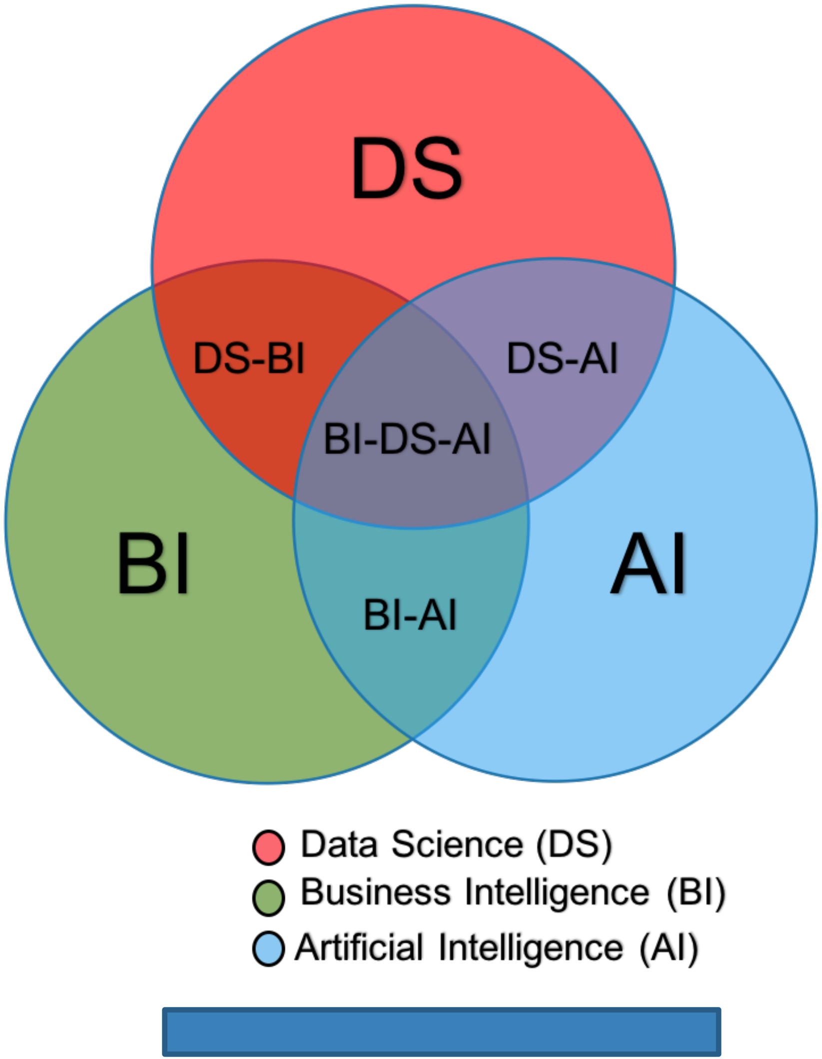

Environment
Sensors
Agent
How is the world like now?
What actions I need to do?
Condition-Action Rule
Effectors

What is machine learning?

- part of standard computer science curriculum since the 90s
- inferring knowledge from data
- generalizing to unseen data
- usually no parametric model assumptions
- emphasizing the computational challenges

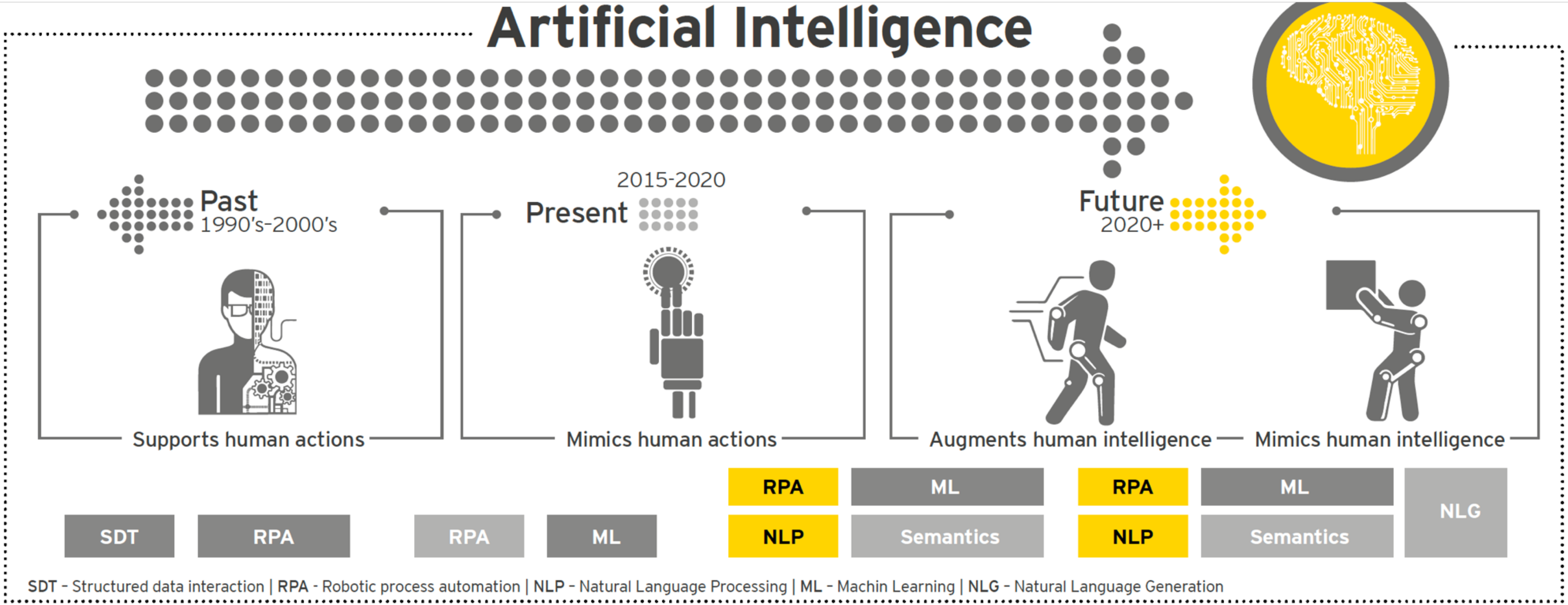



DATA SCIENCE
Transformation
Human decision
Insight
Analytics
Unsupervised learning
Data
Features
Action
Supervised
learning
Deep learning
Automated decision
Prediction
MACHINE LEARNING
ARTIFICIAL INTELLIGENCE
paulvanderlaken.com



Methods/Techniques	Discipline
Unsupervised Machine Learning	AI
Reinforcement Learning	AI
Reporting/Decision Support	BI
Multi-Dimensional Analysis	BI
Natural Language (Query)	BI-AI
Data Management	BI-DS-AI
Big Data Management	BI-DS-AI
Inferential Statistics	DS
Predictive Modeling	DS
Supervised Machine Learning	DS-AI
Descriptive Statistics	DS-BI
Data Discover/Query	DS-BI
Data Visualization	DS-BI

HOGESCHOOL ROTTERDAM


Artificial Intelligence
Past
1990's-2000's
2015-2020
Present
Future
2020+
Supports human actions
Mimics human actions
Augments human intelligence
Mimics human intelligence
SDT
RPA
RPA
ML
RPA
NLP
ML
Semantics
RPA
NLP
ML
Semantics
NLG
SDT - Structured data interaction | RPA - Robotic process automation | NLP - Natural Language Processing | ML - Machin Learning | NLG - Natural Language Generation

Understand data

Understand unstructured information and weave together structured and unstructured data

Reasoning ability

Reason, grasp underlying concepts, form hypotheses and infer to extract ideas

Learn continuously

Make sense of constantly changing data, interactions and outcomes

Human-like interaction

Interact with humans in a natural way by leveraging NLP, NLG and text analytics

Actionable insights

Deliver meaningful insights, can make decisions and call up on intelligent automation services to execute

software

algorithms in decision making

fuzzy boundaries

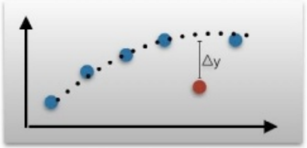
rule-based decision making

if condition fulfilled *then*
activity 1
else
activity 2

boolean data
(yes or no)

Examples:
- phone notification
- time- or threshold-based alarms
- simple pattern matching

statistical reasoning

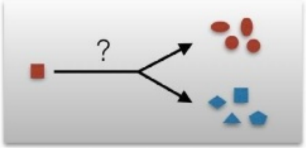


simple regression

numerical data
allowing for curve fitting

Examples:
- extra- and interpolation
- outlier detection
- predictive maintenance

machine learning

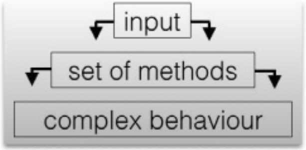


classification tasks

arbitrary data
that needs to be abstracted into numbers

Examples:
- identification of relevant features from large input datasets
- quality control using various metrics

artificial intelligence



dynamic adaptation to novelty

autonomous selection of best methodology when presented with **arbitrary data**

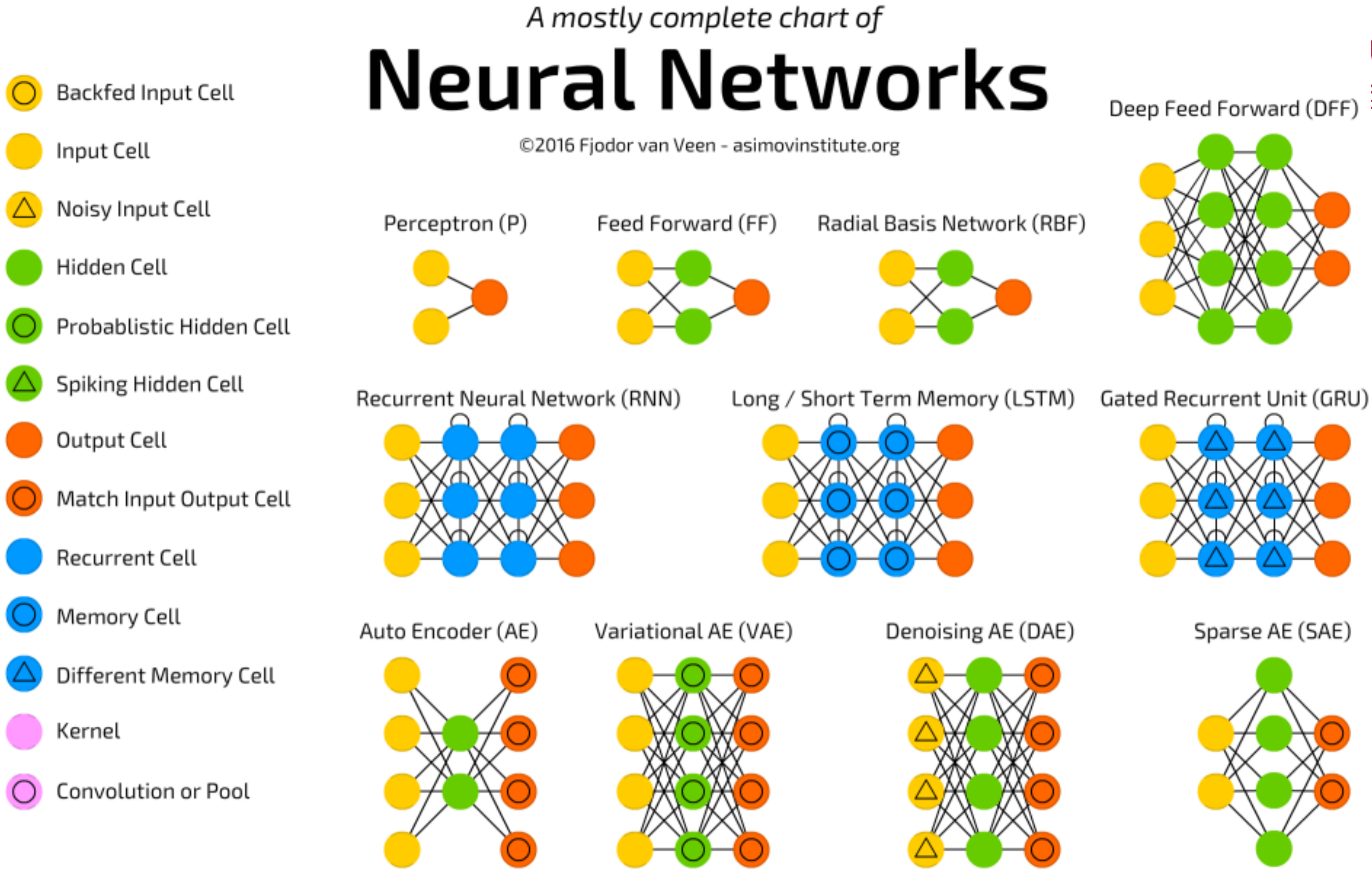
Examples:
- autonomous vehicles
- human-like conversational skills
- intelligent digital assistant

every programmer

data science types

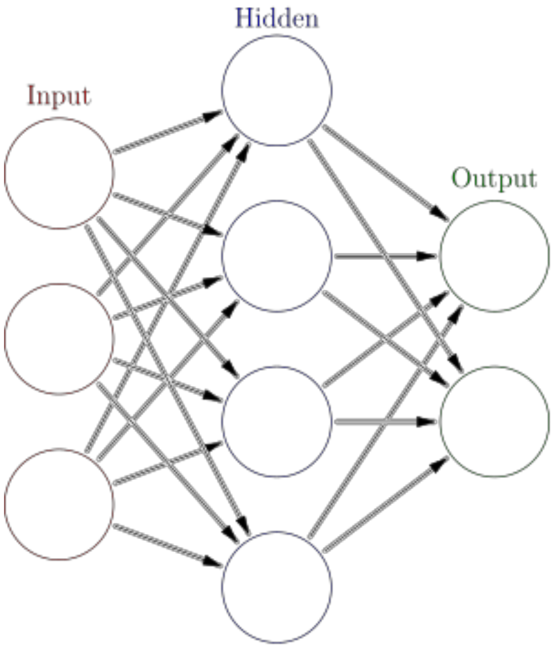
complex systems specialists

HOGESCHOOL ROTTERDAM



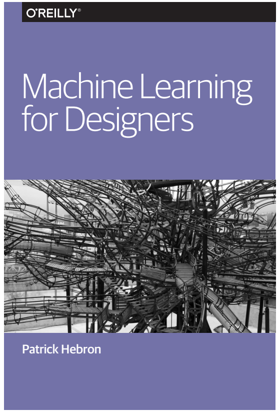
A mostly complete chart of
Neural Networks
©2016 Fjodor van Veen - asimovinstitute.org
Backfed Input Cell
Input Cell
Noisy Input Cell
Hidden Cell
Probablistic Hidden Cell
Spiking Hidden Cell
Output Cell
Match Input Output Cell
Recurrent Cell
Memory Cell
Different Memory Cell
Kernel
Convolution or Pool
Perceptron (P)
Feed Forward (FF)
Radial Basis Network (RBF)
Deep Feed Forward (DFF)
Recurrent Neural Network (RNN)
Long / Short Term Memory (LSTM)
Gated Recurrent Unit (GRU)
Auto Encoder (AE)
Variational AE (VAE)
Denoising AE (DAE)
Sparse AE (SAE)

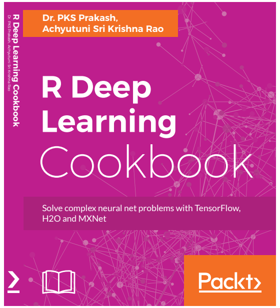
BRIEF INTRODUCTION
Machine Learning (ML) with Artificial Neural Networks (ANN)



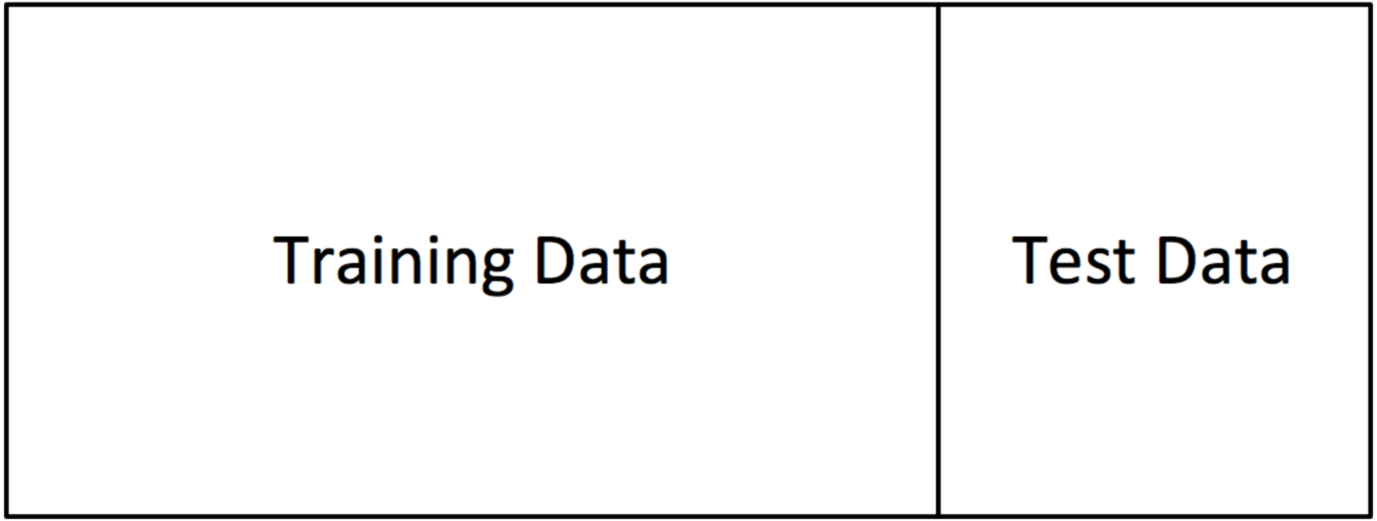
1. Activation function
2. Weights
3. Cost function
4. Learning algorithm

Live Demo





Full Dataset:



Labradoodle or fried chicken

Puppy or bagel

Sheepdog or mop

Chihuahua or muffin

Barn owl or apple

Parrot or guacamole

Raw chicken or Donald Trump


IMAGES
FEATURES
MACHINE LEARNING
WHAT THE OBJECT IS

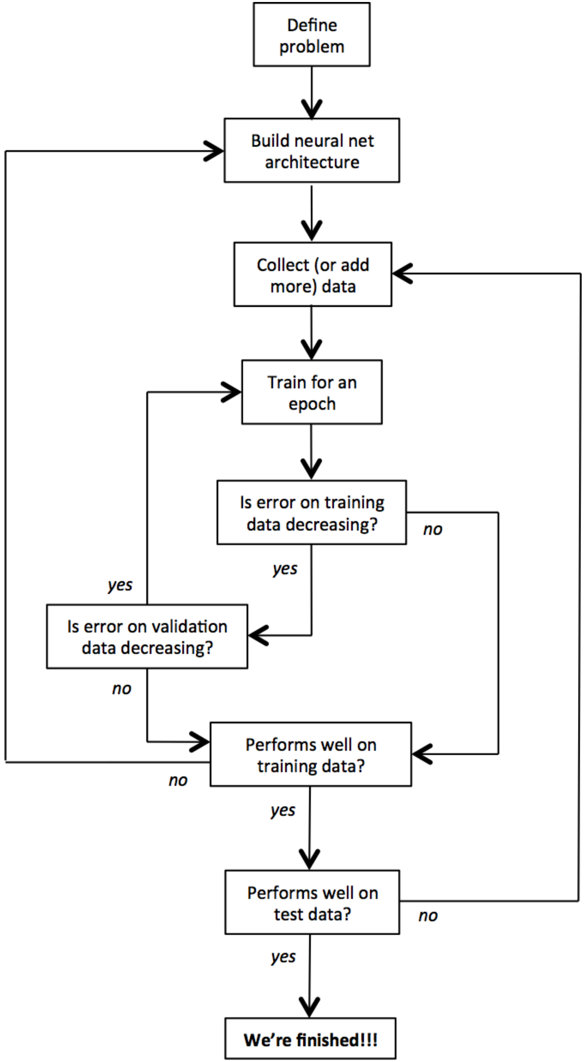
Deep learning is
a machine learning technique
that learns **features and tasks**
directly from data.

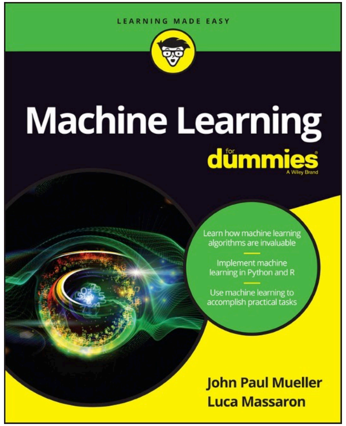
Data can be **images, text,** or **sound.**

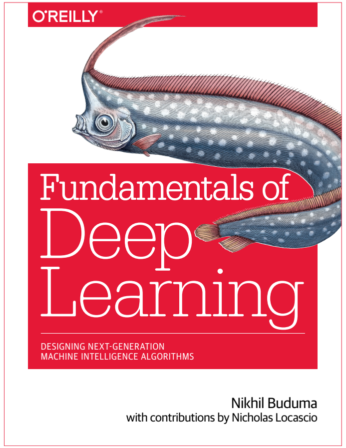
HOGESCHOOL ROTTERDAM

An algorithm is a

step by step process or recipe that **describes how to solve a problem** and/or **complete a task**, which will **always give** the **correct result**

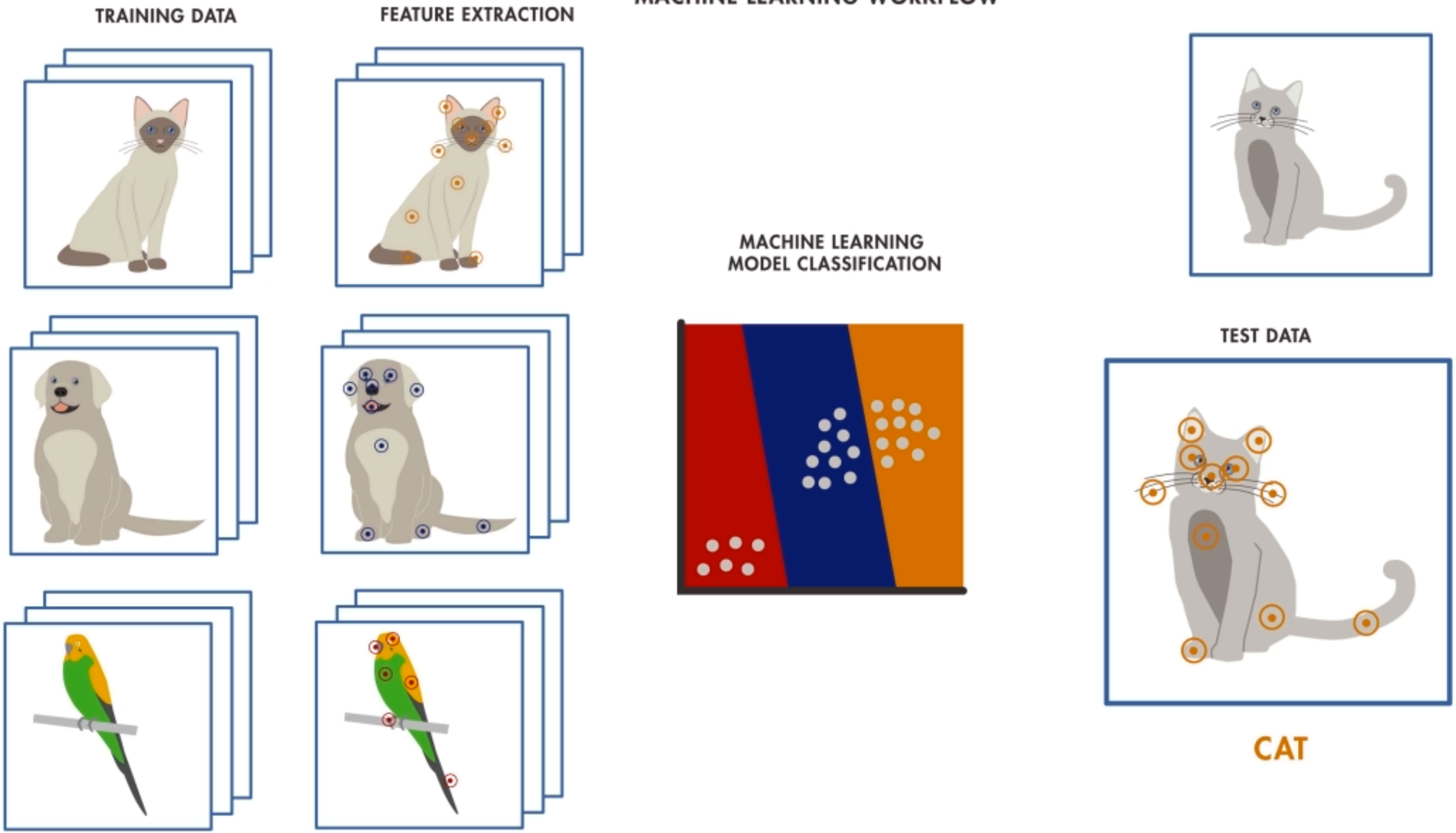

Define problem
Build neural net architecture
Collect (or add more) data
Train for an epoch
Is error on training data decreasing?
yes
no
Is error on validation data decreasing?
yes
no
Performs well on training data?
yes
no
Performs well on test data?
yes
no
We're finished!!!

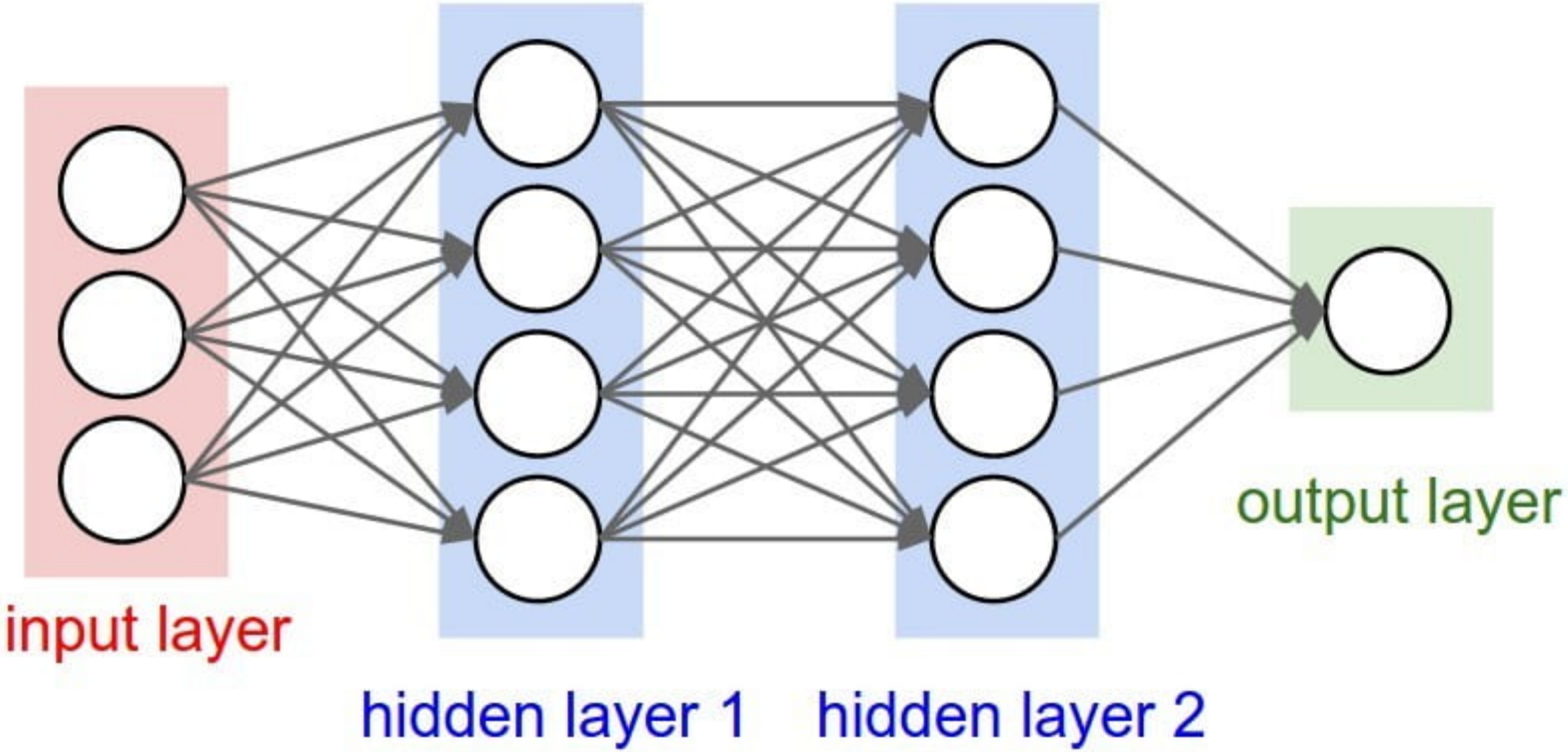

LEARNING MADE EASY
Machine Learning
for dummies
Learn how machine learning algorithms are invaluable
Implement machine learning in Python and R
Use machine learning to accomplish practical tasks
John Paul Mueller
Luca Massaron

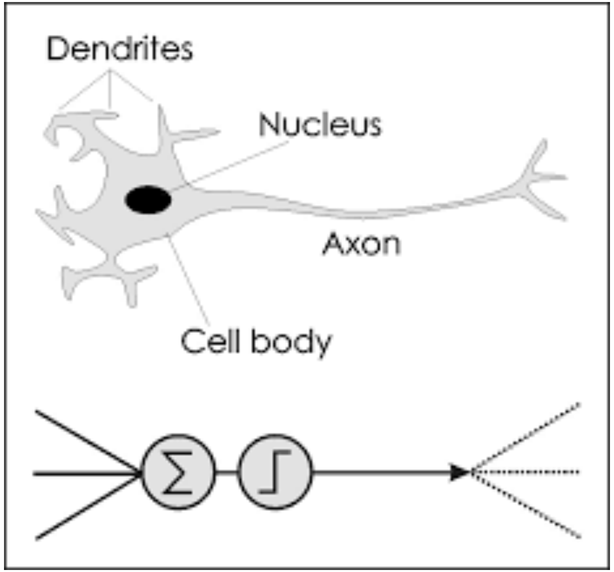

O'REILLY
Fundamentals of Deep Learning
DESIGNING NEXT-GENERATION MACHINE INTELLIGENCE ALGORITHMS
Nikhil Buduma
with contributions by Nicholas Locascio

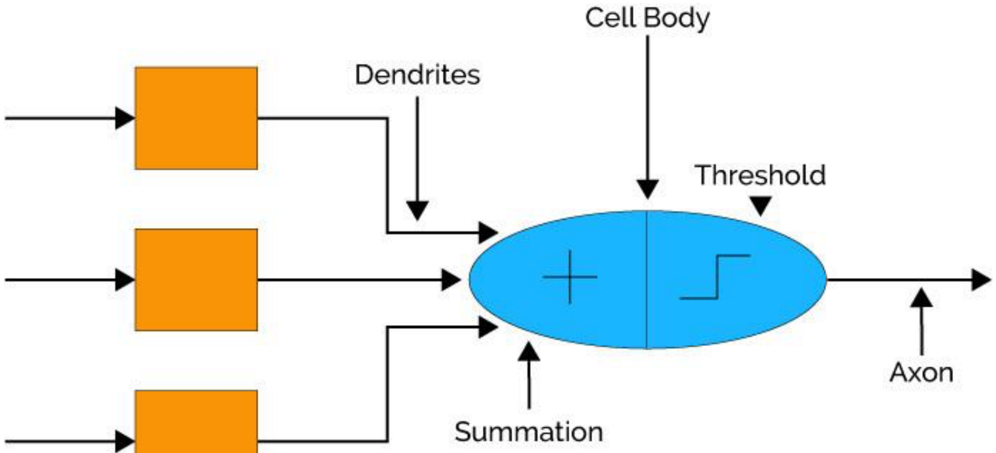
HOGESCHOOL ROTTERDAM

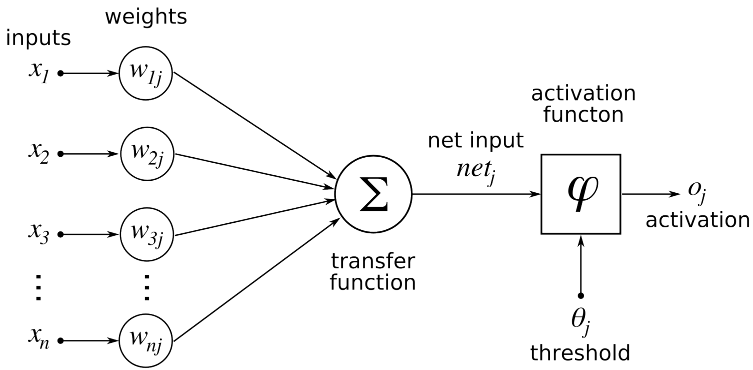
MACHINE LEARNING WORKFLOW



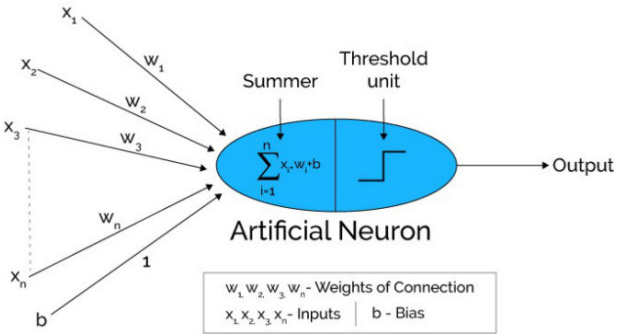

input layer
hidden layer 1
hidden layer 2
output layer


Dendrites
Nucleus
Axon
Cell body

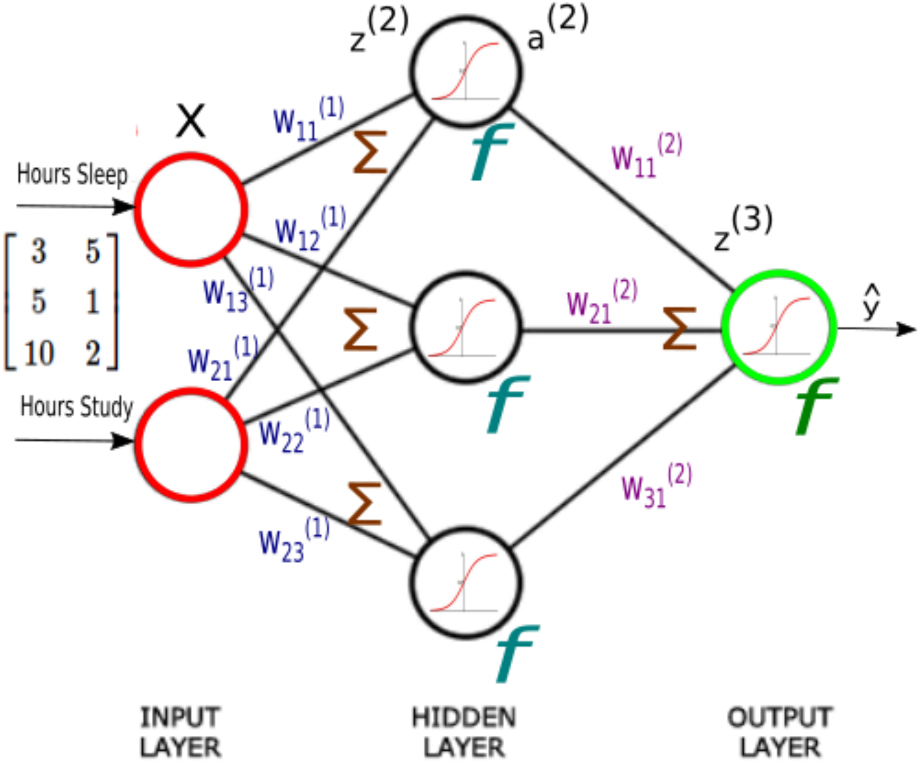

Cell Body
Dendrites
Threshold
Summation
Axon



inputs
weights
x_1
w_{1j}
x_2
w_{2j}
x_3
w_{3j}
x_n
w_{nj}
Σ
transfer function
net input
net_j
activation functon
φ
o_j
activation
θ_j
threshold



Summer
Threshold unit
$\sum_{i=1}^{n} x_i . w_i + b$
Output
Artificial Neuron
w_1, w_2, w_3, w_n - Weights of Connection
x_1, x_2, x_3, x_n - Inputs
b - Bias



X
Hours Sleep
Hours Study
$\begin{bmatrix} 3 & 5 \\ 5 & 1 \\ 10 & 2 \end{bmatrix}$
$W_{11}^{(1)}$
$W_{12}^{(1)}$
$W_{13}^{(1)}$
$W_{21}^{(1)}$
$W_{22}^{(1)}$
$W_{23}^{(1)}$
$z^{(2)}$
$a^{(2)}$
$W_{11}^{(2)}$
$W_{21}^{(2)}$
$W_{31}^{(2)}$
$z^{(3)}$
$\hat{y}$
INPUT LAYER
HIDDEN LAYER
OUTPUT LAYER

Backpropagation Learning Rule

Backpropagation Learning Rule

- Each weight changed by:

$$\Delta w_{ji} = \eta \delta_j o_i$$

$$\delta_j = o_j(1-o_j)(t_j - o_j) \quad \text{if } j \text{ is an output unit}$$

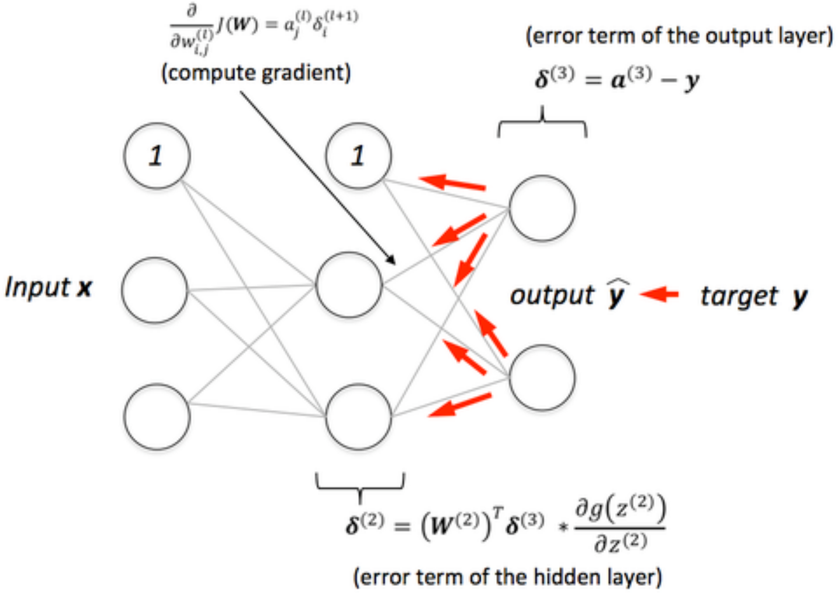
$$\delta_j = o_j(1-o_j)\sum_k \delta_k w_{kj} \quad \text{if } j \text{ is a hidden unit}$$

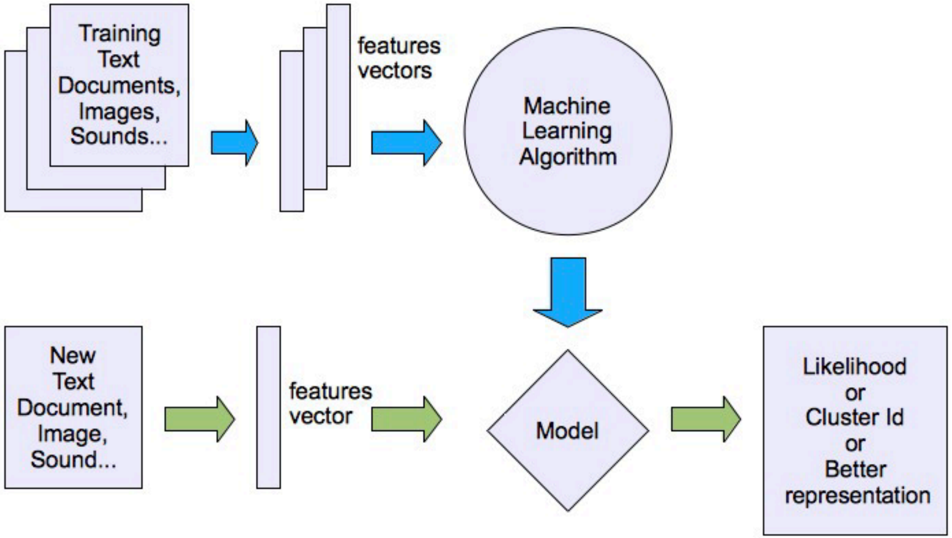
where η is a constant called the learning rate

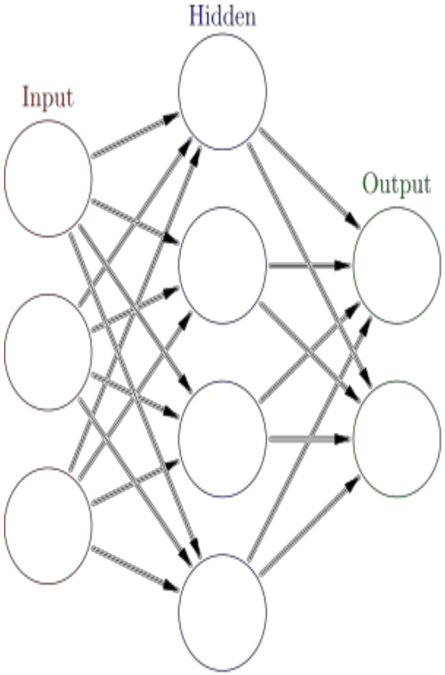
t_j is the correct teacher output for unit j

δ_j is the error measure for unit j

- Initialize weights (typically random!)
- Keep doing epochs
  - For each example **e** in training set do
    - **forward pass** to compute
      - O = neural-net-output(network,e)
      - miss = (T-O) at each output unit
    - **backward pass** to calculate deltas to weights
    - update all weights
  - end
- until tuning set error stops improving







HOGESCHOOL ROTTERDAM

m : Number of training examples

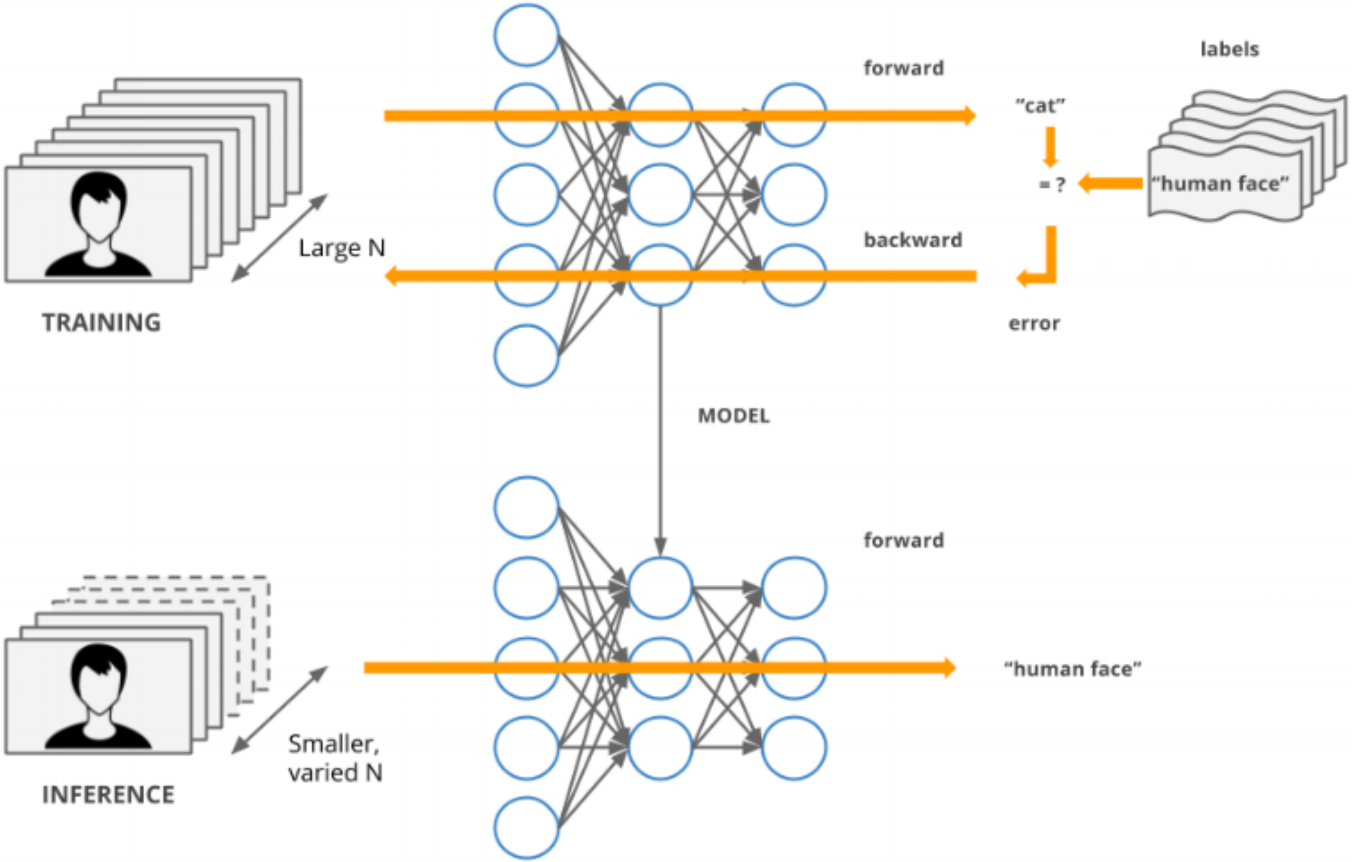
x : Input variables (Features)

y: Output variables (Targets)

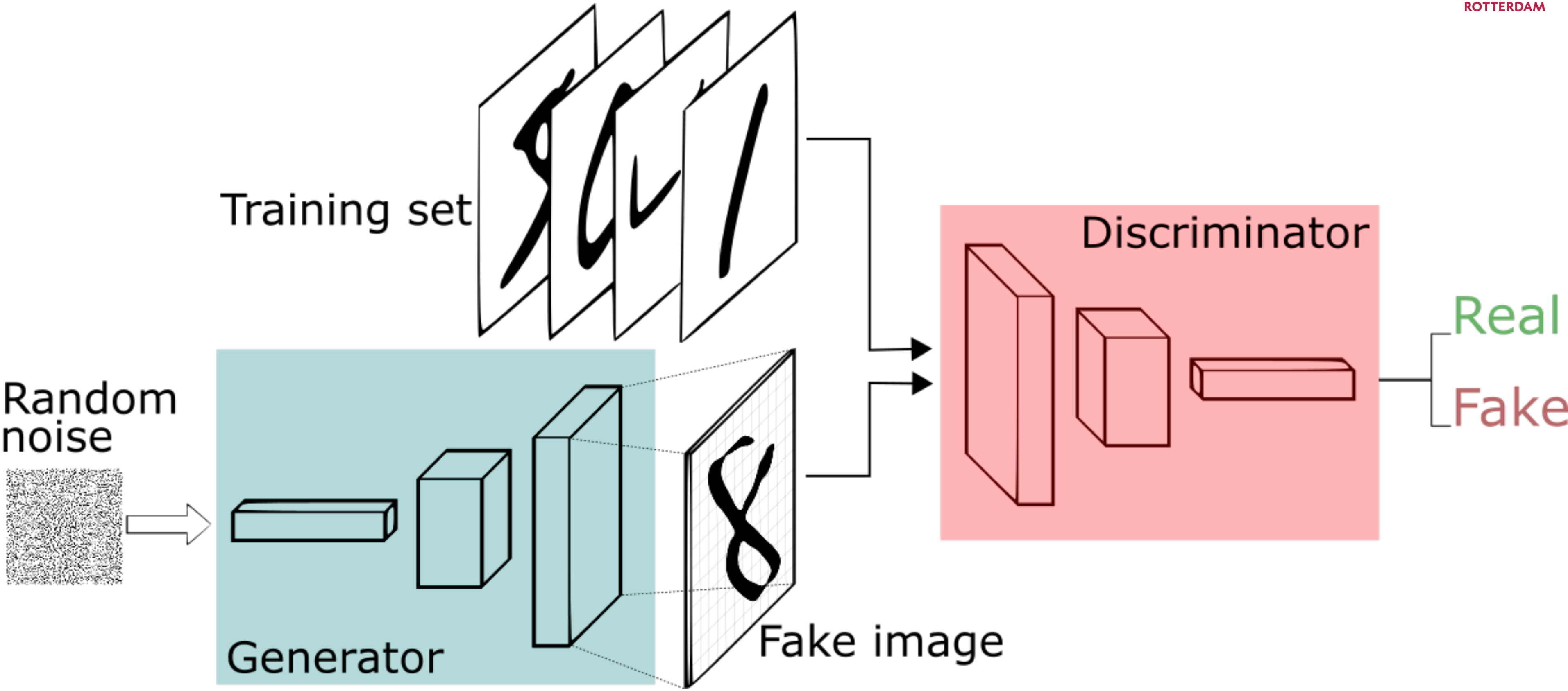
(x,y): Training Example (Represents 1 row on the table)

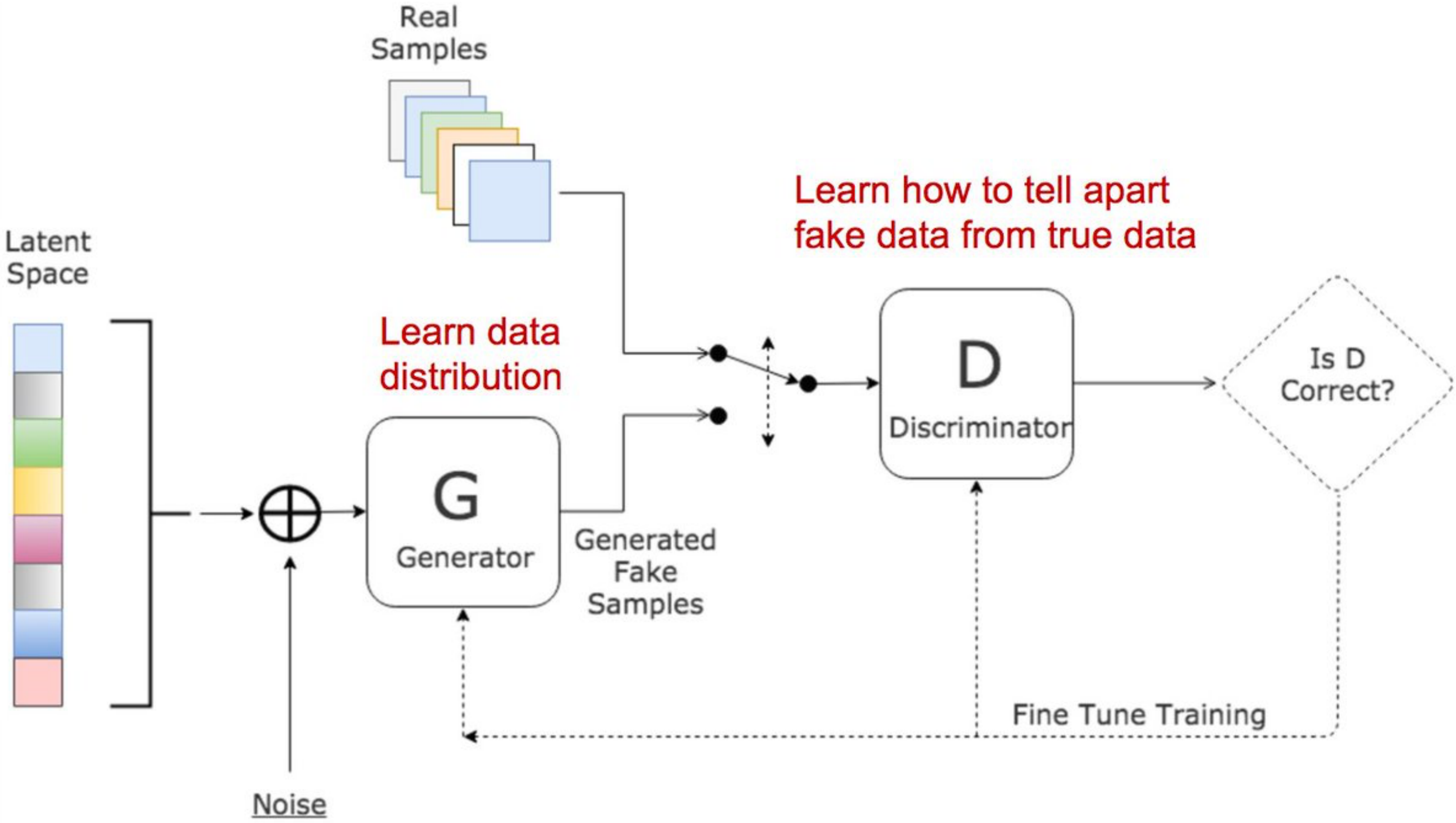
(x_i,y_i) : i_th training example (Represent's i_th row on the network)

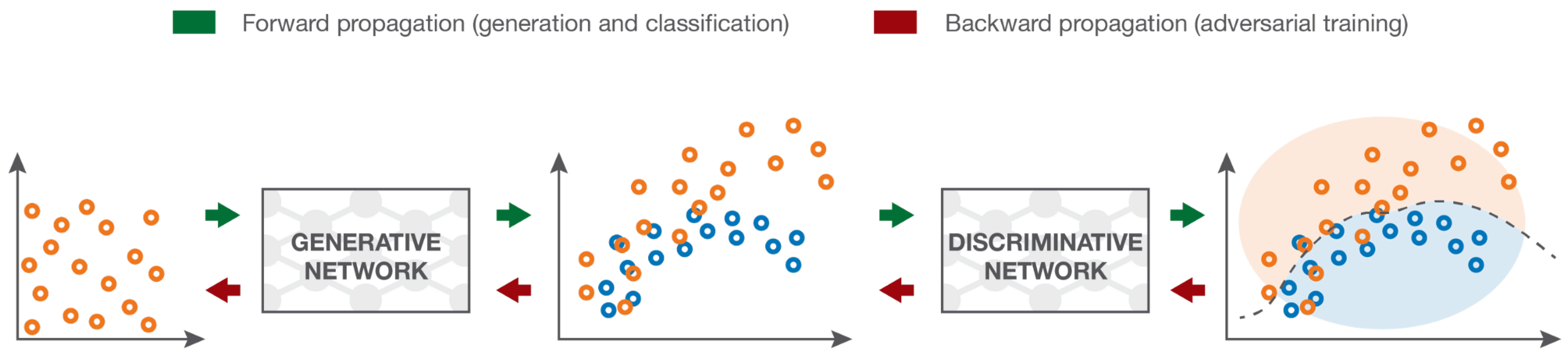
n : Number of features (Dimensionality of the input)


forward
labels
"cat"
= ?
"human face"
backward
Large N
TRAINING
error
MODEL
forward
"human face"
Smaller,
varied N
INFERENCE

generative adversarial networks

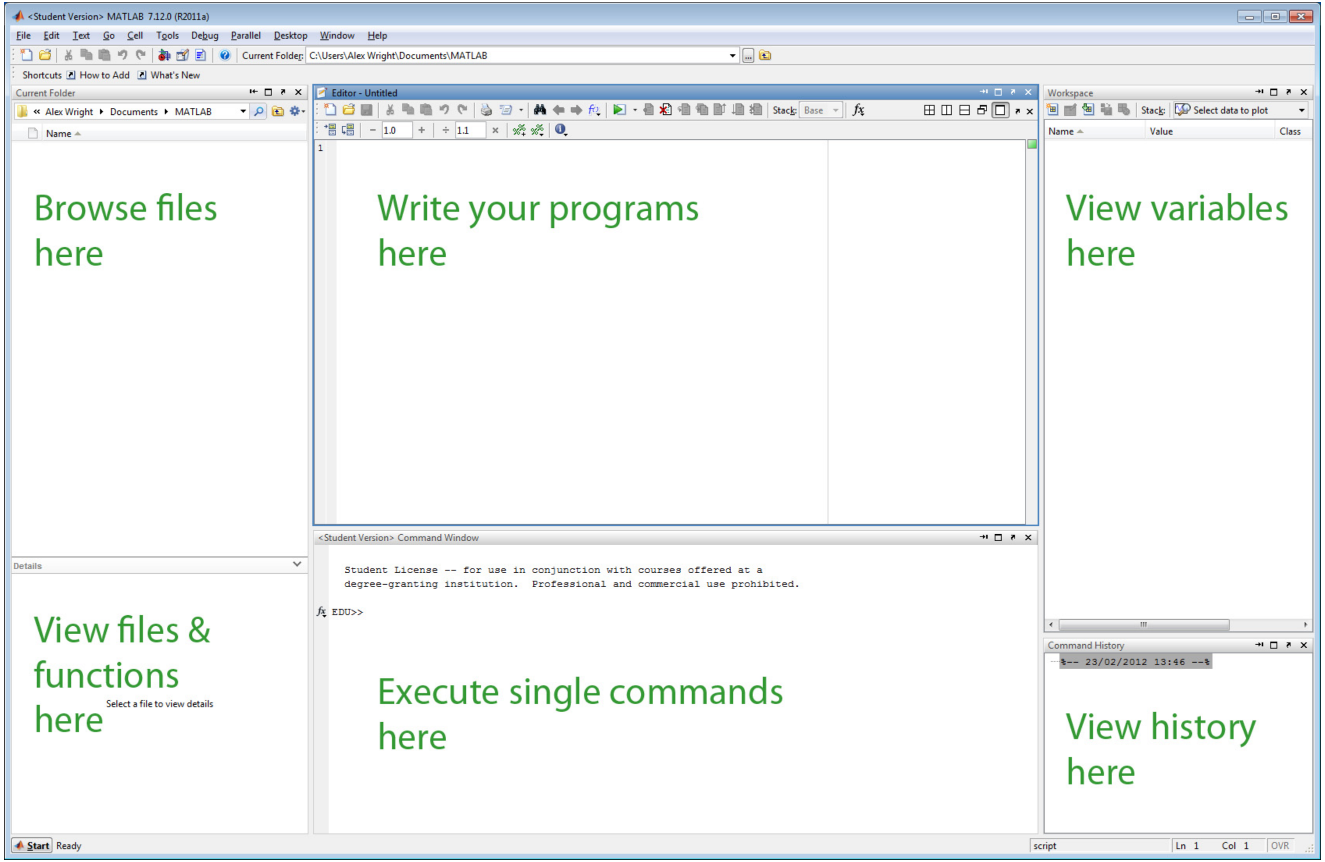

Training set
Discriminator
Real
Fake
Random noise
Generator
Fake image


Real Samples
Learn how to tell apart fake data from true data
Latent Space
Learn data distribution
D
Discriminator
Is D Correct?
G
Generator
Generated Fake Samples
Fine Tune Training
Noise



Input random variables.	The generative network is trained to **maximise** the final classification error.	The generated distribution and the true distribution are not compared directly.	The discriminative network is trained to **minimise** the final classification error.	The classification error is the basis metric for the training of both networks.

MATLAB IDE



<Student Version> MATLAB 7.12.0 (R2011a)
File Edit Text Go Cell Tools Debug Parallel Desktop Window Help
Current Folder: C:\Users\Alex Wright\Documents\MATLAB
Shortcuts How to Add What's New
Current Folder
Alex Wright Documents MATLAB
Name
Browse files here
Details
View files & functions here
Select a file to view details
Editor - Untitled
Stack: Base
Write your programs here
Workspace
Stack: Select data to plot
Name Value Class
View variables here
<Student Version> Command Window
Student License -- for use in conjunction with courses offered at a degree-granting institution. Professional and commercial use prohibited.
EDU>>
Execute single commands here
Command History
%-- 23/02/2012 13:46 --%
View history here
Start Ready
script Ln 1 Col 1 OVR

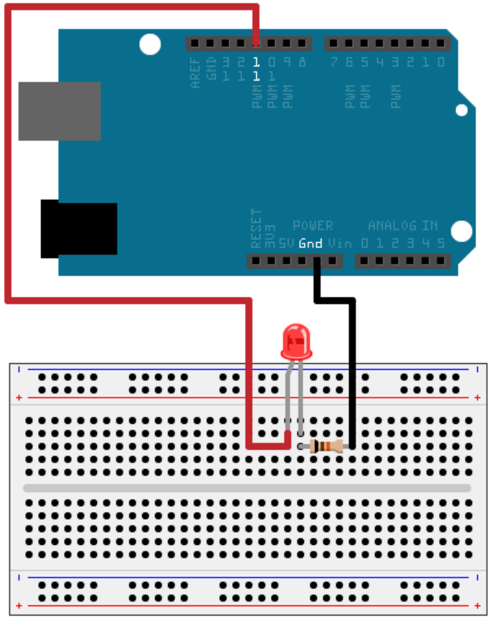
MATLAB IDE

Getting Started with MATLAB Support Package for Arduino Hardware

This example shows how to use MATLAB® Support Package for Arduino® Hardware to perform basic operations on tl
LEDs and playing sound on a speaker.

Hardware setup

Connect an LED to digital pin 11 on the Arduino hardware through a 1kOhm resistor.



https://nl.mathworks.com/help/supportpkg/arduinoio/examples/getting-started-with-matlab-support-package-for-arduino-hardware.html

Create an arduino object

```
a = arduino();
```

If you have more than one Arduino board connected, specify the port and board type.

```
clear a;
a = arduino('COM4', 'Uno');
```

Turn LED on and off

Write value 1 or true to digital pin 11 turns on the built-in LED and write a value of 0 or false t

```
writeDigitalPin(a, 'D11', 0);
pause(2);
writeDigitalPin(a, 'D11', 1);
```

Configure the LED to blink at a period of 1 second.

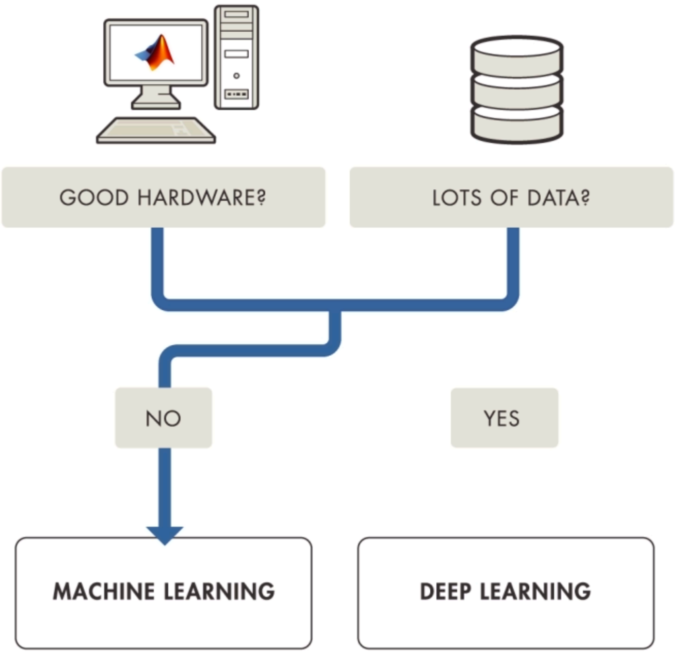
```
for i = 1:10
   writeDigitalPin(a, 'D11', 0);
   pause(0.5);
   writeDigitalPin(a, 'D11', 1);
   pause(0.5);
end
```

Brighten and dim LED

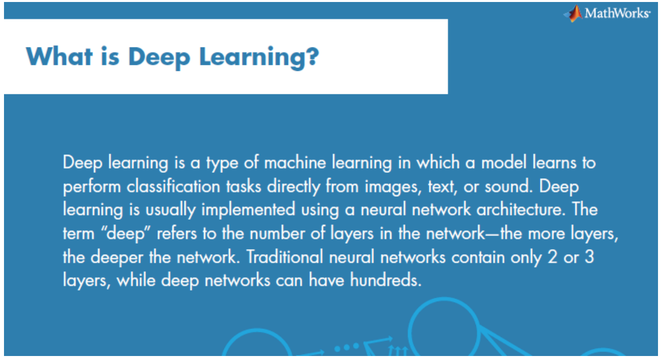
Send pulse signals of specified width to the PWM pins on the Arduino hardware. PWM signal
that the LED brightens and dims by dividing the max and min duty cycle for the pin by the nu

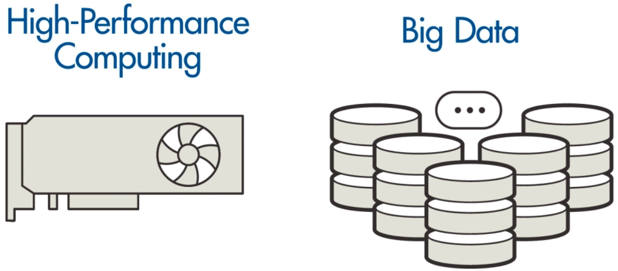
```
brightness_step = (1-0)/20;
for i = 1:20
   writePWMDutyCycle(a, 'D11', i*brightness_step);
   pause(0.1);
end

for i = 1:20
   writePWMDutyCycle(a, 'D11', 1-i*brightness_step);
   pause(0.1);
end
```

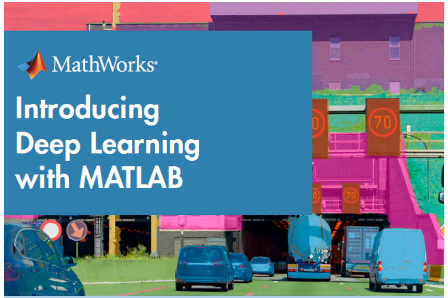
MATLAB IDE



	Machine Learning	Deep Learning
Training dataset	Small	Large
Choose your own features	Yes	No
# of classifiers available	Many	Few
Training time	Short	Long







https://nl.mathworks.com/campaigns/offers/deep-learning-with-matlab.html

https://nl.mathworks.com/videos/introduction-to-deep-learning-what-is-deep-learning--1489502328819.html

3Ms for Instruction

Reviews of Maple, Mathematica, and Matlab

By Norman Chonacky and David Winch

MOST *CiSE* READERS HAVE PROBABLY USED MAPLE, MATHEMATICA, OR MATLAB FOR SEVERAL YEARS. WITH THIS REVIEW SERIES, OUR GOAL IS TO HELP YOU NOW DECIDE WHETHER ONE OF THE OTHERS IS BETTER SUITED TO YOUR TEMPERAMENT and current practice than your original choice. For those of you new to integrative computing packages, our goal is to enable you to make an informed first choice.

In this installment, we begin to examine how these tools serve the professional work of undergraduate education. Within this context, we'd like to raise several significant issues for those teaching undergraduates to be scientists and engineers. We point to some exemplary materials and offer our own paradigms for major educational uses, which provide a framework for discussing the packages and drawing some implications for those issues in a concluding installment. In subsequent issues, we'll explore how the tools serve scientific and engineering research and communication.

Undergraduate Education

We begin with the premise that science and engineering undergraduates should have experience in using modern computational tools. Indeed, this is already an explicit criterion for engineering schools' curricula in the US as prescribed by the Accreditation Board for Engineering and Technology (ABET; www.abet.org/criteria.html). In this article, we examine the extent to which these tool packages so qualify: What kinds of computational experiences with them are appropriate for undergraduate students?

We're aware of the multiple goals that educational uses of computing technology must serve, as well as the challenge they present to a fair evaluation of computing software. Foremost in our minds as instructors experienced in the design of electronic instructional materials is the importance of appearance, simplicity, and user-interface functionality to the success of such materials. Yet, there are several types of user interfaces that connect users to different computing tasks according to different educational goals. This begs several questions: What are some major educational goals for science and engineering undergraduates? How are specific computing tasks related to those goals? How does each of the three productivity packages realize the required computations?

Undergraduates have a variety of learning styles and abilities, and they must simultaneously master material while learning how to learn. Ease of use in the packages' user interfaces as well as their adaptability to the variety of interactive mechanisms used in educational applications are key issues. Keep in mind, however, that the way

and degree to whi
tant depends on w
as well as the goals
College and un
must be judicious
tensity of develop
undertake in creati
terials, with respe
and resources they
do these tool packa
rials-development
will likely perforr
cient are they wher
are required for n
expensive are the
equally important,
Our approach to
to describe the fun
other elements the
and allow you to jud
on your values an
this, we depart fron
tures devoid of us
setting contexts in a
both real and ideal
pally drawn the real
company's Web sit
mented in only on
ages. By examining
software, albeit dev
tional purpose tha
your own, we hope
that help you envisi
you create could w
From our persp
ples provide a con
which we can refe
how each package v
idealized example.
ample as a paradigm
rected to one of th

Julia (programming language) - Wikipedia 15/05/2018, 22:55

WIKIPEDIA

Julia (programming language)

Julia is a high-level dynamic programming language designed to address the needs of high-performance numerical analysis and computational science, without the typical need of separate compilation to be fast, while also being effective for general-purpose programming,[15][16][17][18] web use[19][20] or as a specification language.[21]

Distinctive aspects of Julia's design include a type system with parametric polymorphism and types in a fully dynamic programming language and multiple dispatch as its core programming paradigm. It allows concurrent, parallel and distributed computing, and direct calling of C and Fortran libraries without glue code.

Julia is garbage-collected,[22] uses eager evaluation and includes efficient libraries for floating-point calculations, linear algebra, random number generation, fast Fourier transforms (using FFTW but only in current release versions; one of the library dependencies moved out of the standard library to a package[23] because it is GPL licensed, and thus will not included in Julia 1.0 by default) and regular expression matching.

Julia

Official Julia logo

Paradigm	Multi-paradigm: multiple dispatch ("object-oriented"), procedural, functional, meta, multistaged[1]
Designed by	Jeff Bezanson, Alan Edelman, Stefan Karpinski, Viral B. Shah
Developer	Jeff Bezanson, Stefan Karpinski, Viral B. Shah, and other contributors[2][3]
First appeared	2012[4]
Stable release	0.6.2[5] / 17 December 2017[6] 0.5.2 / 8 May 2017[7][8]
Preview release	0.7.0-DEV / daily updates
Typing discipline	Dynamic, nominative, parametric
Implementation	Julia, C, Scheme

Contents

History
Language features
Interaction
Use with other languages
Implementation
Current and future platforms
Julia2C source-to-source compiler
Julia Computing company
See also
Notes
References
External links

History

Work on Julia was started in 2009 by Jeff Bezanson, Stefan Karpinski, Viral B. Shah, and Alan Edelman who set out to create a language that was both high-level and fast. On 14 February 2012 the team launched[24] a website with

https://en.wikipedia.org/wiki/Julia_(programming_language) Page 1 of 8

TECHNOLOGY REVIEWS

Editor: Norman Chonacky, CiSE-Editor@aip.org

TRUCTION, PART 2:

, AND MATLAB

avid Winch

ITH THIS TECHNOLOGY REVIEW IS TO PRE-
MEWORK THAT HELPS EDUCATORS MAKE
L COMPARISON OF MAPLE, MATHEMATICA,
IDIDATE COMPUTATIONAL PRODUCTIVITY
ictional pro-
e to our pro-
of our own,
w. In the first
common set
understand-
as well as an
example—
framework.
lar subset of
ience and en-
garding com-
his issue, we
uilding strat-
common fea-
ally describe
how Maple,
vork.
es and to re-
engineering
rical, rather
ional exam-
ckages per-
One cost of
't discuss all
omputation
natica and
ic computa-
Maple.) Al-
choice based
eophyte and
educational
erviewed, it

highlights a major compromise, one of many, needed to create this type of technology review. As an experiment, we set out to give a broad scope of readers the material they'll need to address educational issues, along with helpful and concise evaluation guidance. We hope we've struck a proper balance, avoiding both superficiality and technicality.

Development and Delivery Environments

What's it like to work with these packages? Users who wish to create or modify content must work within the associated development environments. Such users will be both faculty developing educational materials and students writing computational code, the only exception being students using applications mediated by custom-created, application-specific graphical user interfaces (GUIs).

Both Maple and Mathematica supply standard interfaces for their development environments that are already GUIs of a kind. These consist of book-like content windows that hold interactive text and graphics; these content windows also have pull-down menus from a menu bar and palettes of tools. In Mathematica, these palettes are movable and can be custom-created as part of the development environment or attached to applications. Figures 1 and 2 are screenshots of the Mathematica and Maple development interfaces, respectively.

Matlab's development environment is quite different. Basically, it has a command line displayed in one of several windows. The main, circumscribing window (called the Desktop) has pull-down menus from an overhead menu bar. The default Desktop configuration, shown in Figure 3, is subdivided into several partitions, each of which is itself a resizable window. The partitions contain a command line, a command stack, and a directory tree.

Developers using Mathematica or Maple enter the computing objects—such as variables, operations, descriptive text, and so on—into segmented cells. In Maple, these have a single logical level, whereas they can be nested hierarchically in Mathematica. These cells extend the command-line concept by encapsulating commands, but they also integrate narrative text, making their aggregate—the Mathematica Notebook or the Maple Worksheet—similar to interactive books. Any entities the developer creates in a session, such as variable names or session histories, are maintained implicitly by the system; however, commands, sometimes several commands, are needed to explicate them.

In contrast, developers working in the Matlab environment use a conventional command line. They encapsulate commands by placing them in separate files, which is one reason to have a file directory partition visible. In general,

ConvNetJS
Deep Learning in your browser

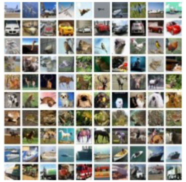
Intro | Deep Learning Resources | Getting Started | Documentation

ConvNetJS is a Javascript library for training Deep Learning models (Neural Networks) entirely in your browser. Open a tab and you're training. No software requirements, no compilers, no installations, no GPUs, no sweat.

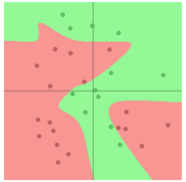
Browser Demos

Classify **MNIST** digits with a Convolutional Neural Network

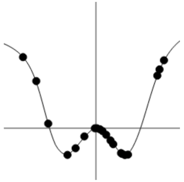
Classify **CIFAR-10** with Convolutional Neural Network



Interactively classify toy 2-D data with a Neural Network

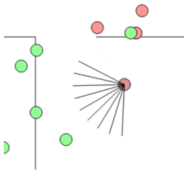


Interactively regress toy 1-D data



Train an MNIST digits Autoencoder

Reinforcement Learning with Deep Q Learning



Neural Network "paints" an image

Comparing SGD/Adagrad/Adadelta

ConvnetJS demo: Image "Painting"

This demo that treats the pixels of an image as a learning problem: it takes the (x,y) position on a grid and learns to predict the color at that point using regression to (r,g,b). It's a bit like compression, since the image information is encoded in the weights of the network, but almost certainly not of practical kind :)

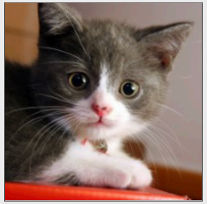
Note that the entire ConvNetJS definition is shown in textbox below and it gets eval()'d to create the network, so feel free to fiddle with the parameters and hit "reload". I found that, empirically and interestingly, deeper networks tend to work much better on this task given a fixed parameter budget.

Report questions/bugs/suggestions to @karpathy.

```
layer_defs = [];
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:2}); // 2 inputs: x, y
layer_defs.push({type:'fc', num_neurons:20, activation:'relu'});
layer_defs.push({type:'fc', num_neurons:20, activation:'relu'});
layer_defs.push({type:'fc', num_neurons:20, activation:'relu'});
layer_defs.push({type:'fc', num_neurons:20, activation:'relu'});
layer_defs.push({type:'fc', num_neurons:20, activation:'relu'});
layer_defs.push({type:'fc', num_neurons:20, activation:'relu'});
layer_defs.push({type:'fc', num_neurons:20, activation:'relu'});
layer_defs.push({type:'regression', num_neurons:3}); // 3 outputs: r,g,b

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01, momentum:0.9, batch_size:5, l2_decay:0.0});
```

reload network | Choose your own image: Choose File no file selected

Original Image



Neural Network output

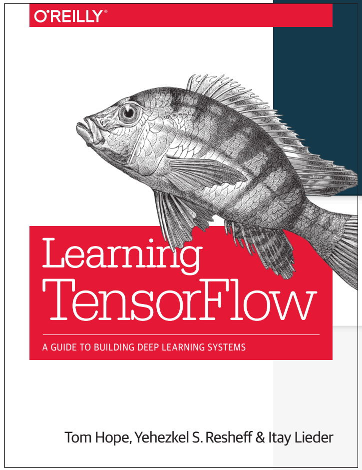
loss: 0.008217805256473332
iteration: 1677

Learning rate: 0.01

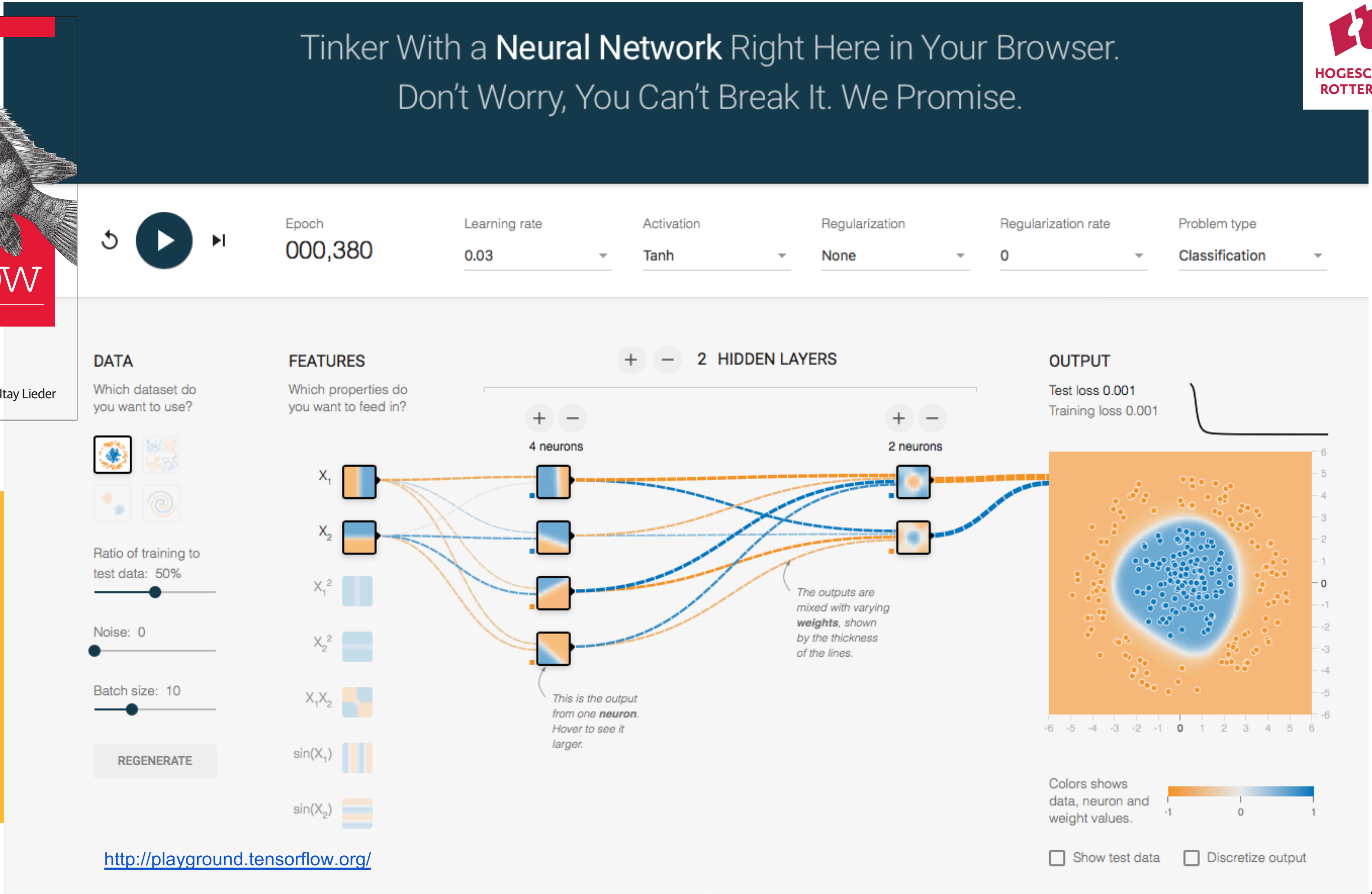
The learning rate should probably be decreased over time (slide left) to let the network better overfit the training data. It's nice to not have to worry about overfitting.

https://cs.stanford.edu/people/karpathy/convnetjs/demo/image_regression.html

HOGESCHOOL ROTTERDAM


O'REILLY
Learning TensorFlow
A GUIDE TO BUILDING DEEP LEARNING SYSTEMS
Tom Hope, Yehezkel S. Resheff & Itay Lieder

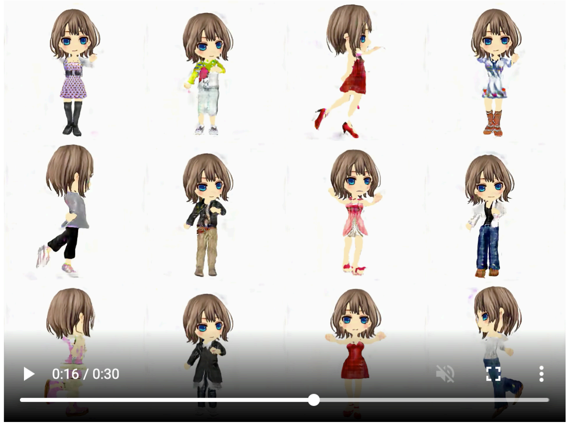

Bleeding Edge Press
TENSORFLOW FOR MACHINE INTELLIGENCE
Sam Abrahams, Danijar Hafner, Erik Erwitt, Ariel Scarpinelli


Tinker With a Neural Network Right Here in Your Browser.
Don't Worry, You Can't Break It. We Promise.
Epoch
000,380
Learning rate
0.03
Activation
Tanh
Regularization
None
Regularization rate
0
Problem type
Classification
DATA
Which dataset do you want to use?
Ratio of training to test data: 50%
Noise: 0
Batch size: 10
REGENERATE
FEATURES
Which properties do you want to feed in?
2 HIDDEN LAYERS
4 neurons
2 neurons
This is the output from one neuron. Hover to see it larger.
The outputs are mixed with varying weights, shown by the thickness of the lines.
OUTPUT
Test loss 0.001
Training loss 0.001
Colors shows data, neuron and weight values.
Show test data
Discretize output
http://playground.tensorflow.org/

Full-body High-resolution Anime Generation with Progressive Structure-conditional Generative Adversarial Networks

Koichi Hamada, Kentaro Tachibana, Tianqi Li, Hiroto Honda, and Yusuke Uchida
DeNA Co., Ltd., Tokyo, Japan

[Paper][ArXiv][Generated Anime 1][Generated Anime 2]

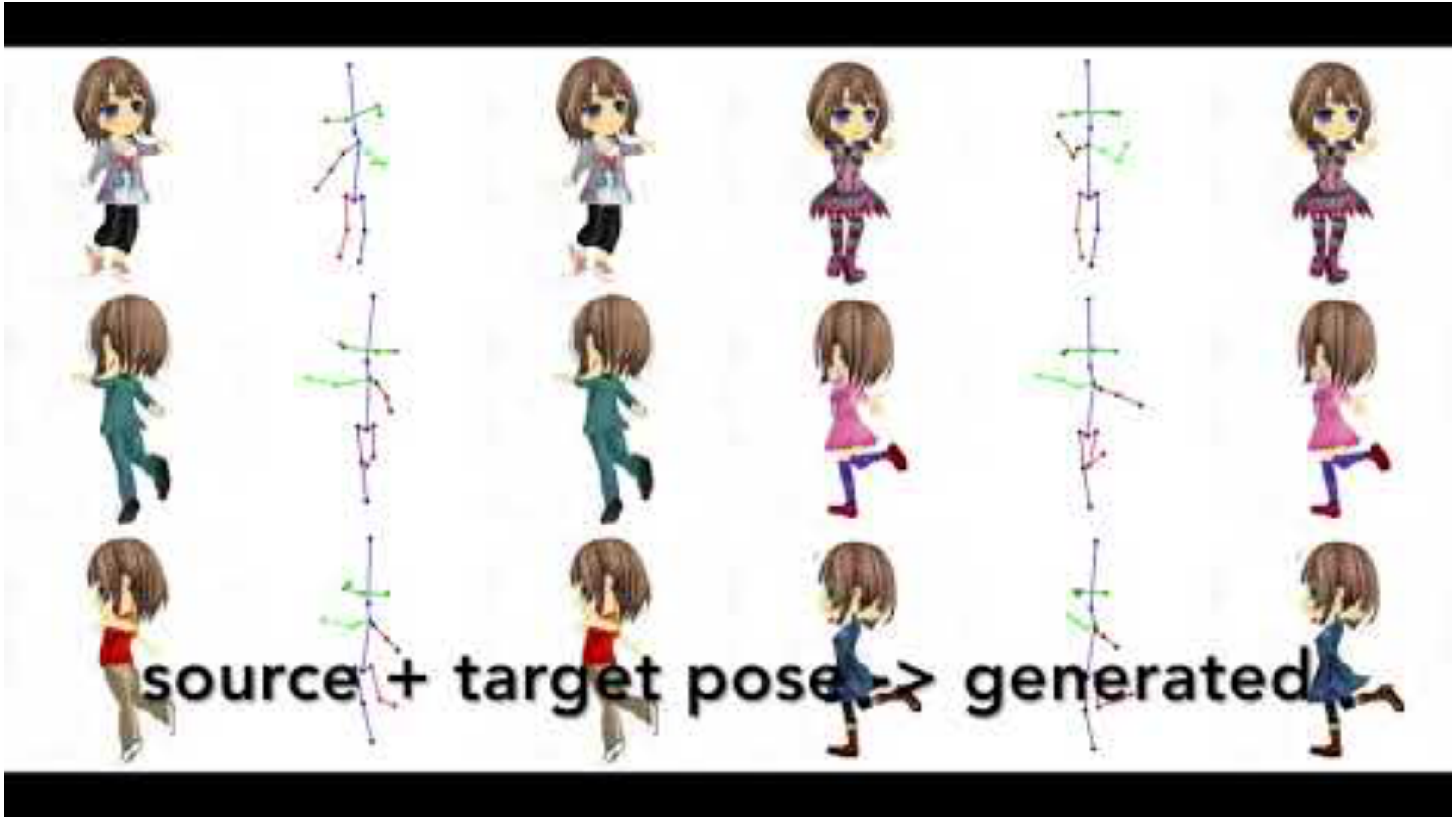


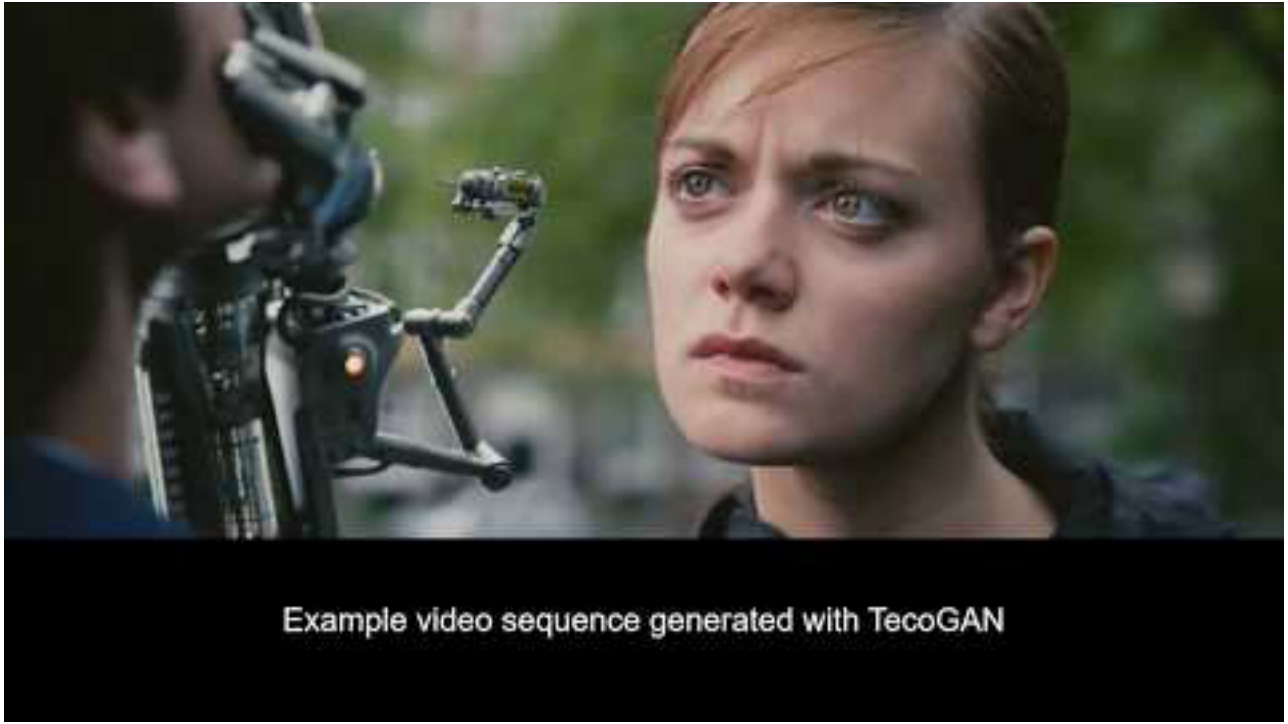
Artwork and paper have been accepted to

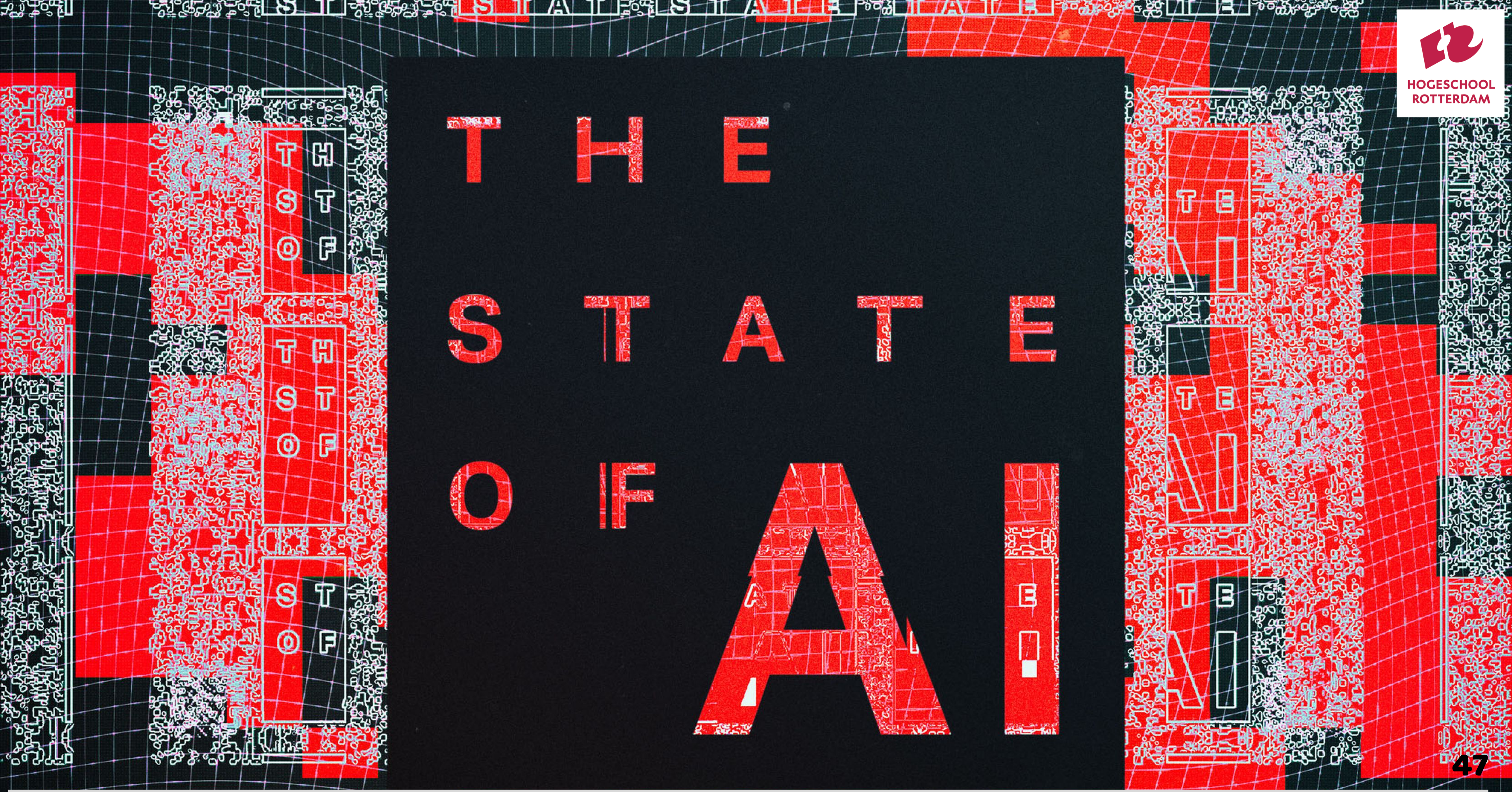
the ECCV Workshop on Computer Vision for Fashion, Art and Design, 2018.

- May 11, 2018: Project page launched.
- September 6, 2018: Submitted to arXiv.
- September 6, 2018: Generated animes updated to 1024x1024 res.
- September 14, 2018: Plan to present at the ECCV Workshop on Computer Vision for Fashion, Art and Design, 2018.

https://dena.com/intl/anime-generation/


source + target pose -> generated


Example video sequence generated with TecoGAN



HOGESCHOOL ROTTERDAM

https://www.theverge.com/2019/1/28/18197520/ai-artificial-intelligence-machine-learning-computational-science

Top 10 Technology Blogs/Magazines

#1 The Verge – Founded in 2011 The Verge (always in italics!) covers broad spectrum of topics with features, product reviews and podcasts across tech and science and culture, including the arts.

#2 Mashable – Founded in 2005 in New York Mashable provides plenty of news and resources for the "Connected Generation." It's got perhaps one of the most engaged audiences of all tech communities with 20m monthly unique visitors and over 6m followers on social media platforms.

#3 TechCrunch – Like Mashable founded in 2005. TechCrunch enjoys over 12m unique visitors a month, with its community numbering over 2m followers on social media networks. As well as often breaking stories on tech businesses – acquisition and fundraising – its CrunchBase database has become the place to go for information on techn companies, funding and major stories.

#4 The Next Web – Founded in 2008 The Next Web has more than 7.2m monthly unique visitors. The Next Web prides itself on giving an international angle to Internet and technology news and culture, and expands its readership through adding new channels and content partnerships. Like TechCrunch is runs events in Europe and North America.

#5 LifeHacker – Launched in 2005 Lifehacker bills itself as the place for "Tips, tricks, and downloads for getting things done." Covering topics as diverse as "From the Tips Box", anything Windows, Mac, Linux, Android or iOS related, careers, health and wine. Lifehacker also has two international editions – Lifehacker Japan & Lifehacker Australia.

#6 Wired – Wired.com (home to WIRED magazine first published in 1993) is part of the massive Conde Nast publishing group – which also owns Reddit (see below) as well as other great blogs such as Ars Technica. Wired's angle is to look at how ideas and innovation are changing the world. Chris Anderson (writer of The Long Tail, Free) set up his blog GeekDad which was later to become Wired.com. WIRED and Wired.com reach more than 14m readers a month.

#7 Reddit – If you've not come across Reddit where have you been? A massive community with a simple bulletin board, users post links or self-posts – other members of the community ranking the post or links up or down. The most popular links making the site's front page. Users build up kudos through the amount of ups/downs their posts and links get – "karma". Although part of Conde Nast, feels far from commercial.

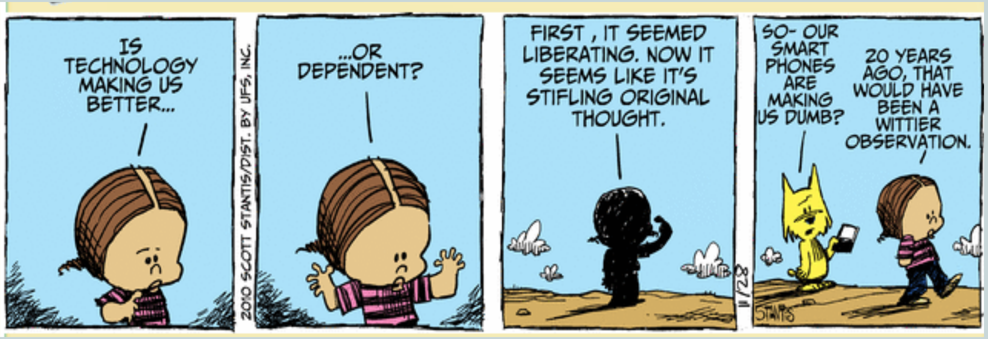
#8 Geek – One of the oldest blogs on our list bust still going strong. Founded in 1996 Geek.com features cover anything from buying guides and review for mobile, gaming, gadgets and computer hardware and software.

#9 Forbes – More of a business and lifestyle website than the rest on the list, but Forbes.com has a vast network of writers meaning that there's always new content – news and commentary. And given Forbes' size and reputation often with CEOs and senior execs of many of the major technology companies.

#10 Hongkiat – The creation of Hongkiat Lim, since 2007 this website has been giving its readers hints, tips and ideas on technology and design. Based in Malaysia this website has grown considerably and is often mentioned itself by many of the bigger sites – such as Lifehacker and TheNextWeb above.

https://www.statuscake.com/statuscakes-top-10-technology-blogs-how-do-you-start-your-day/

HOGESCHOOL ROTTERDAM



BY NC SA

This lesson was developed by:

Robert Frans van der Willigen
CMD, Hogeschool Rotterdam
FEB 2018

Creative Commons License Types

	Can someone use it commercially?	Can someone create new versions of it?
Attribution	👍	👍
Share Alike	👍	Yup, AND they must license the new work under a Share Alike license.
No Derivatives	👍	👎
Non-Commercial	👎	Yup, AND the new work must be non-commercial, but it can be under any non-commercial license.
Non-Commercial Share Alike	👎	Yup, AND they must license the new work under a Non-Commercial Share Alike license.
Non-Commercial No Derivatives	👎	👎

SOURCE
http://www.masternewmedia.org/how-to-publish-a-book-under-a-creative-commons-license/

http://empoweringthenatives.edublogs.org/2012/03/15/creative-commons-licenses/

http://creativecommons.org/licenses/

HOGESCHOOL ROTTERDAM

HOGESCHOOL ROTTERDAM