

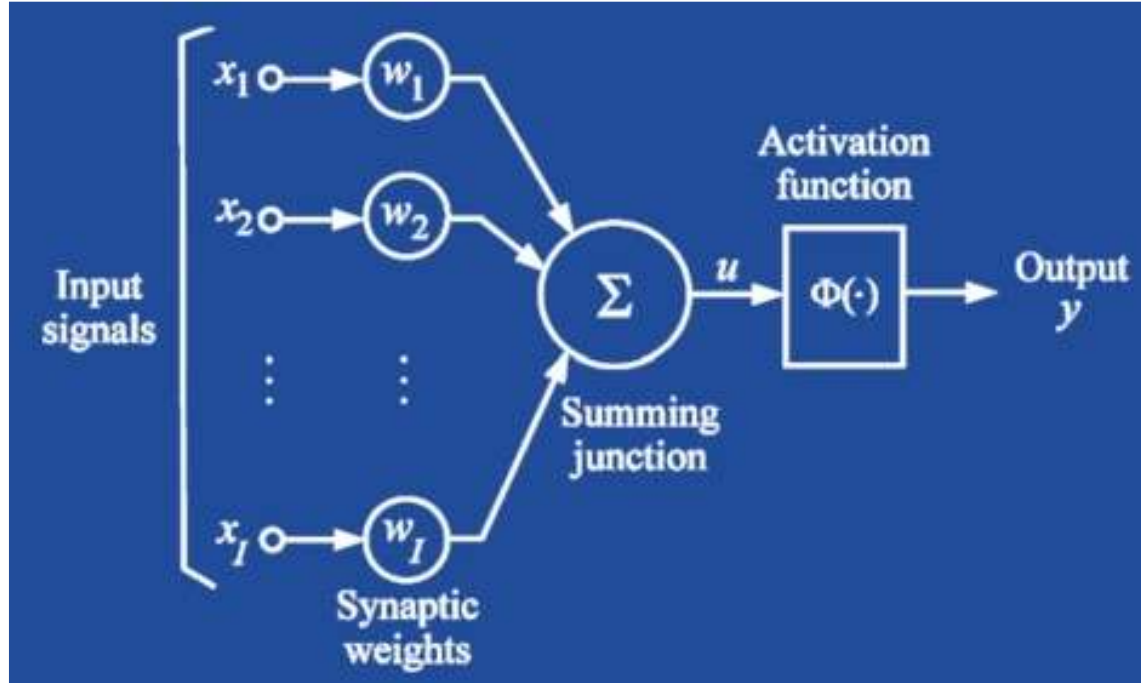


Deep Learning for Developers

Julien Simon
Principal Technical Evangelist, AI & Machine Learning
[@julsimon](https://twitter.com/julsimon)

July 2018

The neuron



$$\sum_{i=1}^l x_i * w_i = u$$

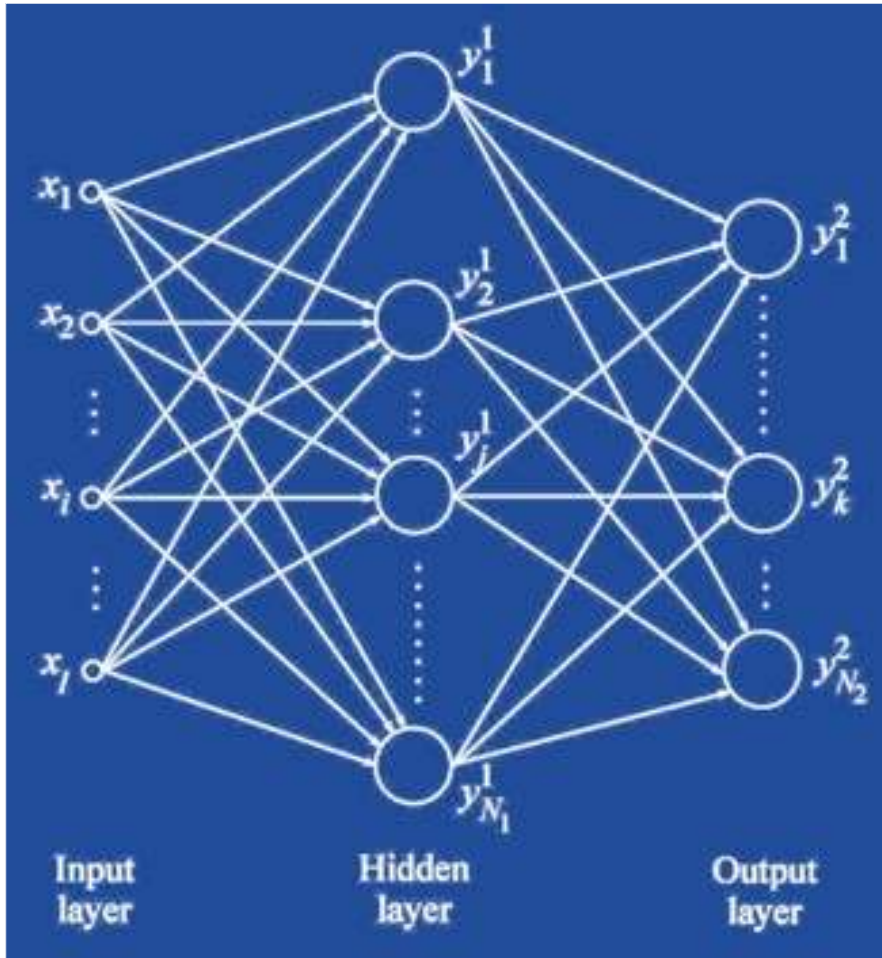
”Multiply and Accumulate”

Activation functions

Name	Plot	Equation
Identity		$f(x) = x$
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Logistic (a.k.a. Soft step)		$f(x) = \frac{1}{1 + e^{-x}}$
TanH		$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$
ArcTan		$f(x) = \tan^{-1}(x)$
Softsign [7][8]		$f(x) = \frac{x}{1 + x }$
Rectified linear unit (ReLU) [9]		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$

Source: Wikipedia

Neural networks



X
=

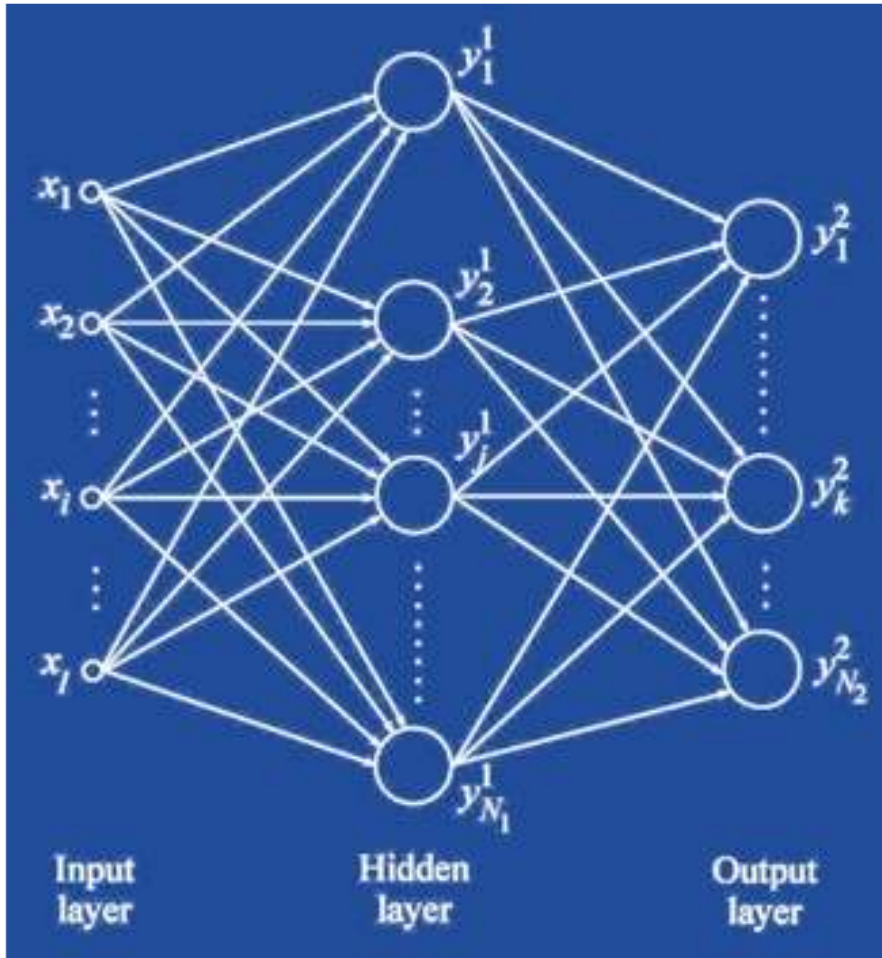
$$\begin{matrix}
 \text{I features} \\
 \left[\begin{array}{c} X_{11}, X_{12}, \dots, X_{1I} \\ X_{21}, X_{22}, \dots, X_{2I} \\ \dots \dots \dots \\ X_{m1}, X_{m2}, \dots, X_{mI} \end{array} \right]
 \end{matrix}
 \quad \left. \vphantom{\begin{matrix} X_{11}, X_{12}, \dots, X_{1I} \\ X_{21}, X_{22}, \dots, X_{2I} \\ \dots \dots \dots \\ X_{m1}, X_{m2}, \dots, X_{mI} \end{matrix}} \right\} m \text{ samples}$$

$y =$

$$\begin{matrix}
 \left[\begin{array}{c} 2 \\ 0 \\ \dots \\ 4 \end{array} \right]
 \end{matrix}
 \quad
 \begin{matrix}
 \left[\begin{array}{c} 0,0,1,0,0,\dots,0 \\ 1,0,0,0,0,\dots,0 \\ \dots \\ 0,0,0,0,1,\dots,0 \end{array} \right]
 \end{matrix}
 \quad \left. \vphantom{\begin{matrix} 0,0,1,0,0,\dots,0 \\ 1,0,0,0,0,\dots,0 \\ \dots \\ 0,0,0,0,1,\dots,0 \end{matrix}} \right\} m \text{ labels, } N_2 \text{ categories}$$

One-hot encoding

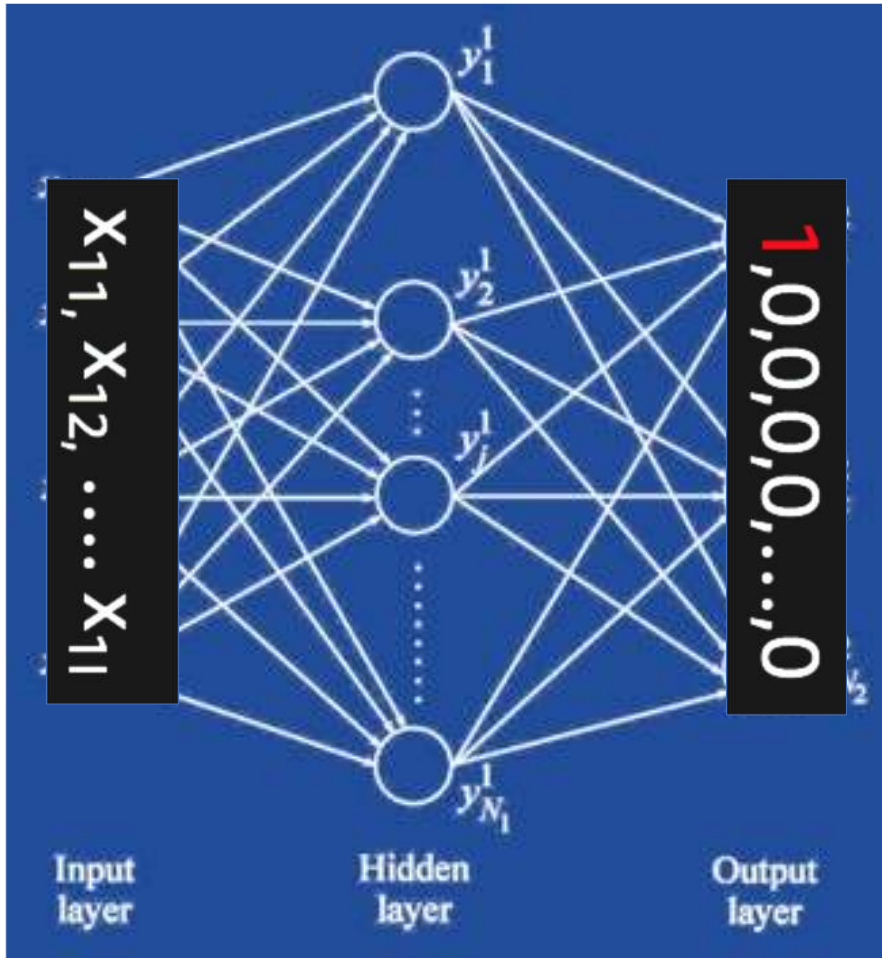
Neural networks



$$\begin{aligned}
 \mathbf{X} &= \begin{bmatrix} x_{11}, x_{12}, \dots, x_{1I} \\ x_{21}, x_{22}, \dots, x_{2I} \\ \vdots \\ x_{m1}, x_{m2}, \dots, x_{mI} \end{bmatrix} \quad \begin{matrix} I \text{ features} \\ m \text{ samples} \end{matrix} \\
 \mathbf{y} &= \begin{bmatrix} 2 \\ 0 \\ \vdots \\ 4 \end{bmatrix} \quad \begin{bmatrix} 0, 0, 1, \dots, 0 \\ 1, 0, 0, \dots, 0 \\ \vdots \\ 0, 0, 0, \dots, 0 \end{bmatrix} \quad \begin{matrix} m \text{ labels,} \\ N_2 \text{ categories} \end{matrix} \\
 &\quad \text{One-hot encoding}
 \end{aligned}$$

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}}$$

Neural networks



Initially, the network will **not** predict correctly

$$f(X_1) = Y'_1$$

A **loss function** measures the difference between the **real label** Y_1 and the **predicted label** Y'_1

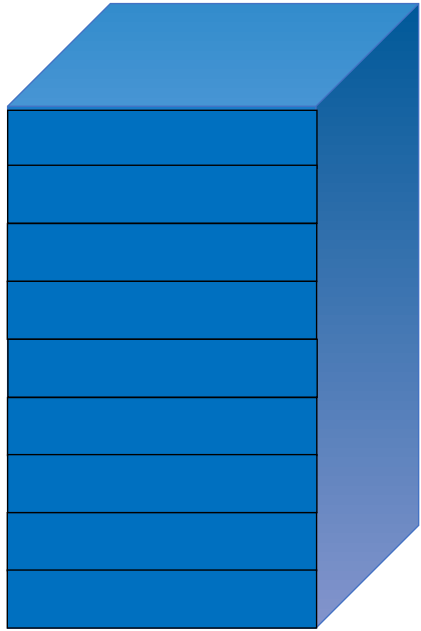
$$\text{error} = \text{loss}(Y_1, Y'_1)$$

For a **batch** of samples:

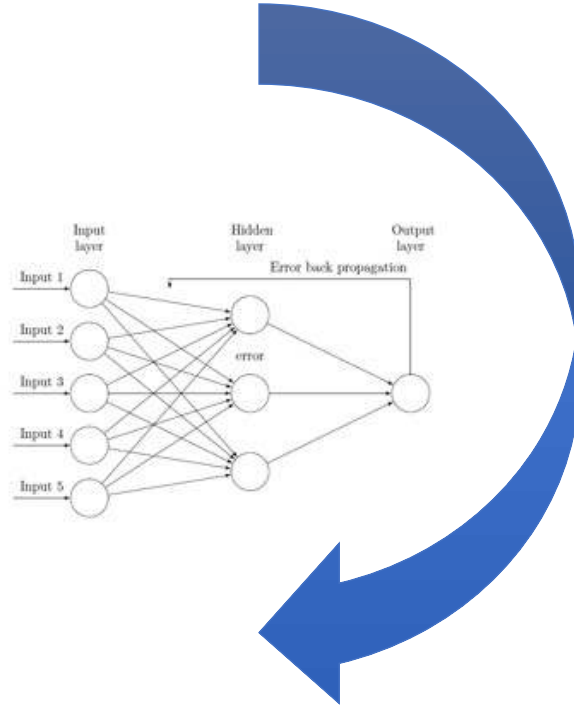
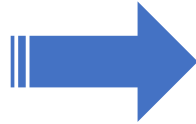
$$\sum_{i=1}^{\text{batch size}} \text{loss}(Y_i, Y'_i) = \text{batch error}$$

The purpose of the training process is to **minimize loss** by gradually **adjusting weights**

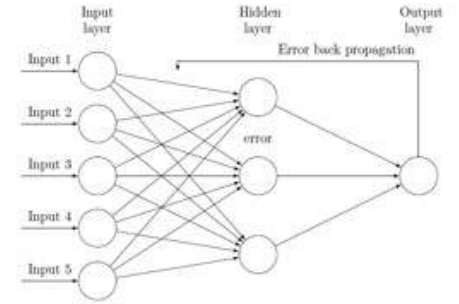
Training



Training data set



Backpropagation



Trained
neural network

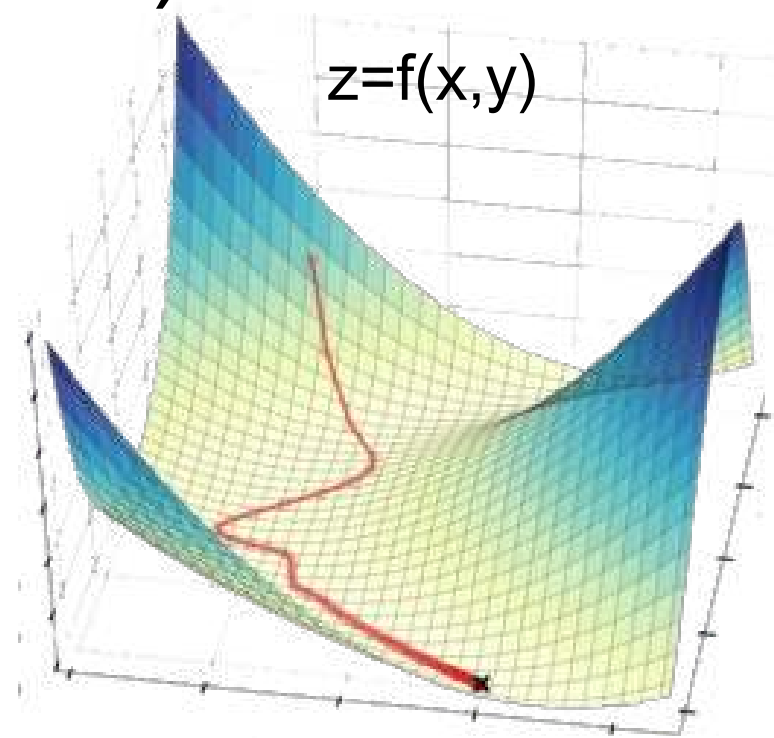
Batch size
Learning rate
Number of epochs

} Hyper parameters

Stochastic Gradient Descent (SGD)

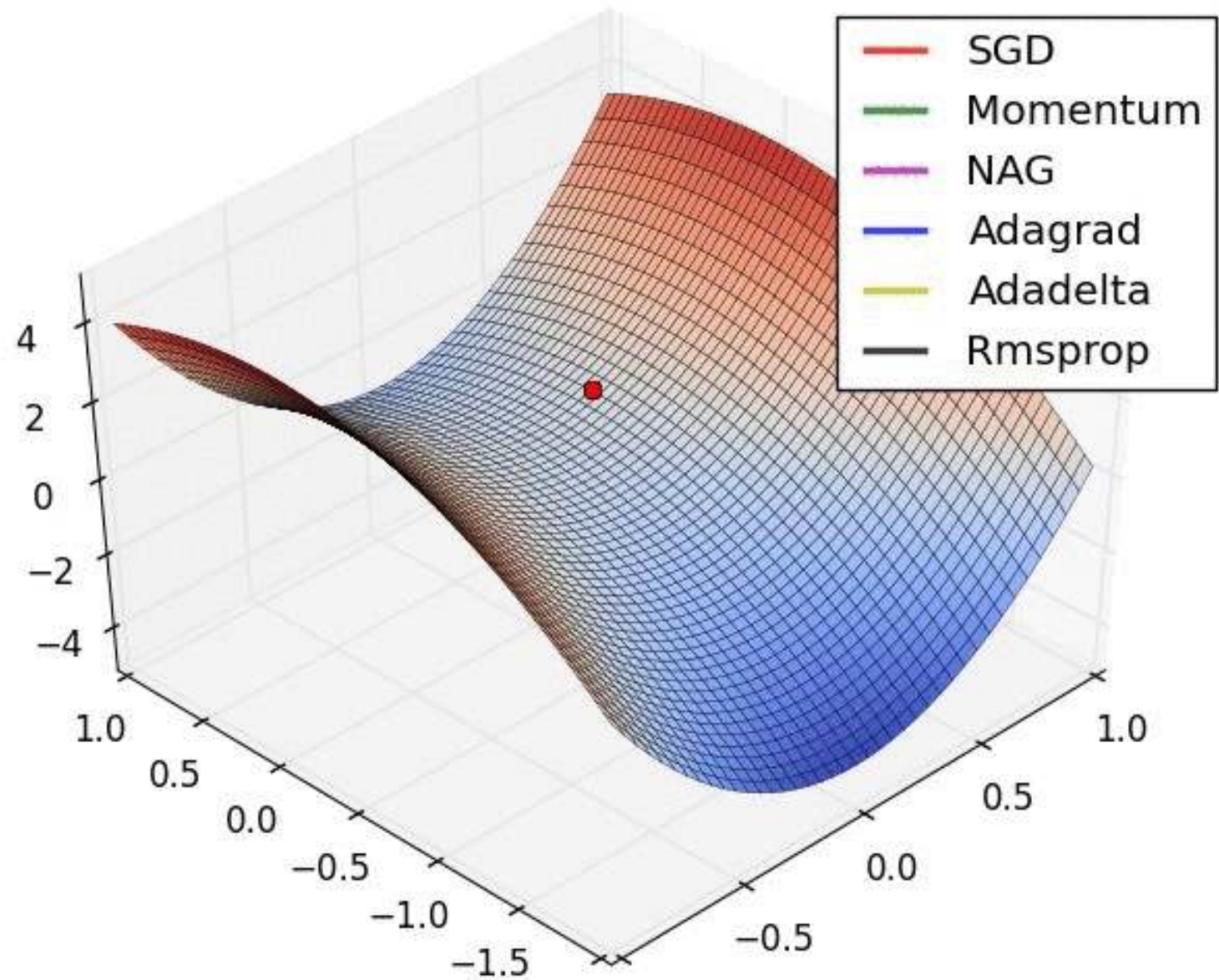
*Imagine you stand on top of a mountain with skis strapped to your feet. You want to get down to the valley as quickly as possible, but there is fog and you can only see your immediate surroundings. How can you get down the mountain as quickly as possible? You look around and *identify the steepest path down*, go down that path for a bit, again *look around* and *find the new steepest path*, go down that path, and *repeat*—this is exactly what gradient descent does.*

Tim Dettmers
University of Lugano
2015



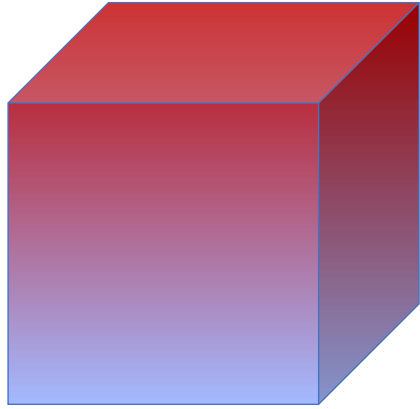
The « step size » is called the **learning rate**

Optimizers

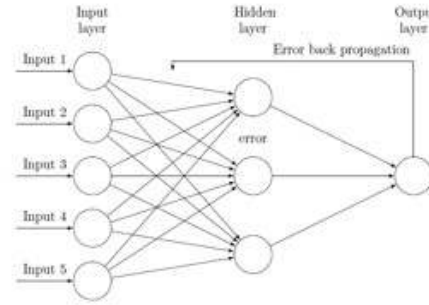


<https://medium.cim/@julsimon/tumbling-down-the-sgd-rabbit-hole-part-1-740fa402f0d7>

Validation



Validation data set



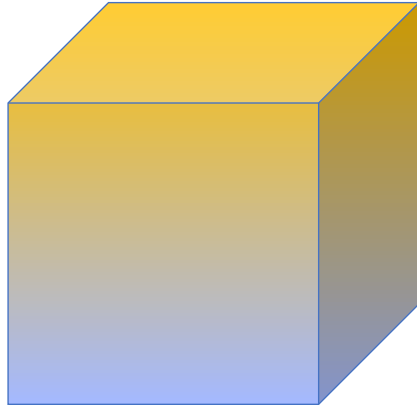
Trained
neural network



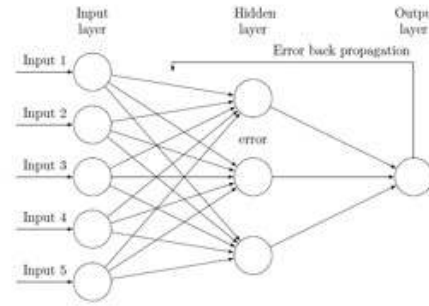
Prediction at
the end of
each epoch

Validation
accuracy

Test



Test data set



Fully trained
neural network



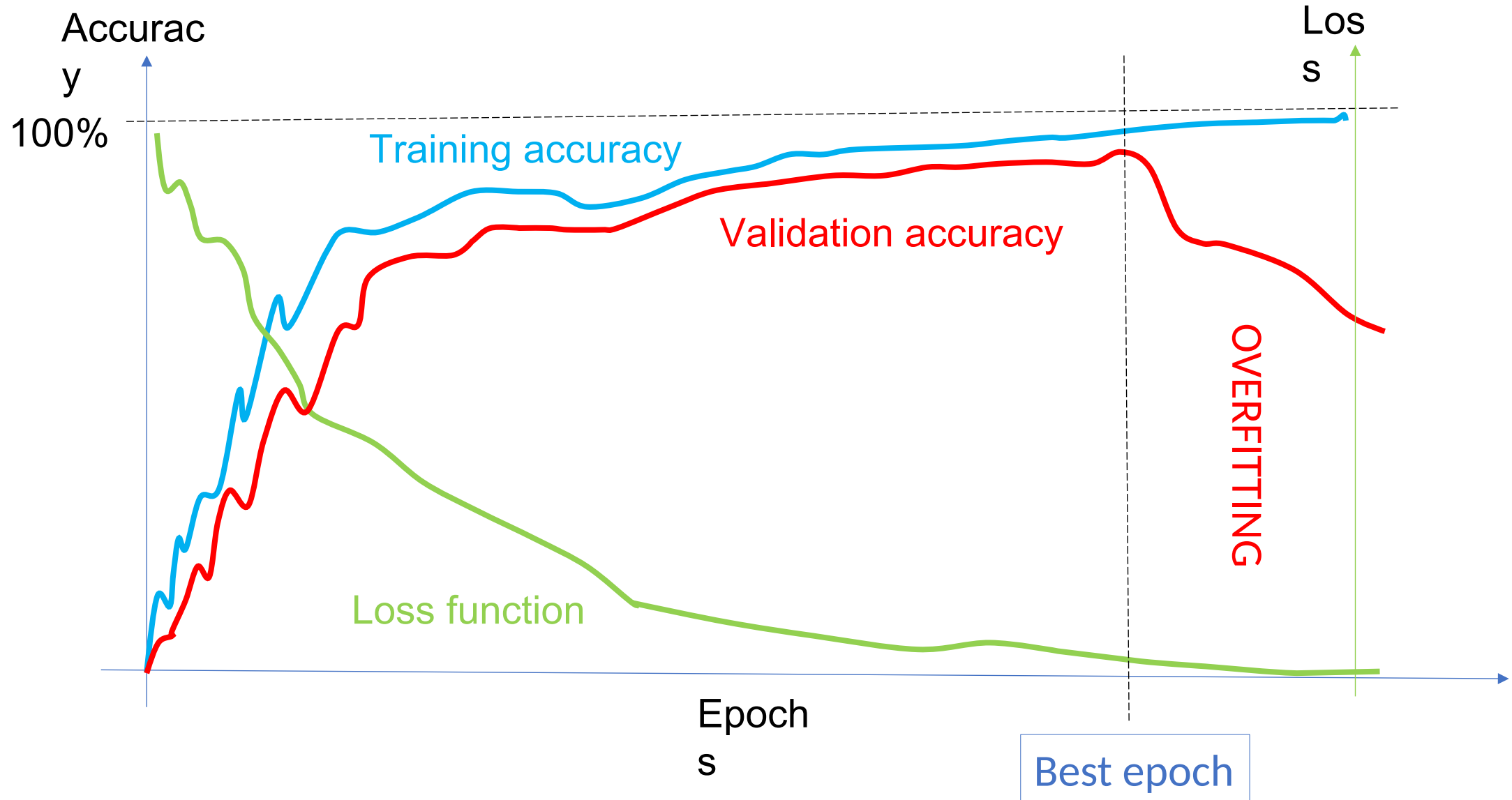
Test accuracy

Prediction at the
end of
experimentation

This data set must have the same distribution as real-life samples,
or else test accuracy won't reflect real-life accuracy.

Early stopping

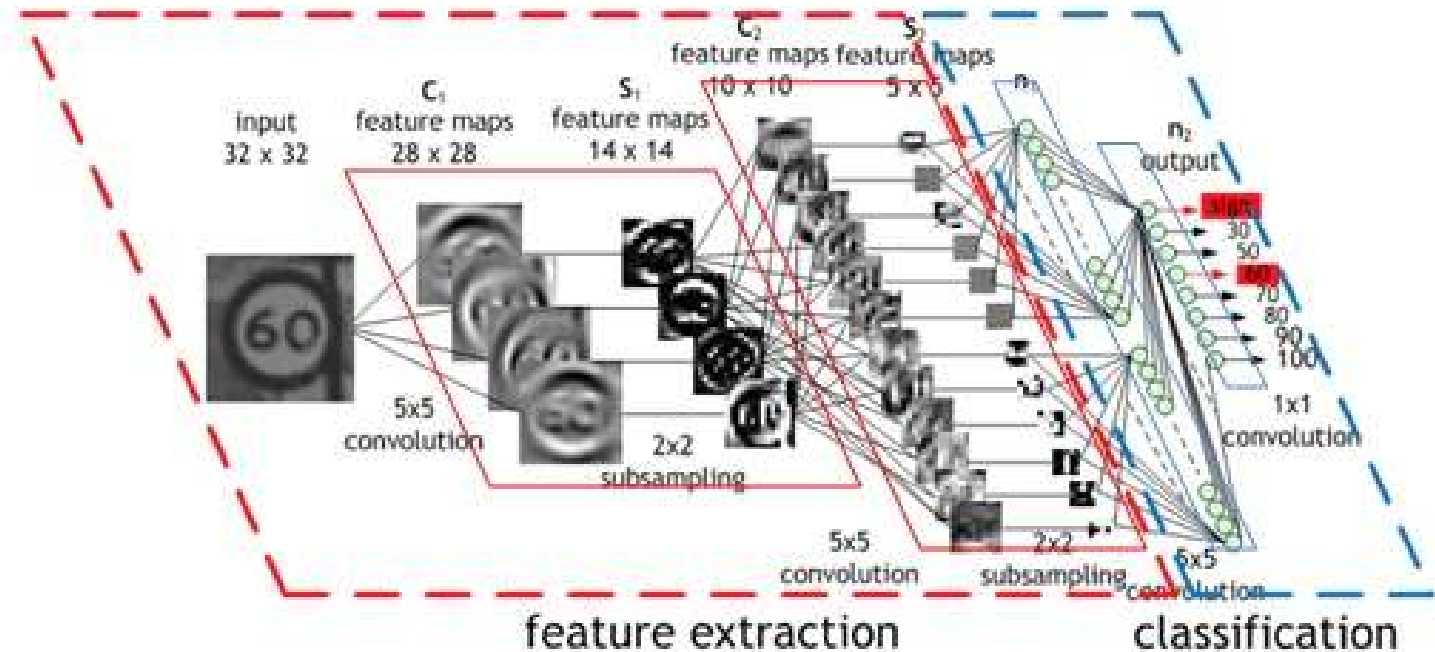
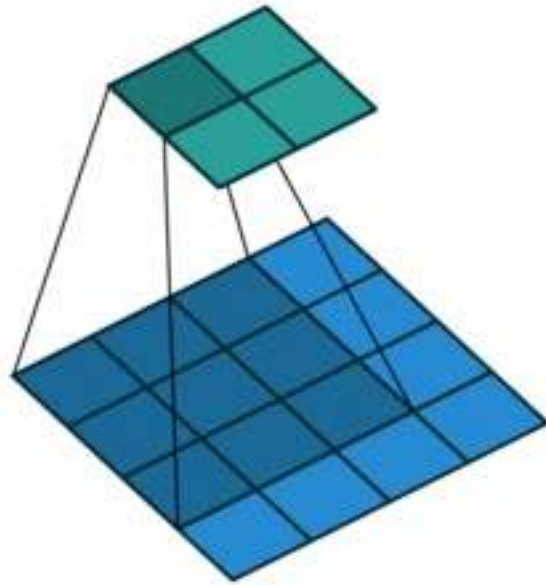
« Deep Learning ultimately is about finding a minimum that generalizes well, with bonus points for finding one fast and reliably », Sebastian Ruder



Demo: fully connected network

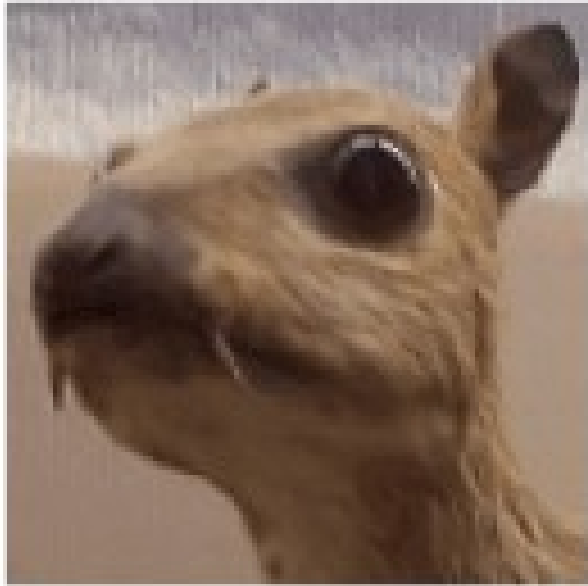
Convolutional Neural Networks (CNN)

Le Cun, 1998: handwritten digit recognition, 32x32 pixels



Extracting features with convolution

Input image



Convolution
Kernel

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

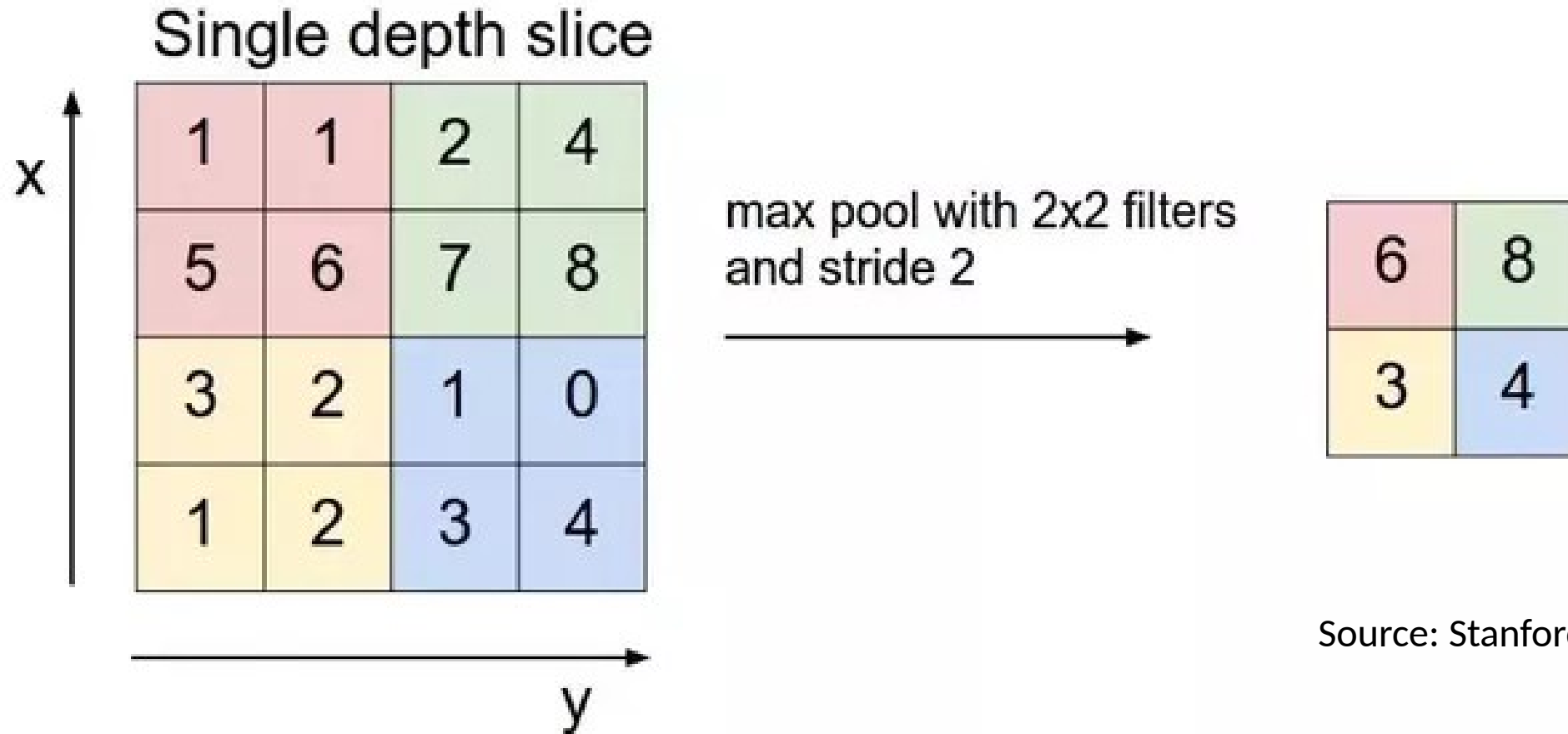
Feature map



Source: <http://timdettmers.com>

Convolution **extracts features** automatically.
Kernel parameters are **learned** during the training process.

Downsampling images with pooling



Source: Stanford University

Pooling shrinks images while preserving **significant** information.

Demo: convolutional network

Gluon CV: state of the art pre-trained models

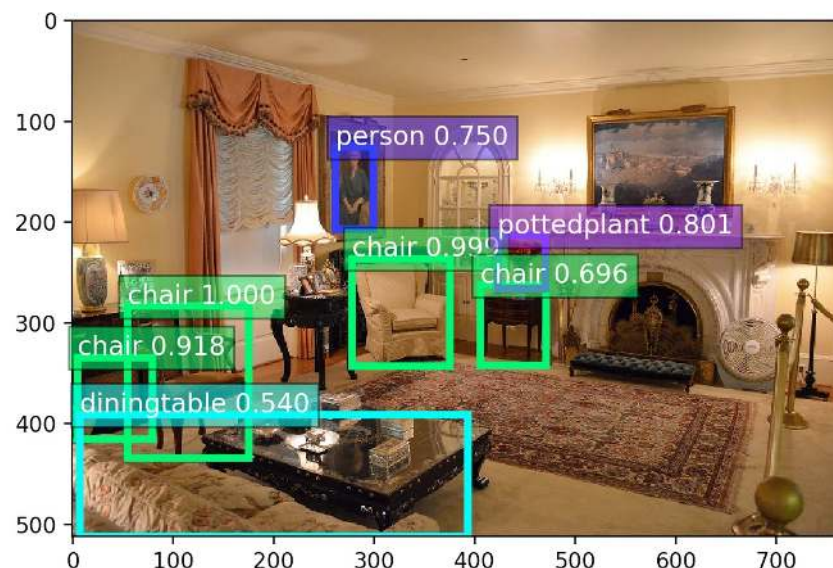


Classification



[electric_guitar],
with probability 0.671

Detection



Segmentation

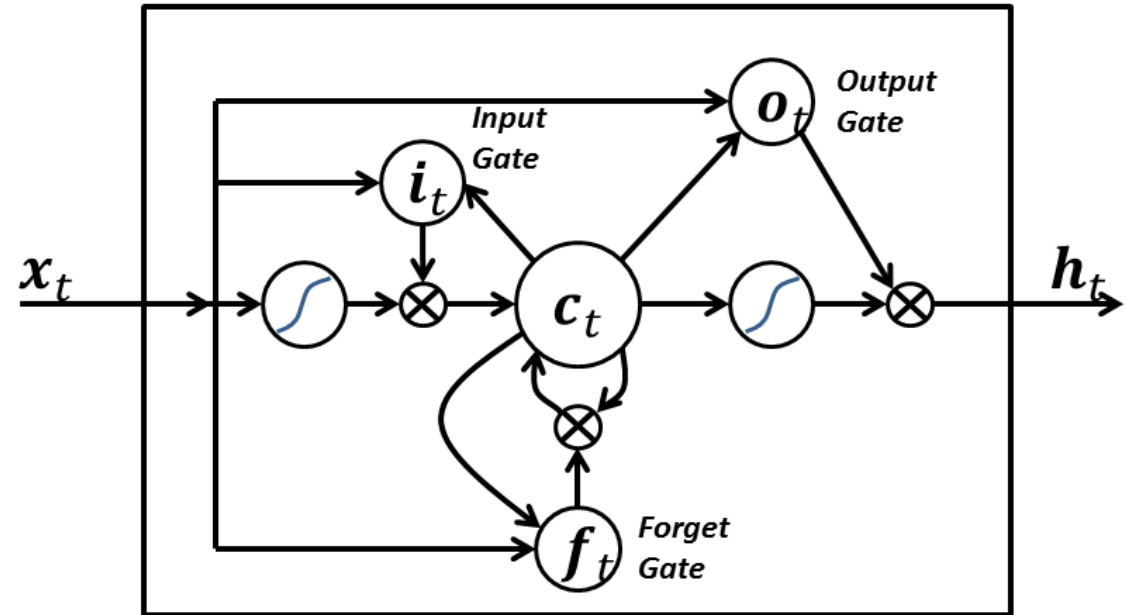


<https://github.com/dmlc/gluon-cv>

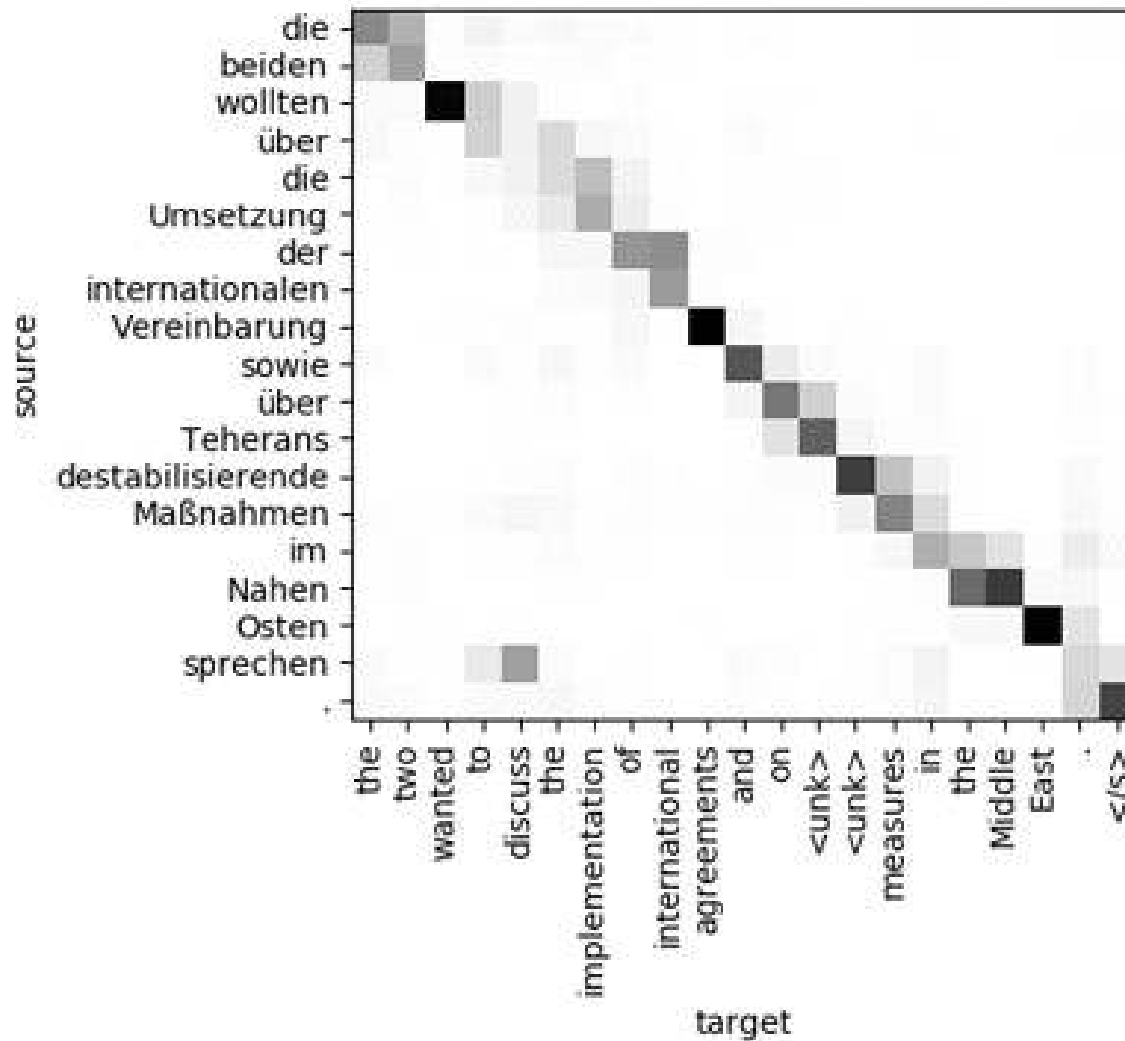
Demo: Gluon CV

Long Short Term Memory Networks (LSTM)

- A LSTM neuron computes the output based on the input and a **previous state**
- LSTM networks have **memory**
- They're great at predicting **sequences**, e.g. machine translation



Machine Translation



GAN: Welcome to the (un)real world, Neo

PyTorch



TF

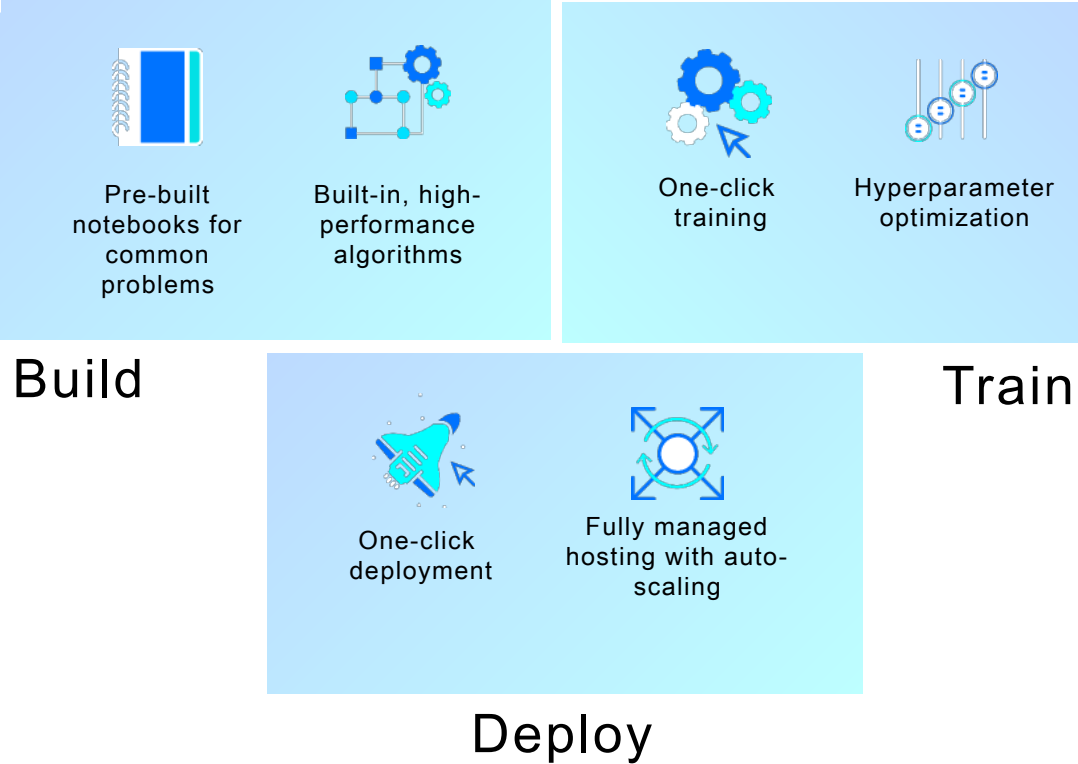


Generating new "celebrity" faces https://github.com/tkarras/progressive_growing_of_gans

From semantic map to 2048x1024 picture
<https://tcwang0509.github.io/pix2pixHD/>

Scalable training on AWS

Amazon SageMaker



AWS Deep Learning AMI



Amazon
EC2

c5



p3



Getting started

<https://ml.aws> | <https://aws.amazon.com/blogs/machine-learning/>

<https://mxnet.incubator.apache.org> | <https://github.com/apache/incubator-mxnet>

<https://gluon.mxnet.io> | <https://github.com/gluon-api> | <https://github.com/dmlc/gluon-cv>

<https://aws.amazon.com/sagemaker>

<https://github.com/awslabs/amazon-sagemaker-examples>

<https://github.com/aws/sagemaker-python-sdk> | <https://github.com/aws/sagemaker-spark>

<https://medium.com/@julsimon>

<https://youtube.com/juliensimonfr>

<https://gitlab.com/juliensimon/dlnotebooks>



Thank you!

Julien Simon
Principal Technical Evangelist, AI & Machine Learning
[@julsimon](#)