End2End Data Science

From Keras to Production!

Contents

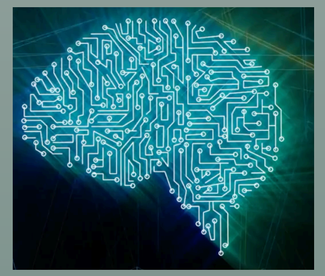
- Learn what neural nets are and what's the big deal with deep learning
- Work with Keras to apply existing models on your own data
- Build your own fruit classification model!

- Get a glimpse of production-readiness
- Version your data and models with DVC
- Deploy your model with Tensorflow Serving
- Learn about Luigi/Airflow and write your own pipeline

What is Deep Learning?

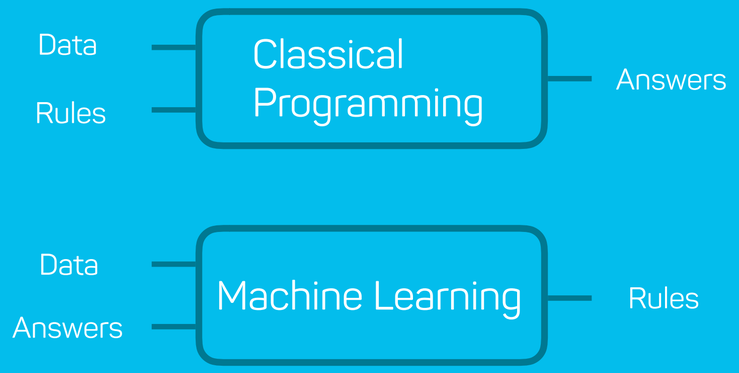
Artificial Intelligence

Any technique that enables computers to mimic human behavior



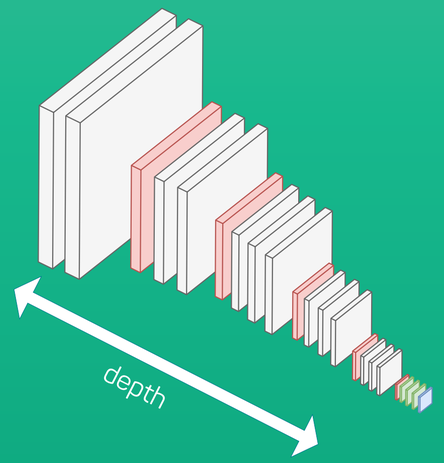
Machine Learning

Ability to learn rules from data



Deep Learning

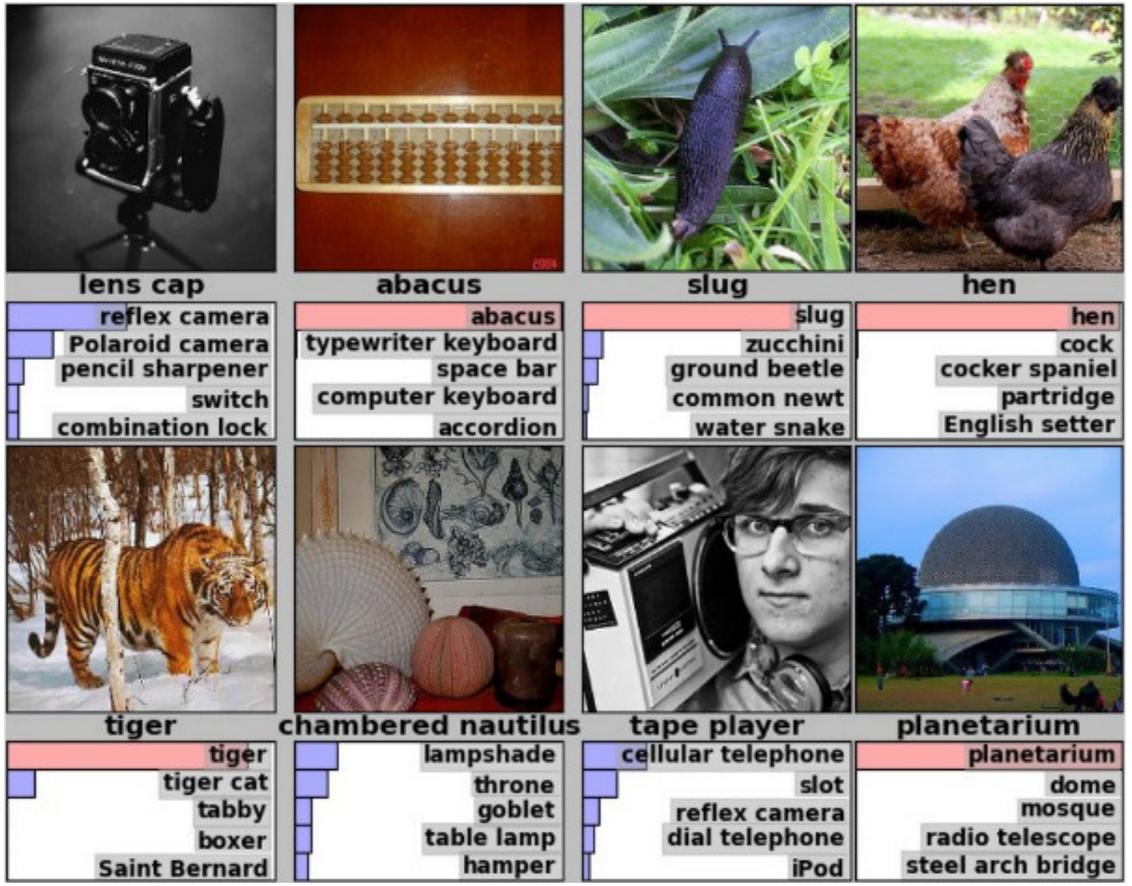
Apply machine learning with „deep“ neural networks



Deep Learning Achievements - ImageNet

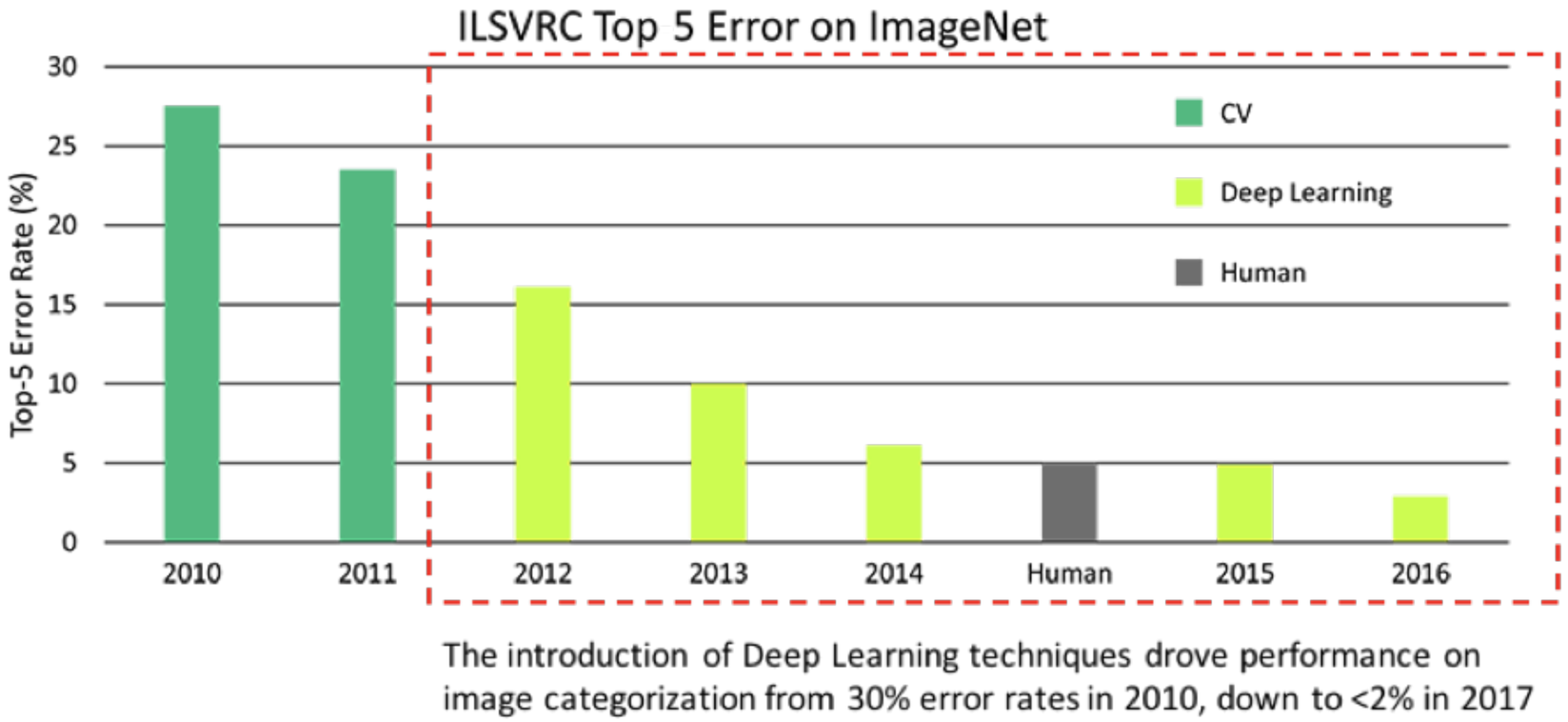
IMAGENET

- 1000 categories
- train set: 1.2 M
- test set: 100 k



https://medium.com/obvious-ventures/our-investment-in-darwinai-d5ea1a7af32e

Deep Learning Achievements - ImageNet

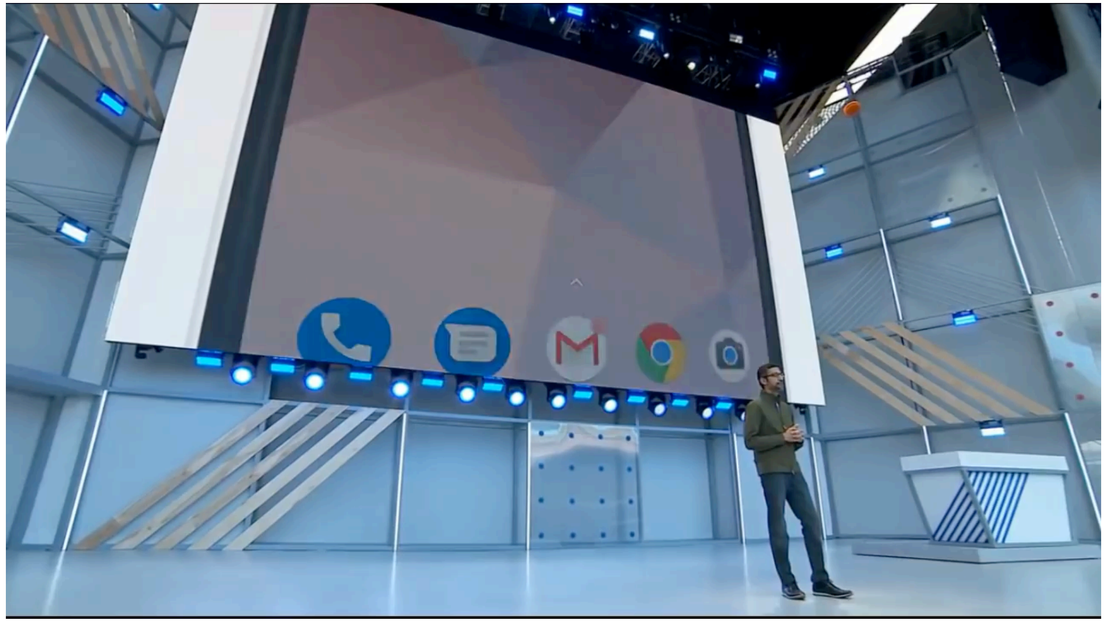


The introduction of Deep Learning techniques drove performance on image categorization from 30% error rates in 2010, down to <2% in 2017

https://medium.com/obvious-ventures/our-investment-in-darwinai-d5ea1a7af32e

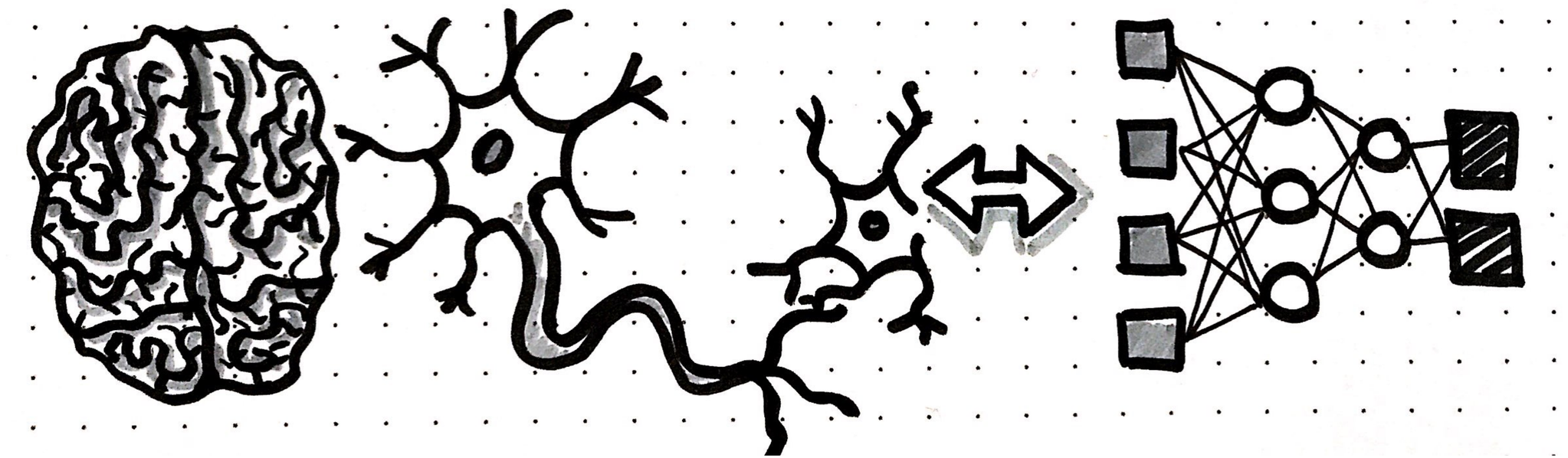
Deep Learning Achievements - Speech Recognition

2006
Microsoft

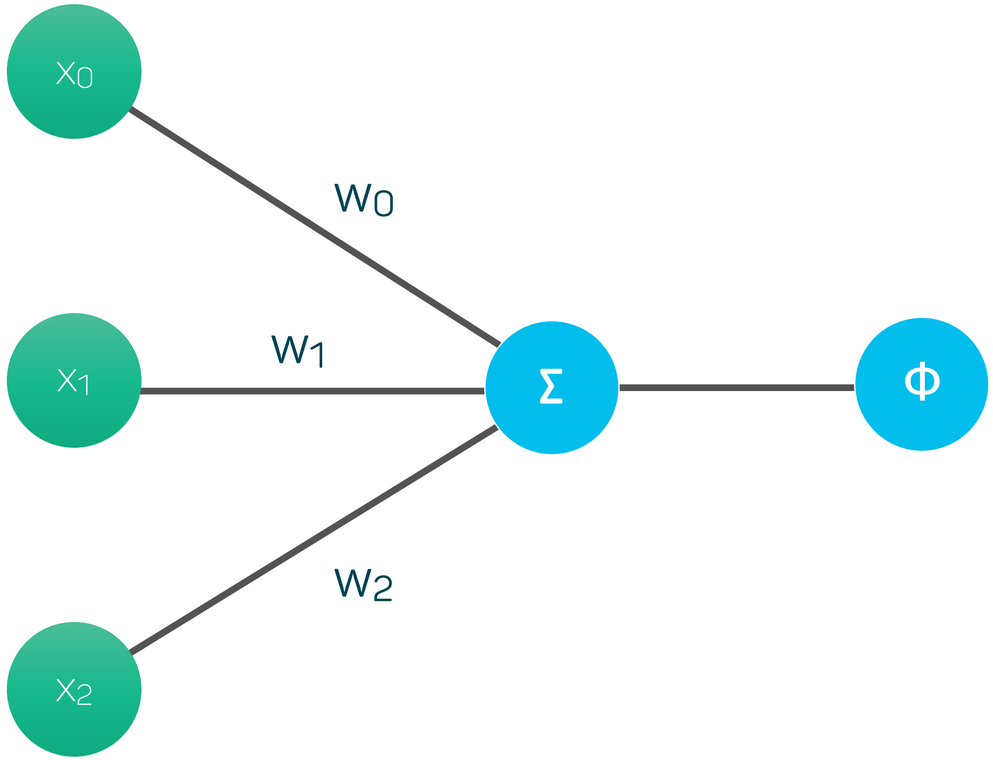


2018
Google

Neural Networks: Inspired by Nature



Fundamentals of Deep Learning - Perceptron



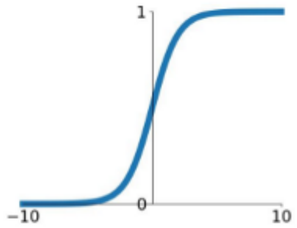
$$f(x) = \phi(\sum_i w_i x_i + b)$$

Fundamentals of Deep Learning

Activation Functions

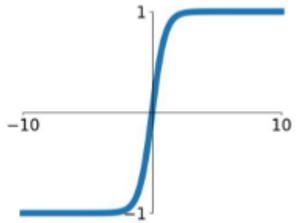
Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



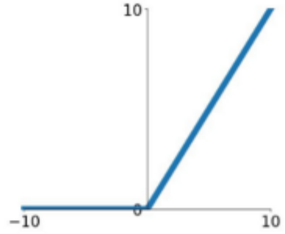
tanh

$$\tanh(x)$$



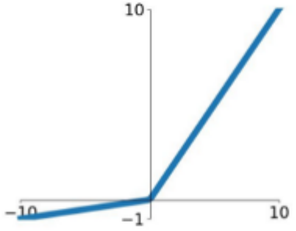
ReLU

$$\max(0, x)$$



Leaky ReLU

$$\max(0.1x, x)$$

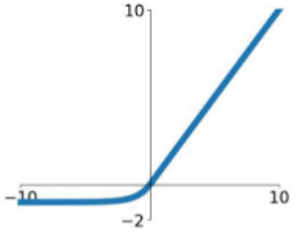


Maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

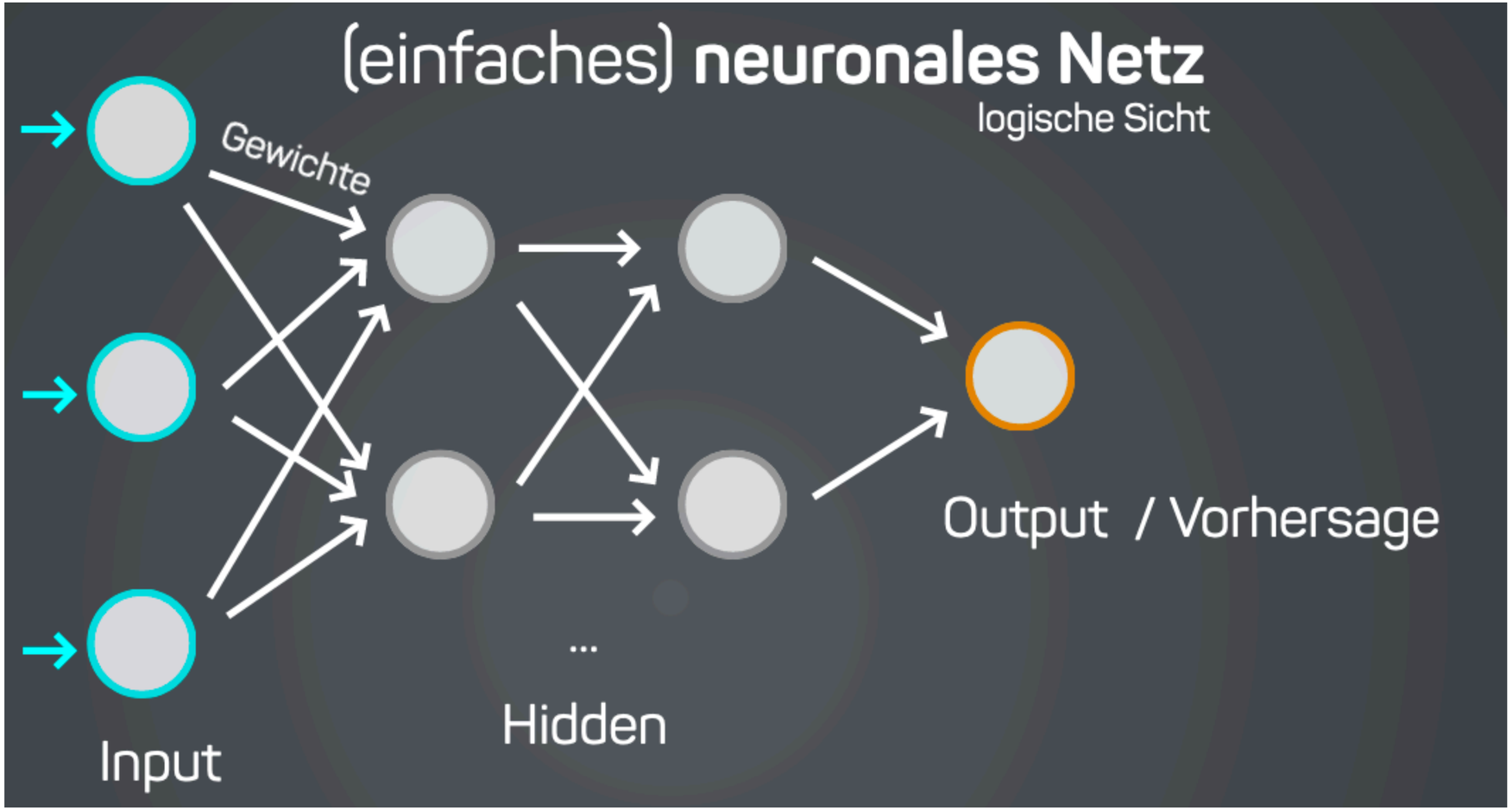
ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$

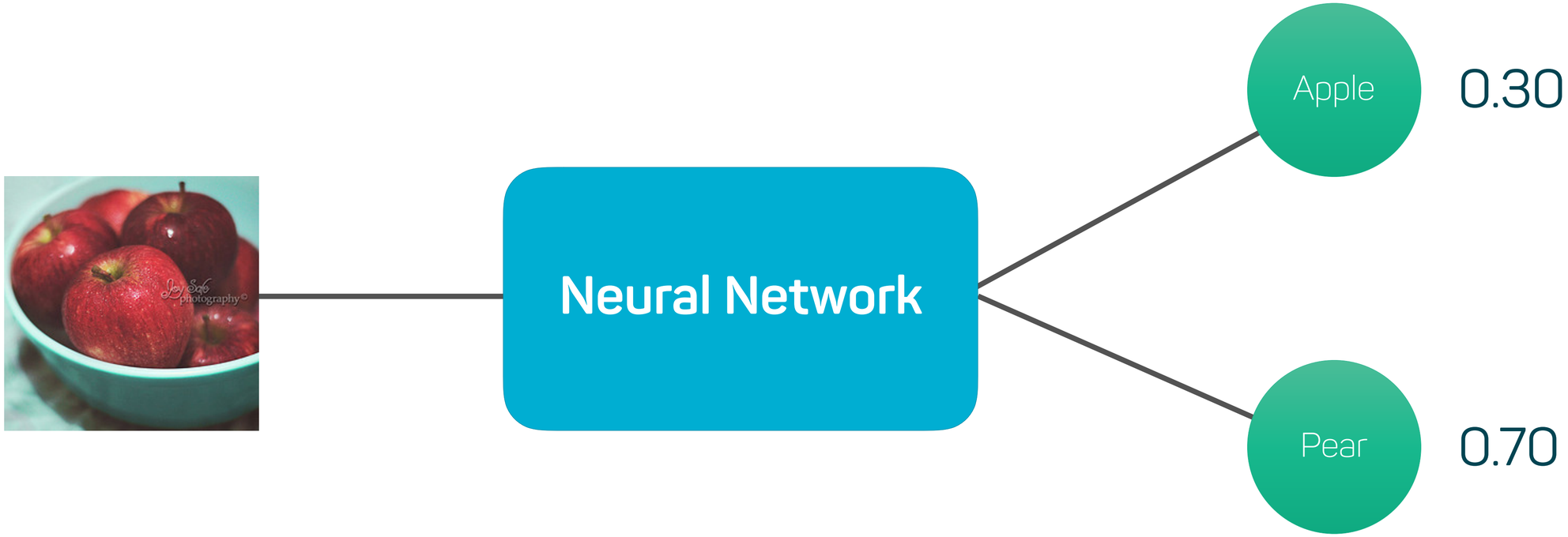


https://cdn-images-1.medium.com/max/1200/1*ZafDv3VUm60Eh10OeJu1vw.png

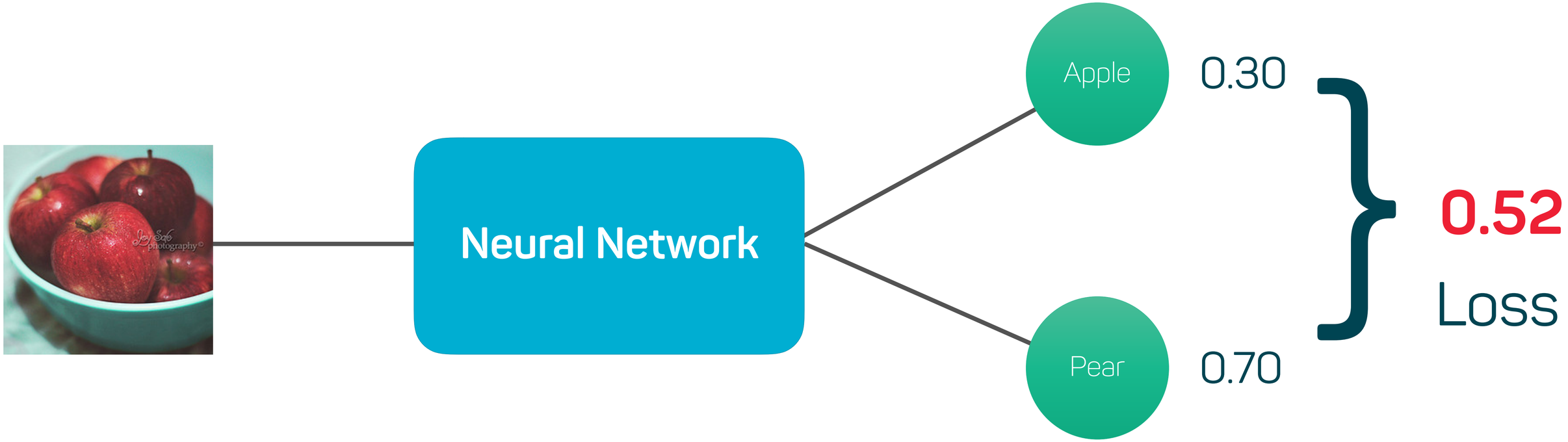
Fundamentals of Deep Learning



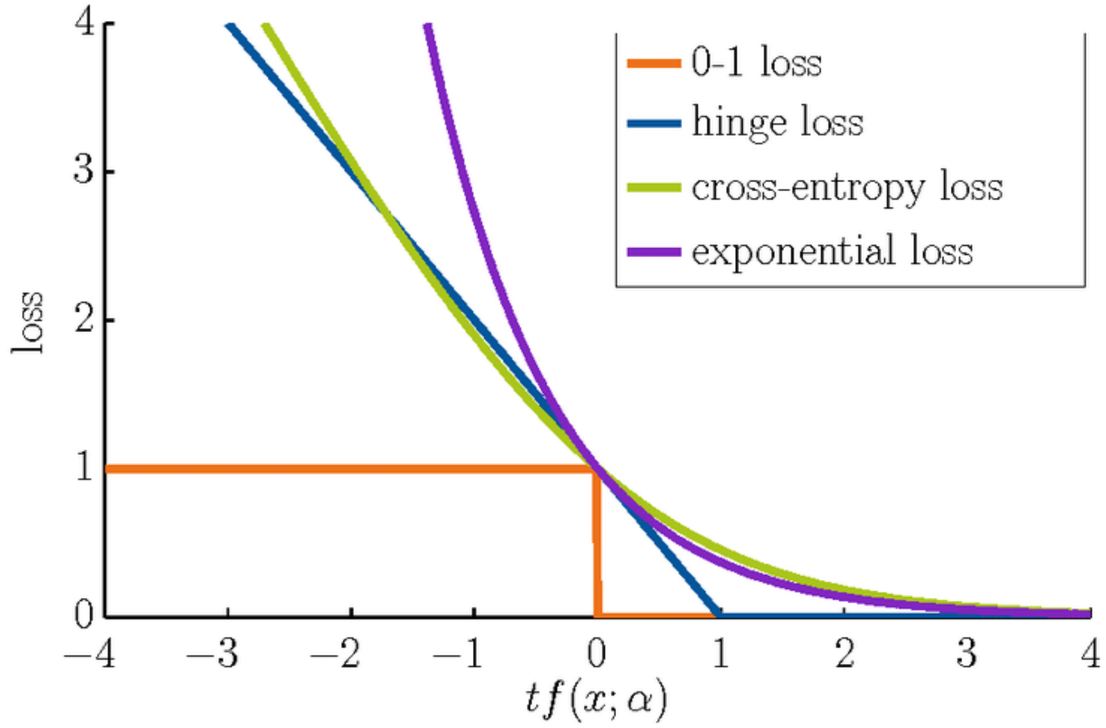
Fundamentals of Deep Learning



Fundamentals of Deep Learning - Loss Function



Fundamentals of Deep Learning - Loss Functions



Any loss function returns a scalar value!

Fundamentals of Deep Learning

Neural Network

Apple

0.30

Pear

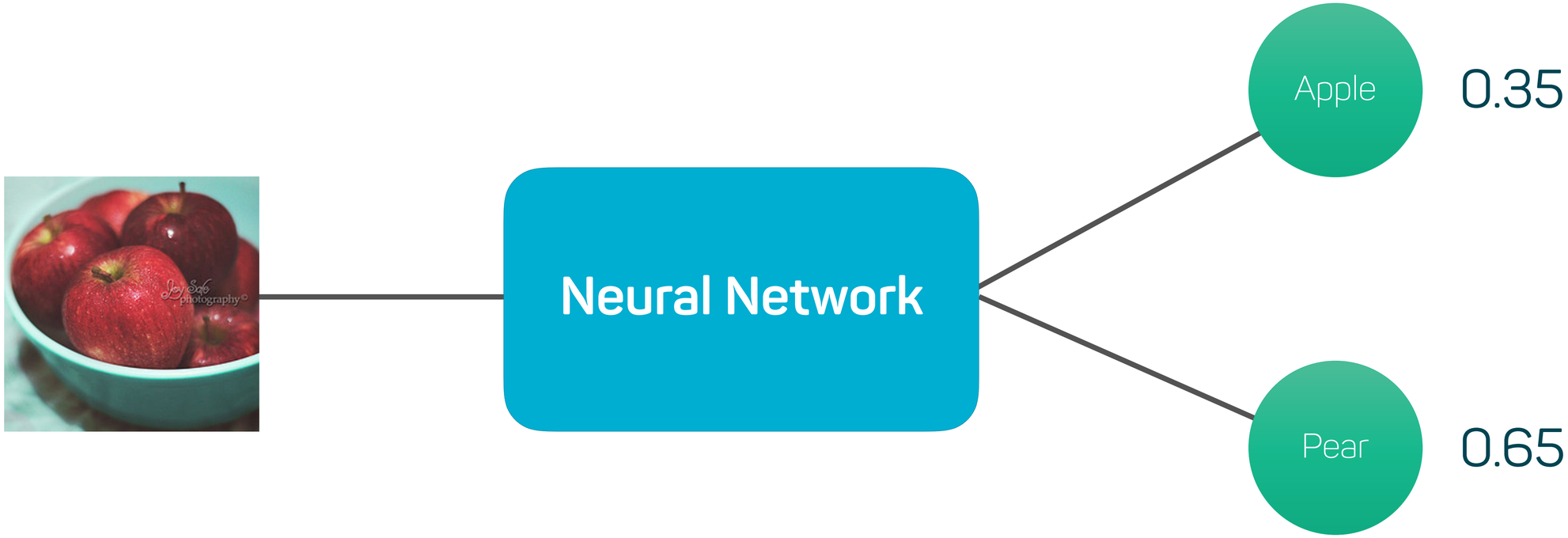
0.70

0.52

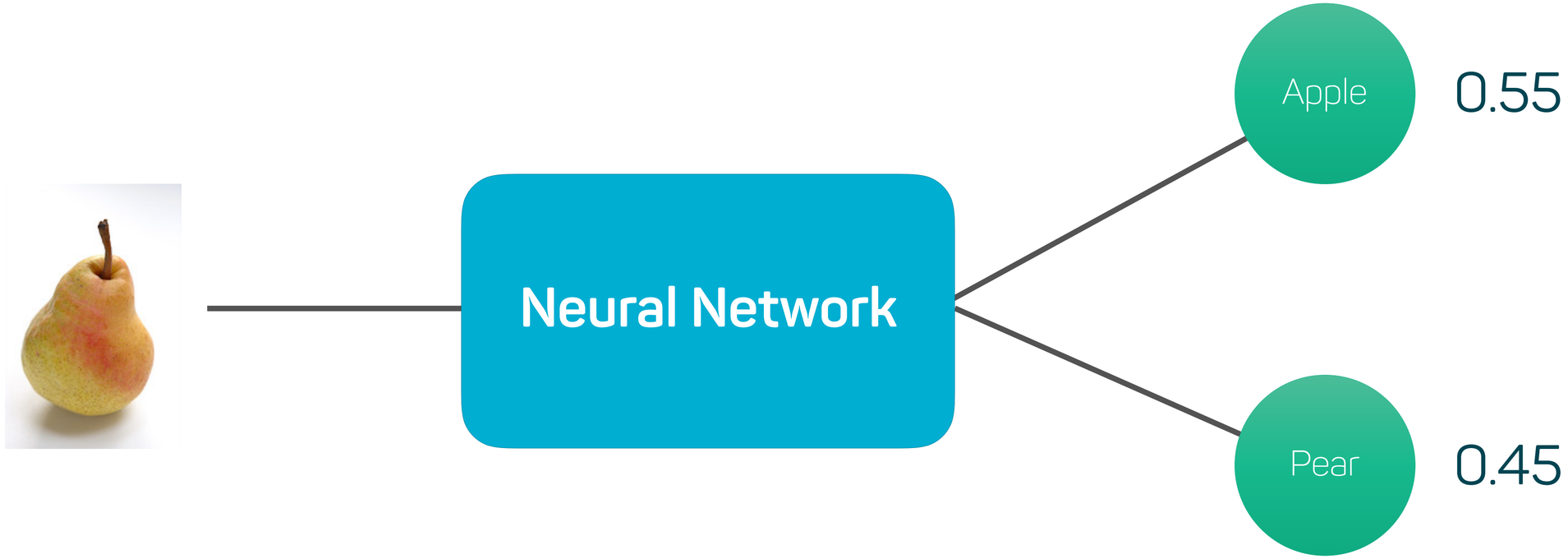
Loss

Back Propagation

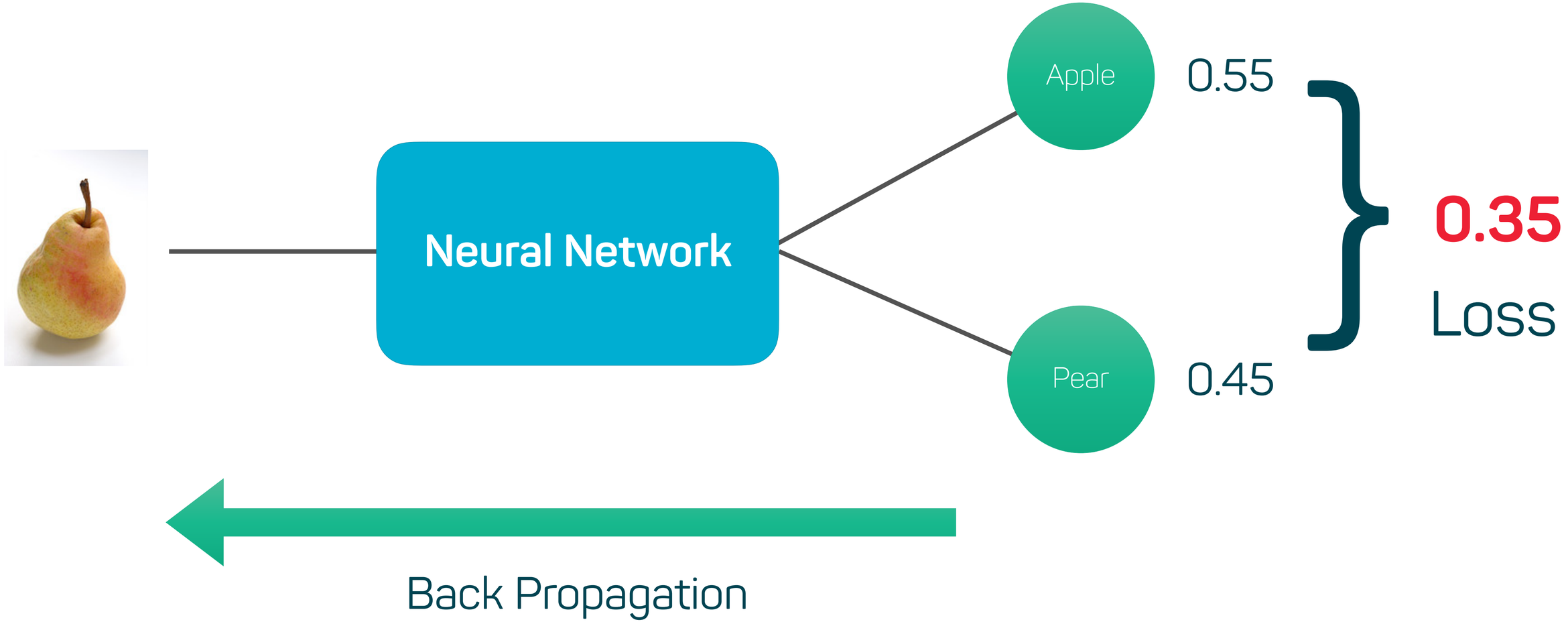
Fundamentals of Deep Learning



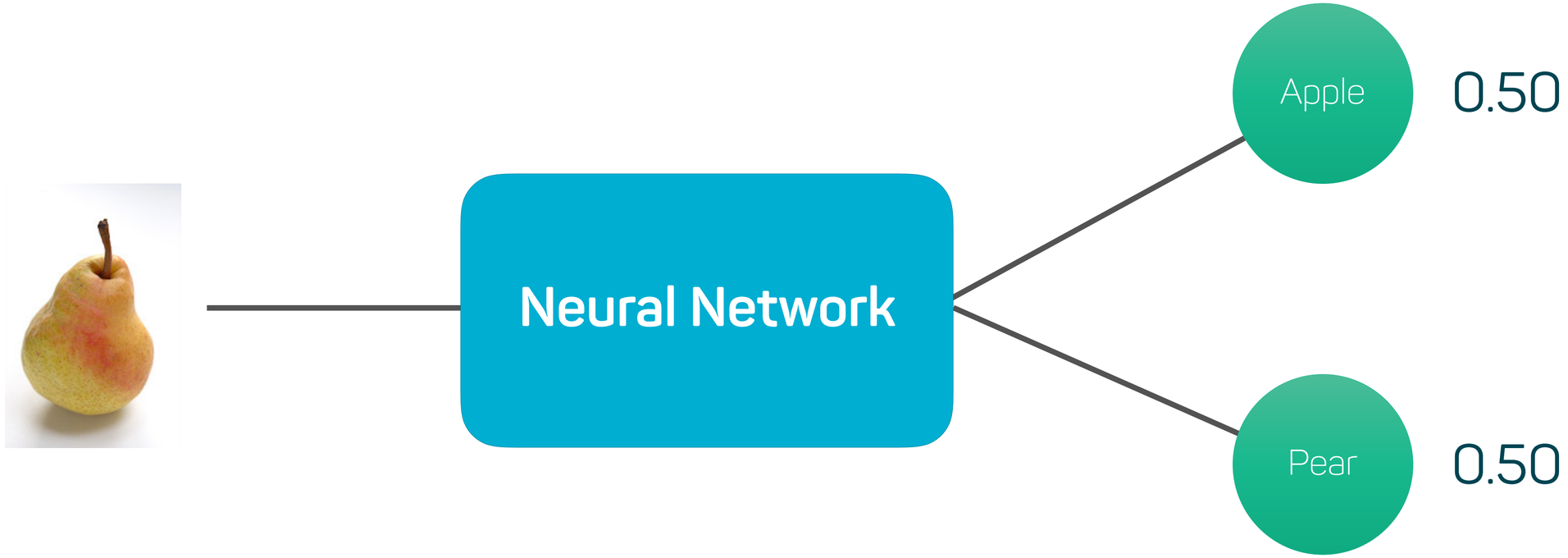
Fundamentals of Deep Learning



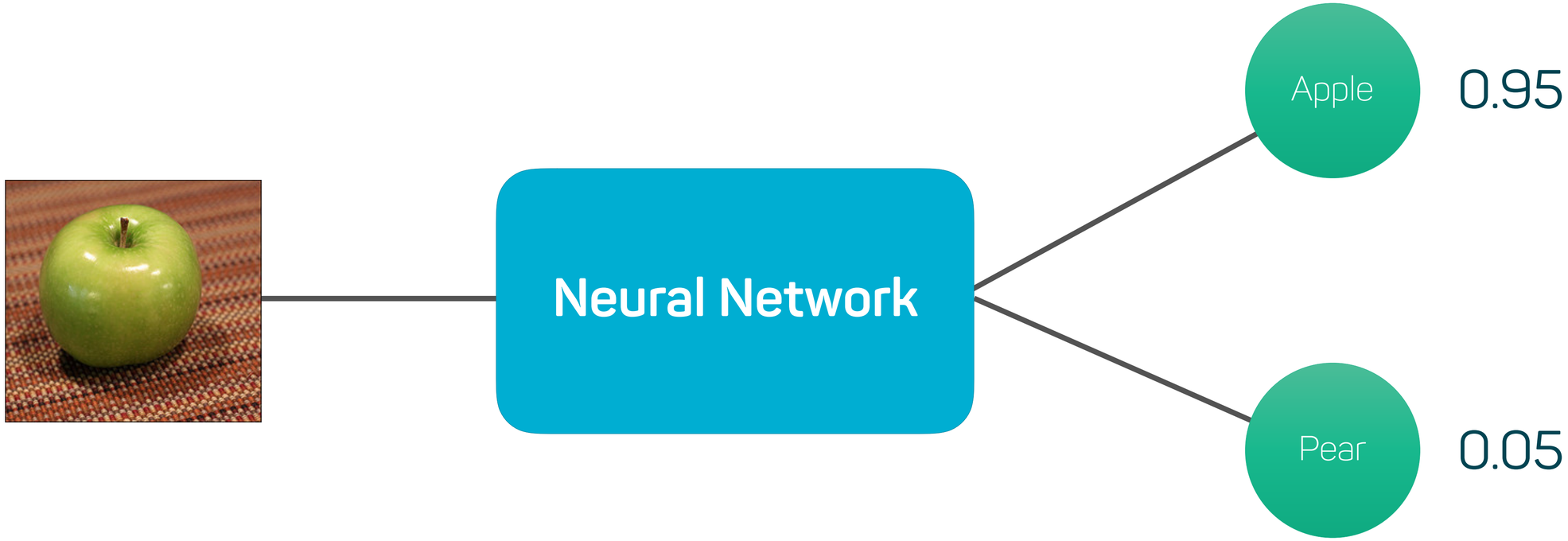
Fundamentals of Deep Learning



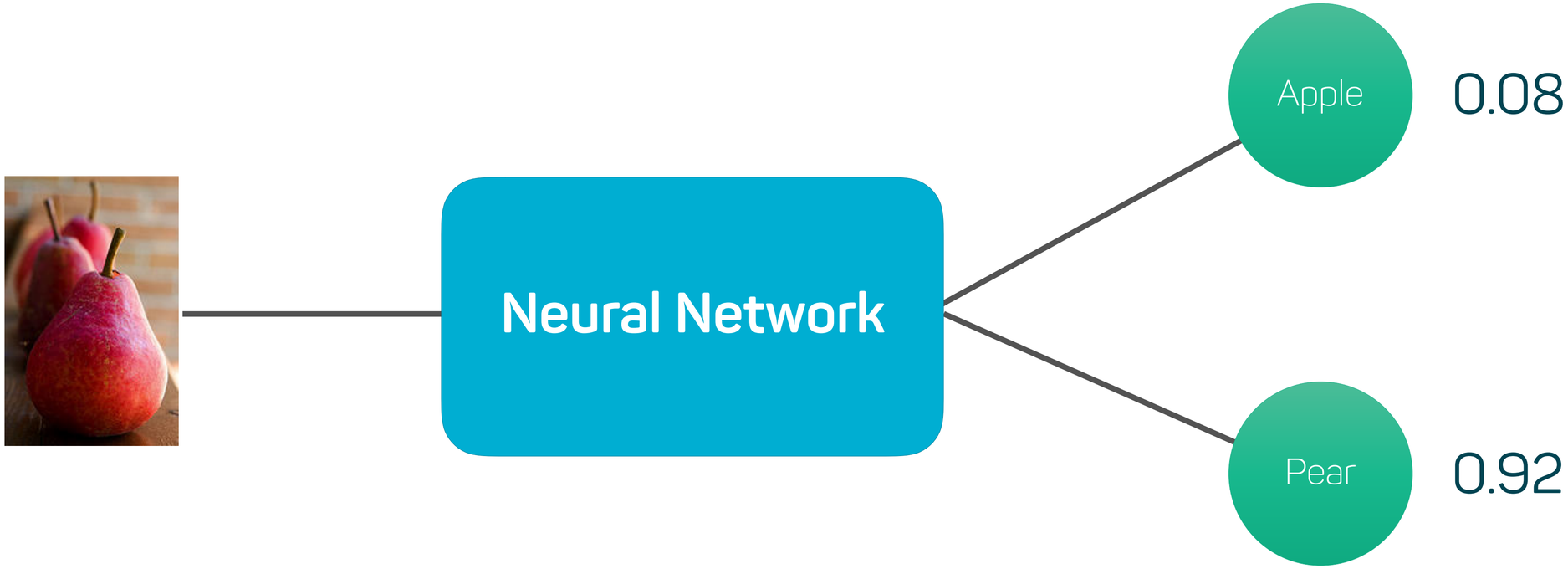
Fundamentals of Deep Learning



Fundamentals of Deep Learning

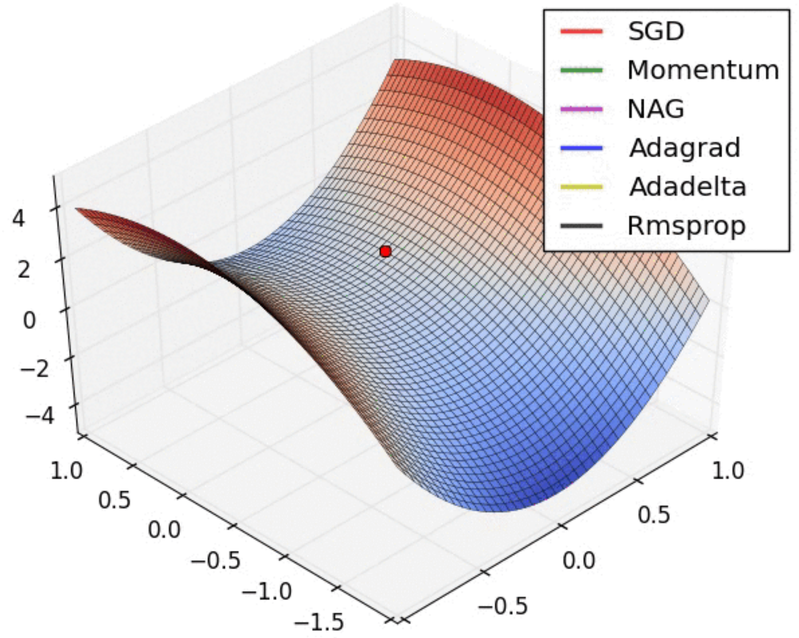


Fundamentals of Deep Learning



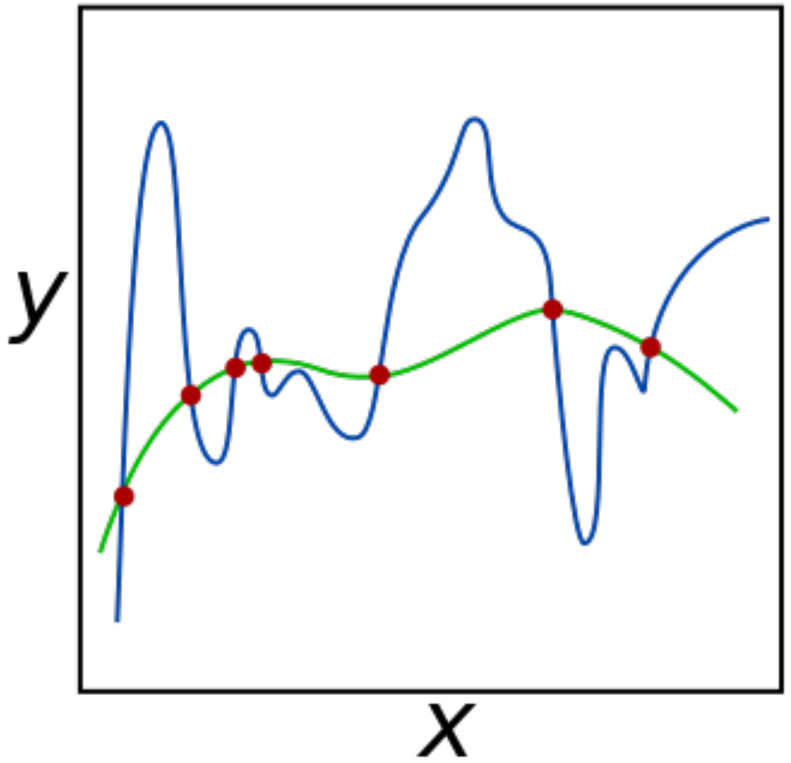
Fundamentals of Deep Learning - Optimizer

- Strategy to minimize loss
- Core idea: **stochastic gradient descent**



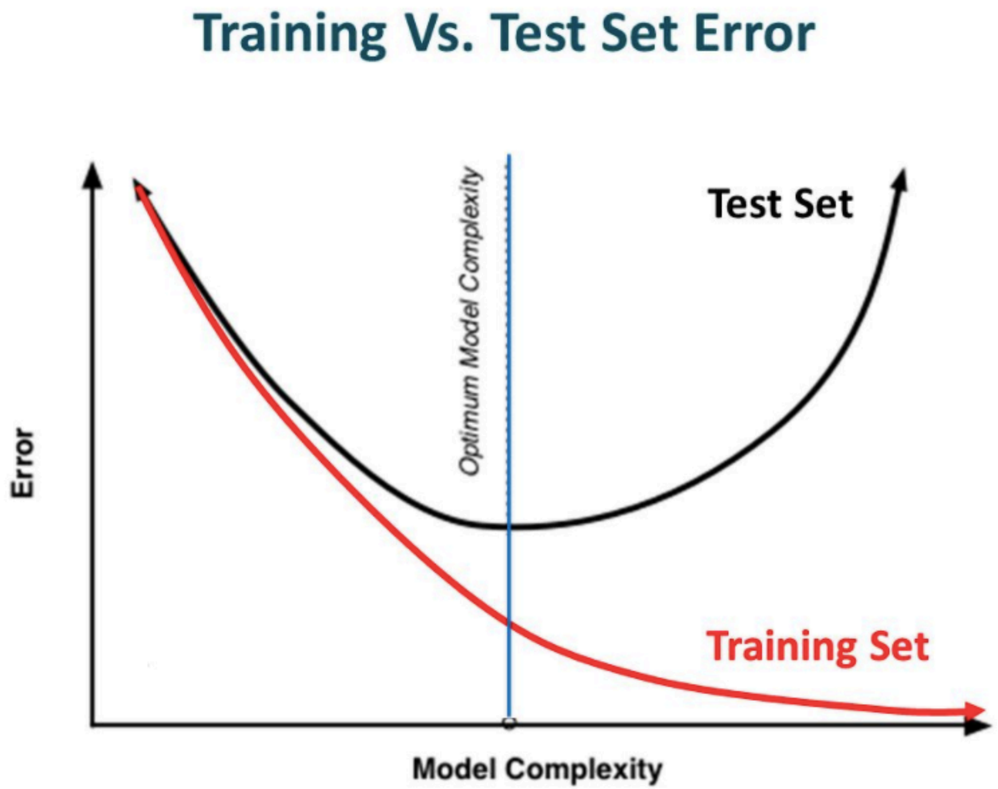
- Momentum: Also consider past gradients
- AdaGrad/**RMSProp**/AdaDelta: Adaptive learning rate based on past gradients
- **Adam**/RAdam: Momentum + RMSProp

Fundamentals of Deep Learning - Overfitting



https://en.wikipedia.org/wiki/Regularization_(mathematics)

Fundamentals of Deep Learning - Overfitting

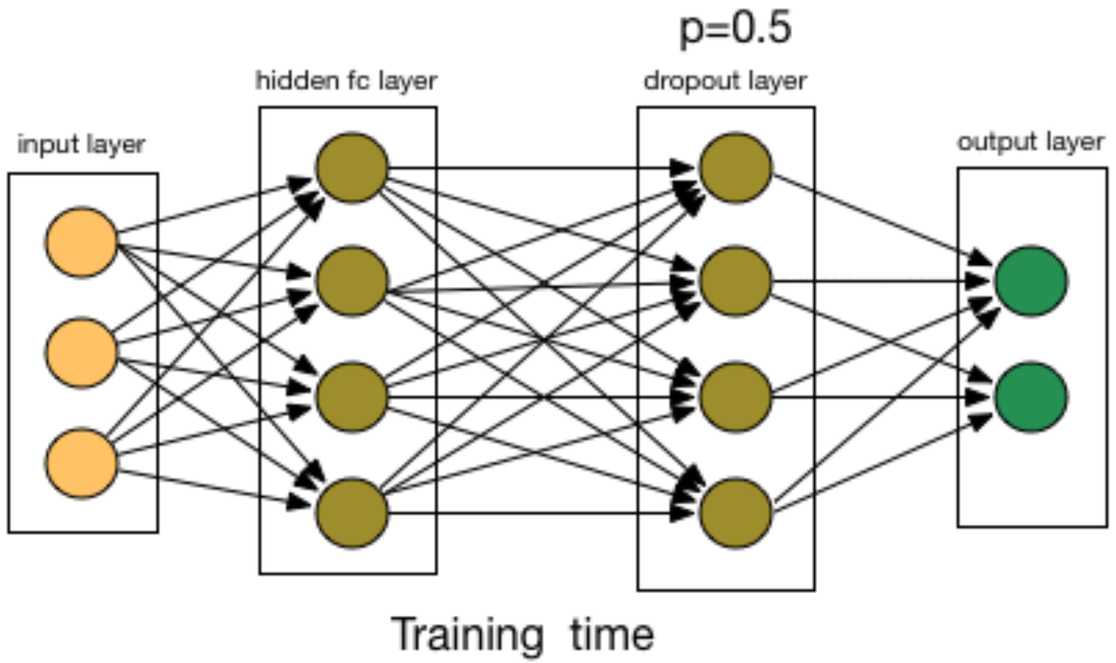


https://www.analyticsvidhya.com/blog/2018/04/fundamentals-deep-learning-regularization-techniques/

Fundamentals of Deep Learning - Data Augmentation

https://www.analyticsvidhya.com/blog/2018/04/fundamentals-deep-learning-regularization-techniques/

Fundamentals of Deep Learning - Dropout

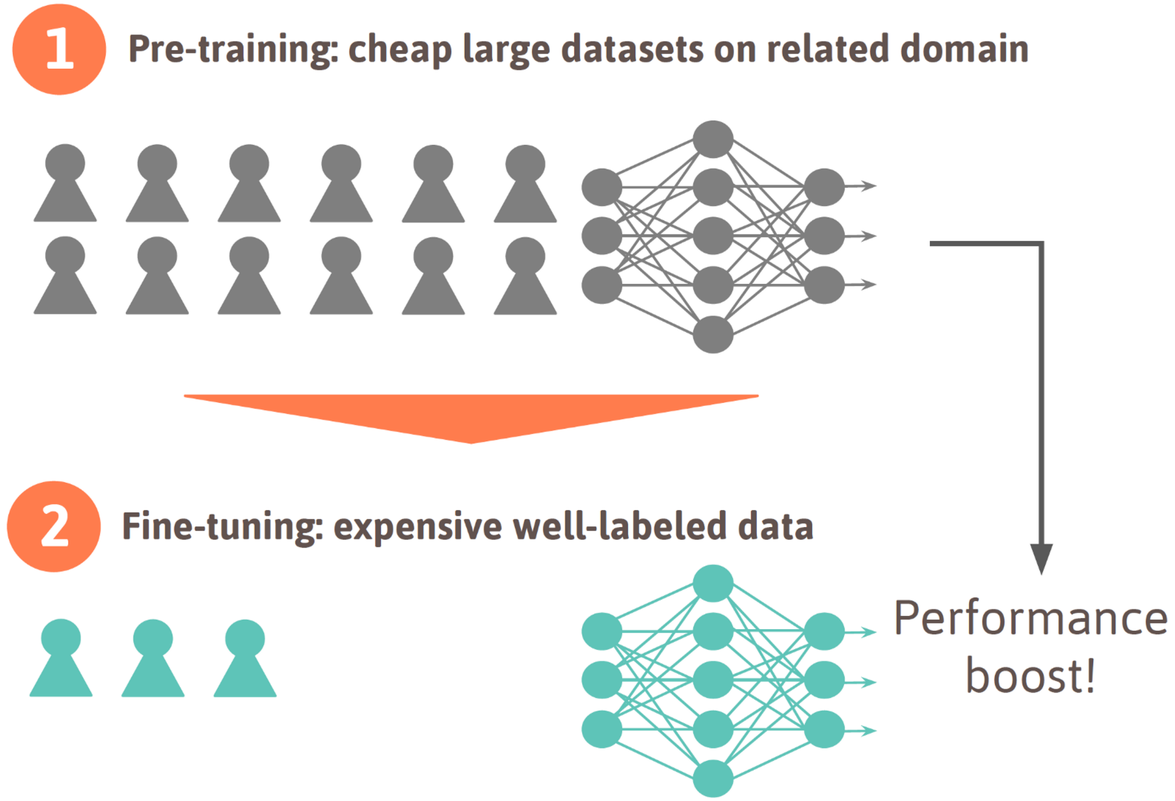


https://www.analyticsvidhya.com/blog/2018/04/fundamentals-deep-learning-regularization-techniques/

Fundamentals of Deep Learning - Early Stopping

https://www.analyticsvidhya.com/blog/2018/04/fundamentals-deep-learning-regularization-techniques/

Fundamentals of Deep Learning - Pretraining

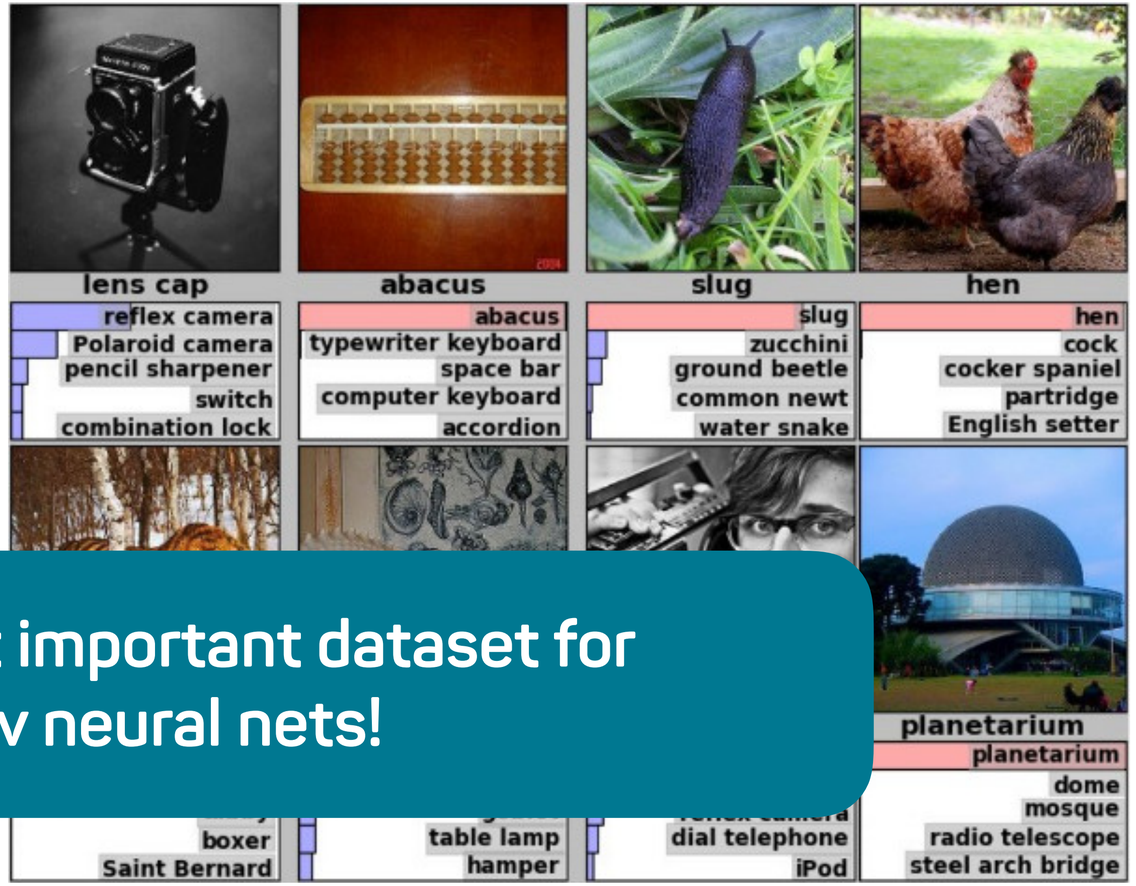


https://medium.com/merantix/applying-deep-learning-to-real-world-problems-ba2d86ac5837

Deep Learning Achievements - Pretraining

IMAGENET

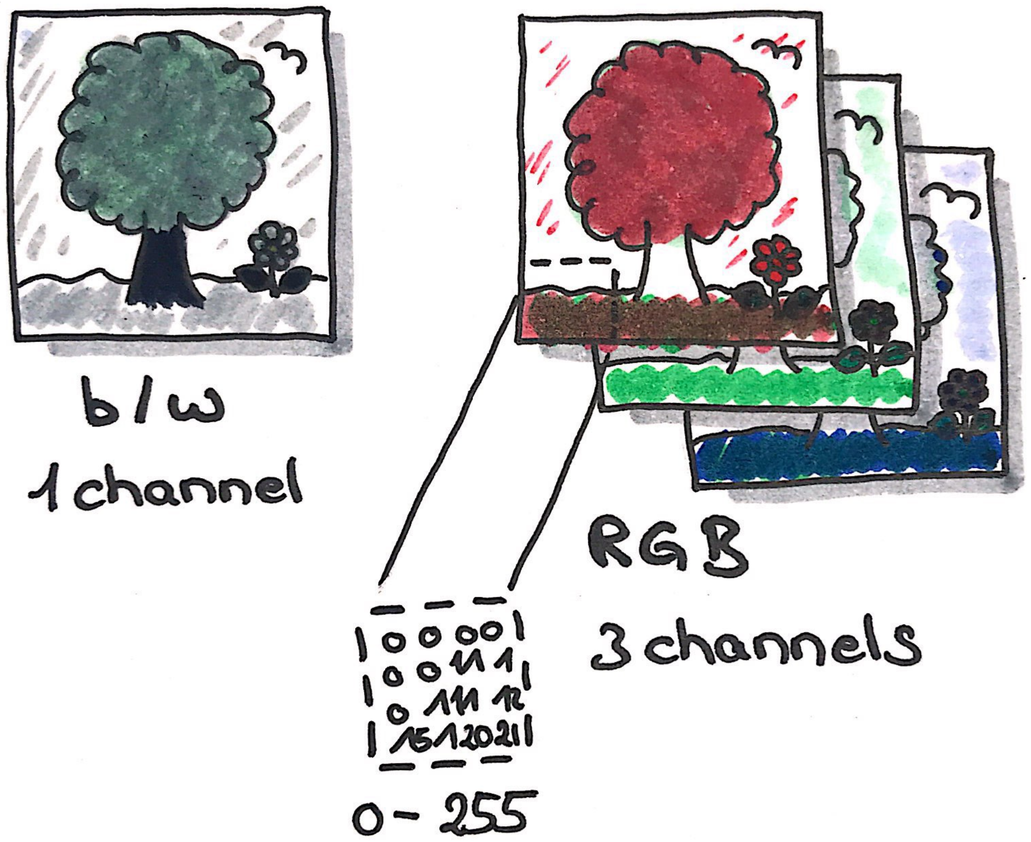
- 1000 categories
- trai
- test



ImageNet is the most important dataset for pretraining cv neural nets!

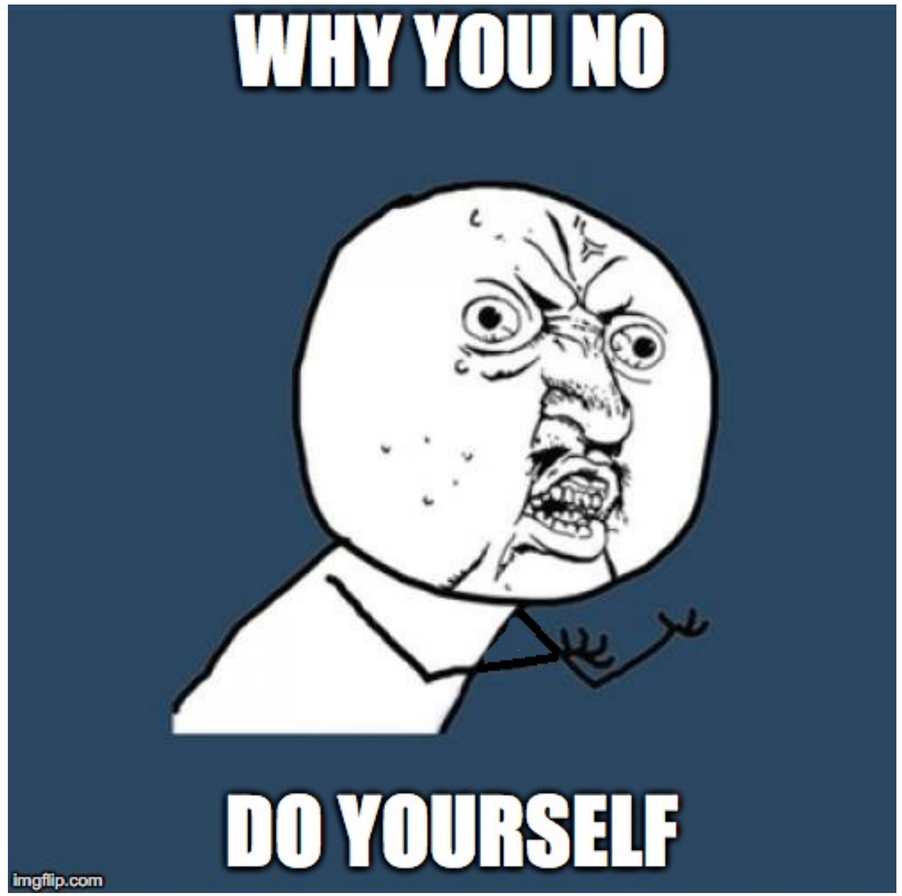
https://medium.com/obvious-ventures/our-investment-in-darwinai-d5ea1a7af32e

The Structure of an Image



Let's get Practical: JupyterLab, Python + Matplotlib

- Jupyter = Julia + Python + R
  - Interactive browser-based environment for script-based languages
- Matplotlib: Python spinoff of Matlabs plotting library



Jupyter tips'n'tricks

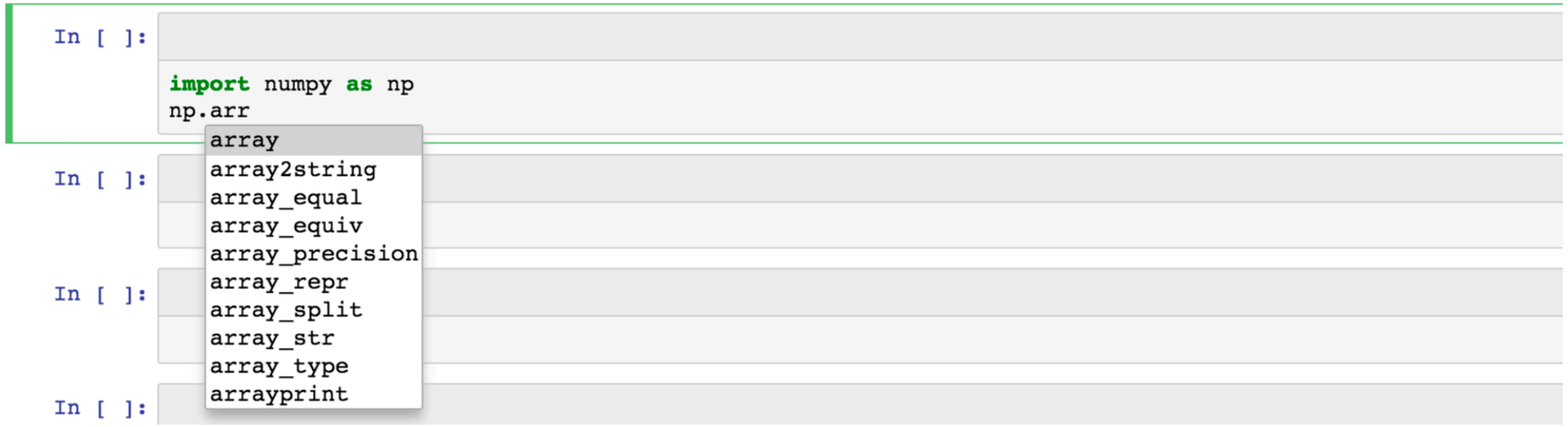
- ?: search documentation

```
In [5]: len?
```

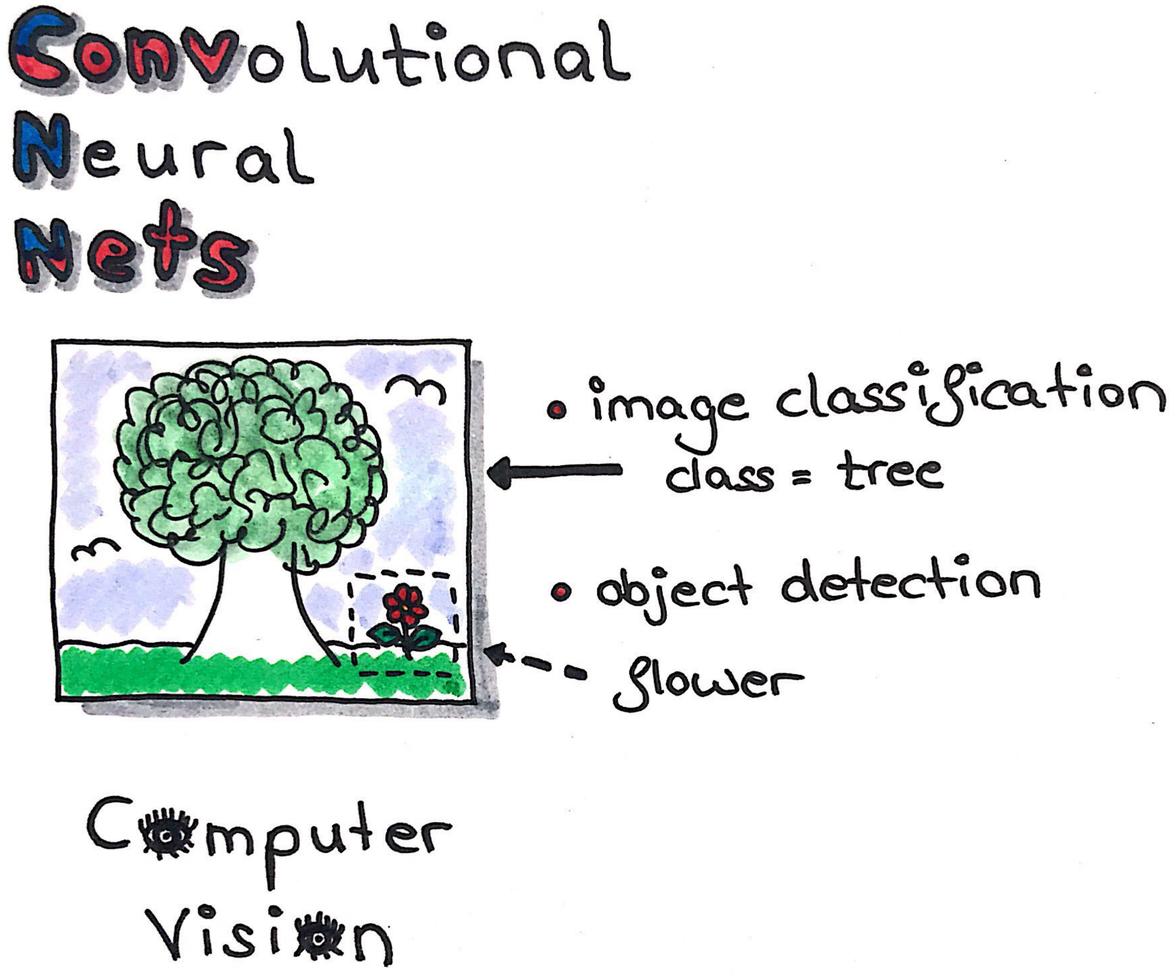
- ??: look at source code

```
In [6]: len??
```

- tab: autocomplete
- shift + tab: look at function call



How does a Computer Learn to See?



Convolutional Neural Networks (CNNs)

Fully Connected

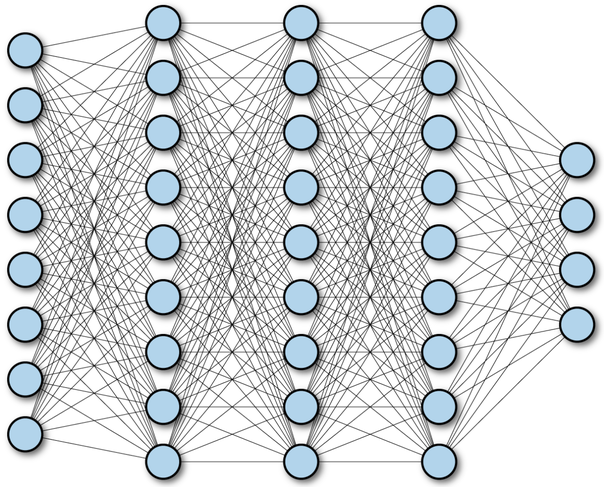


image: www.kdnuggets.com

CNNs

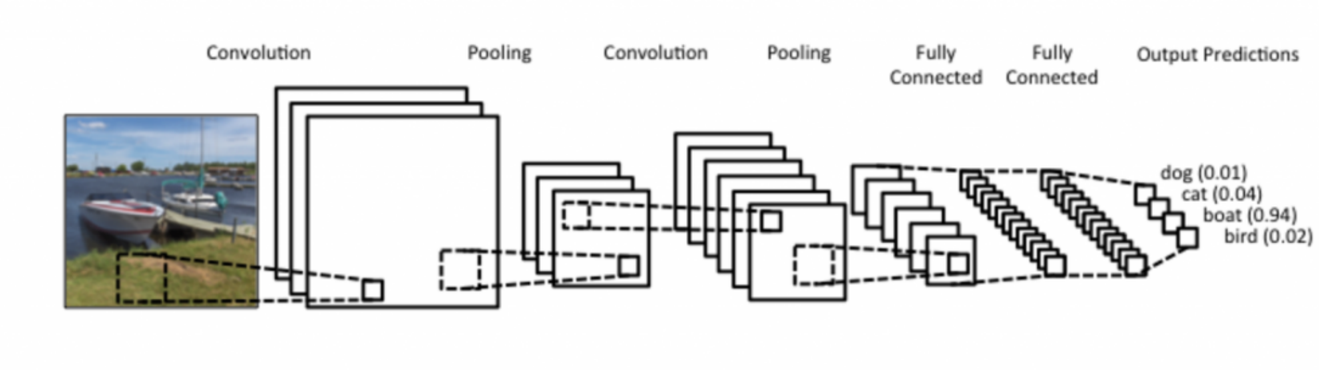
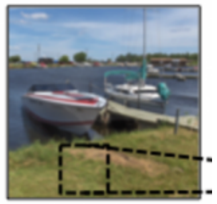


image: www.kdnuggets.com

Convolutional Neural Networks (CNNs)

Input Image
(128x128x3)



Fully Connected

Connections First Layer
(512 nodes)

25.165.824
weights

CNNs

Connections First Layer
(Kernel 3x3, Filters 16)

432
weights

Convolutional Layer

Input Output

3_0	3_1	2_2	1	0
0_2	0_2	1_0	3	1
3_0	1_1	2_2	2	3
2	0	0	2	2
2	0	0	0	1

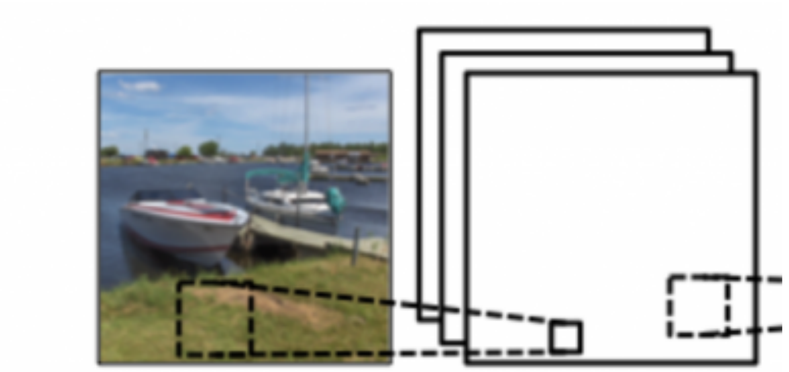
12	12	17
10	17	19
9	6	14

Kernel

$$\begin{pmatrix} 0 & 1 & 2 \\ 2 & 2 & 0 \\ 0 & 1 & 2 \end{pmatrix}$$

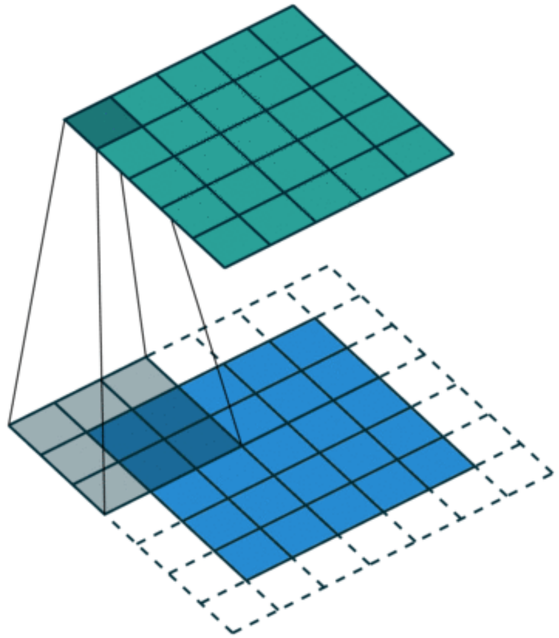
image: www.deeplearning.net

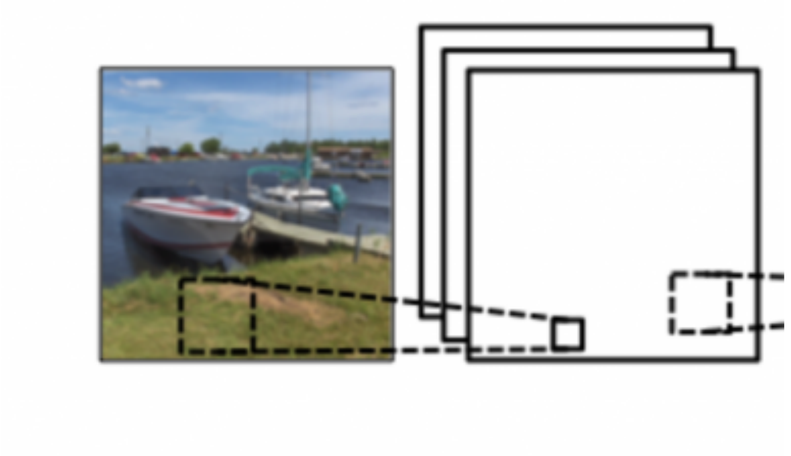
Input Output



Kernel is learned

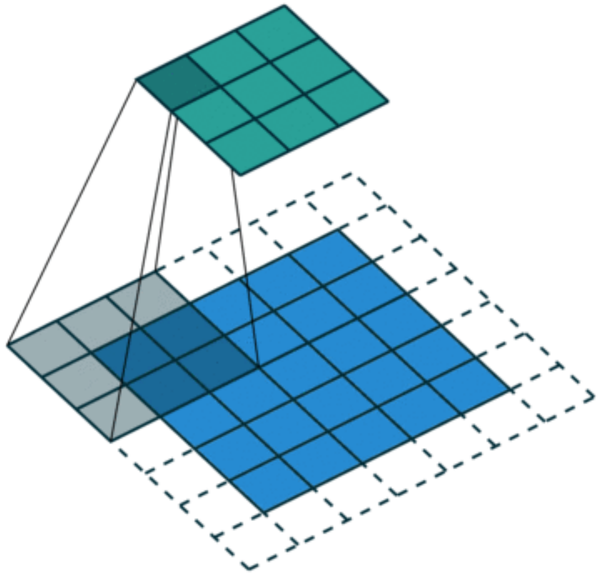
Convolutional Layer - Zero Padding





Convolutional Layer - Strides and Pooling Layer

Stride of 2



Maxpool

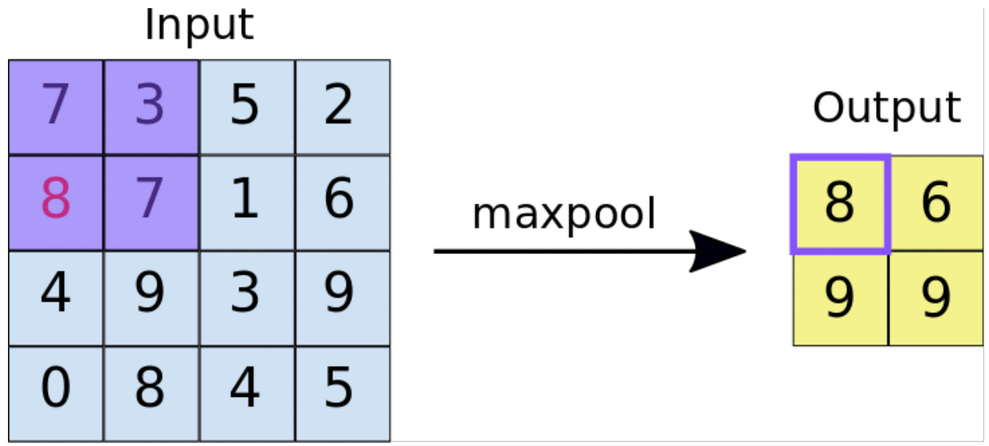
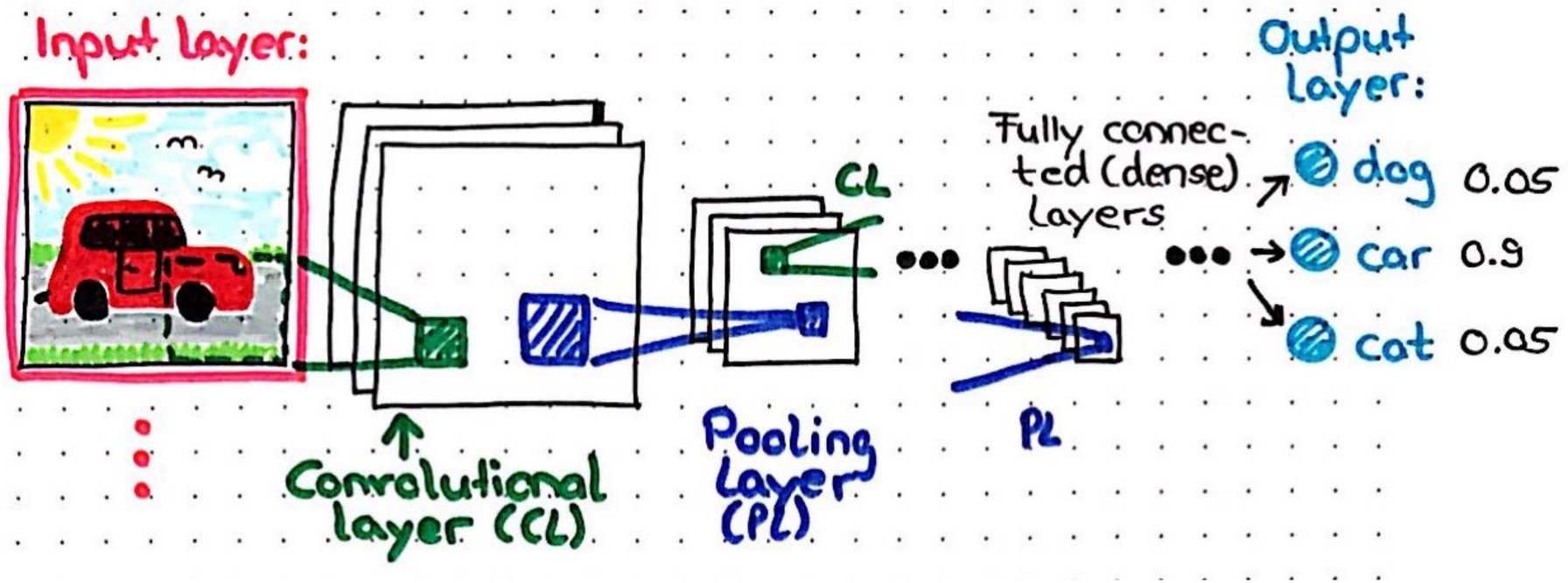
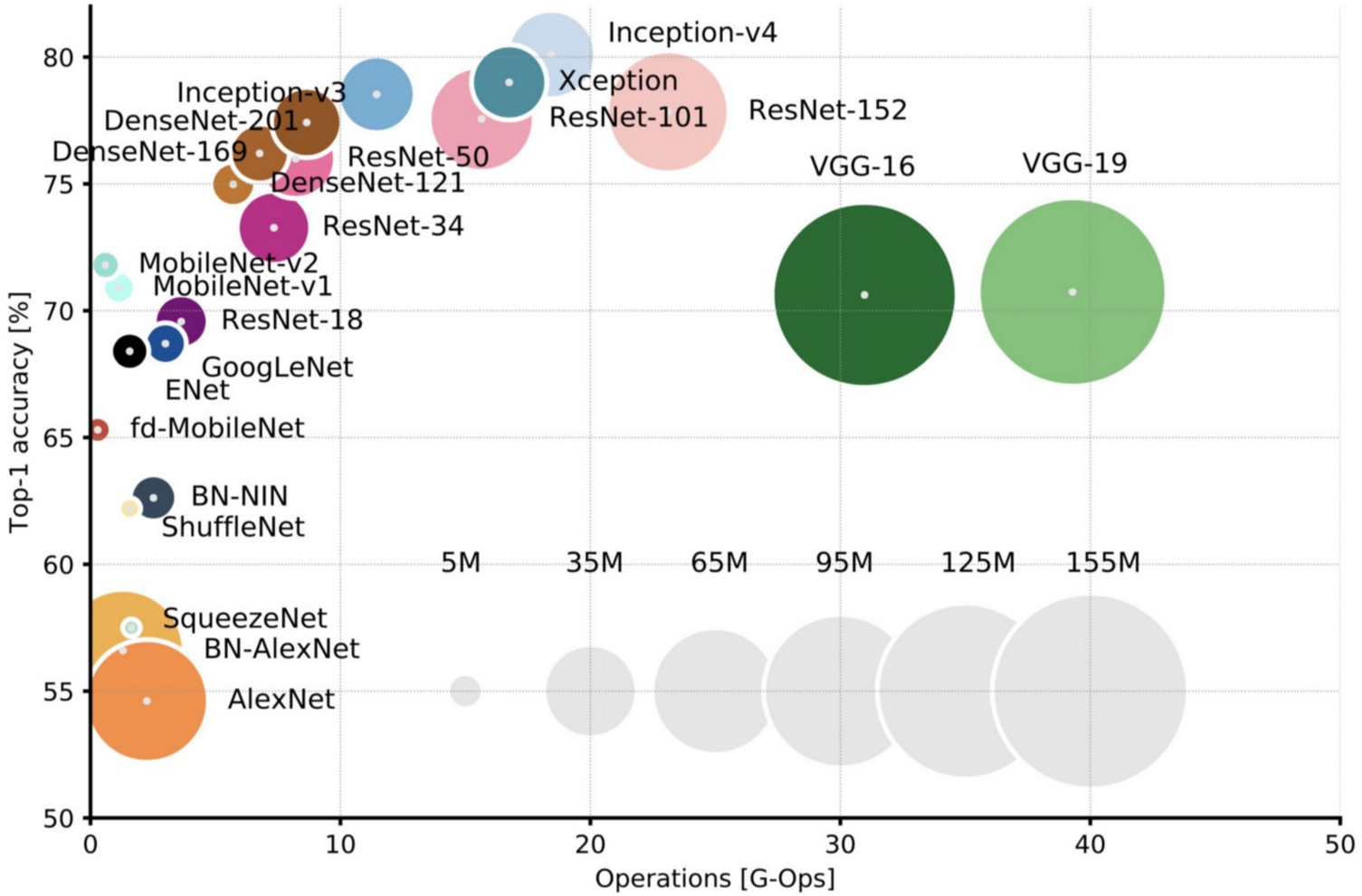


image: developers.google.com

Convolutional Neural Network



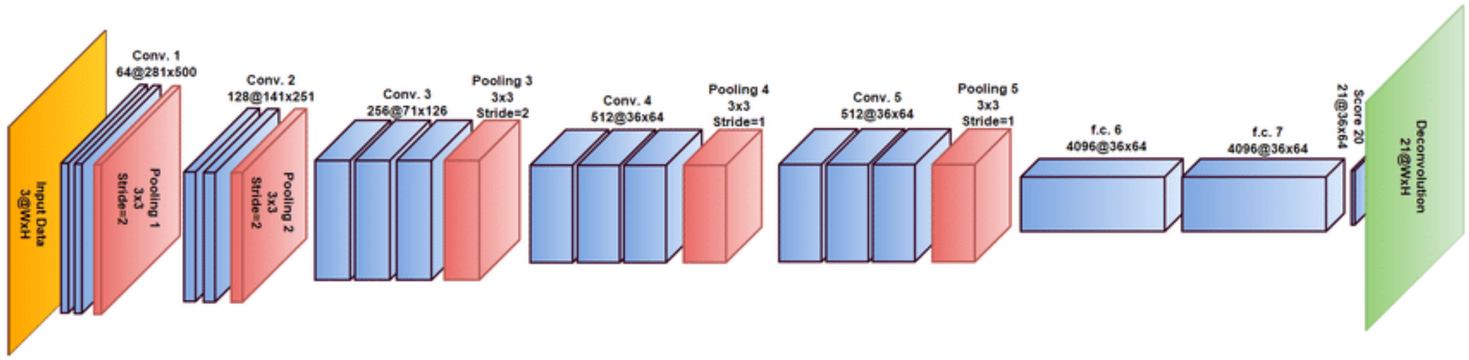
Important CNN Architectures



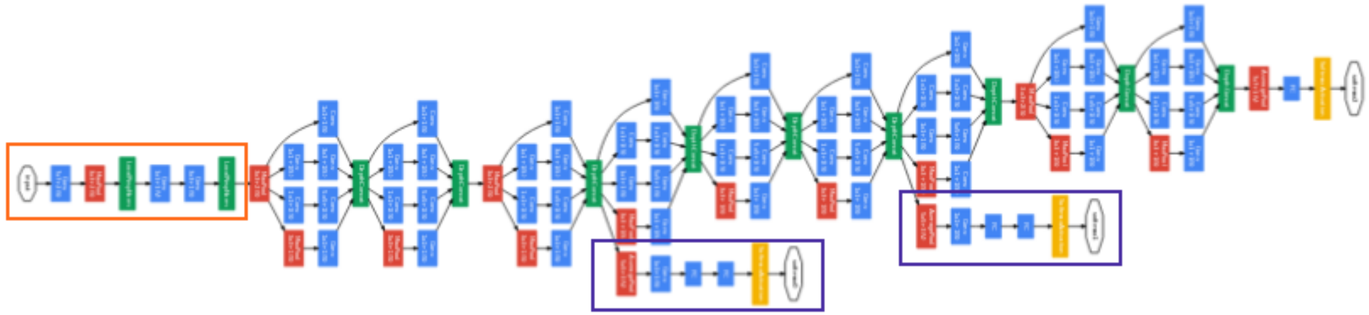
Relevant Architectures

- AlexNet (2012)
- VGG-16 (2014)
- Inception (2014)
- ResNet (2015)
- MobileNet (2017)

Important CNN Architectures

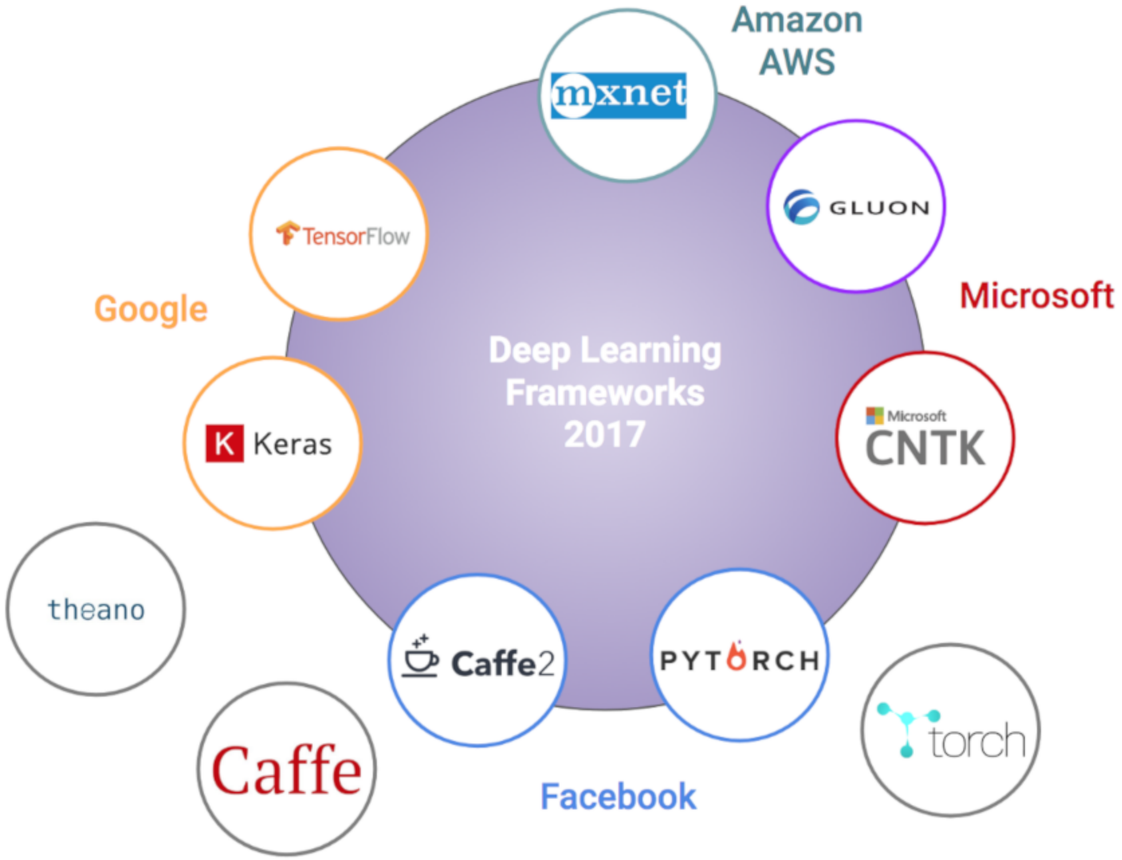


VGG



Inception

Deep Learning Frameworks



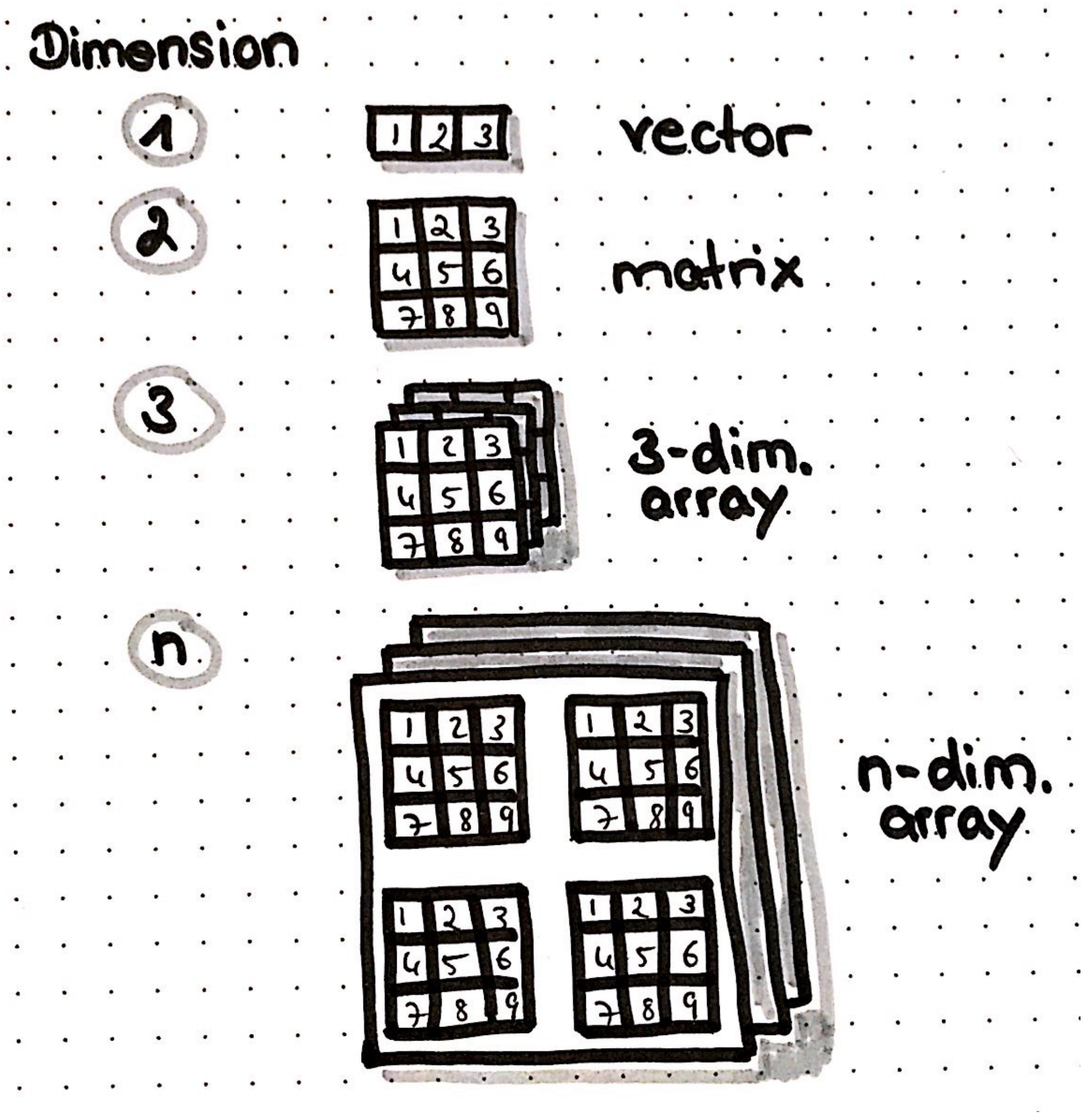
https://devopedia.org/deep-learning-frameworks

Tensorflow


ONE DOESN'T SIMPLY CALL ANY MULTI-DIMENSIONAL THING
A TENSOR
memegenerator.net

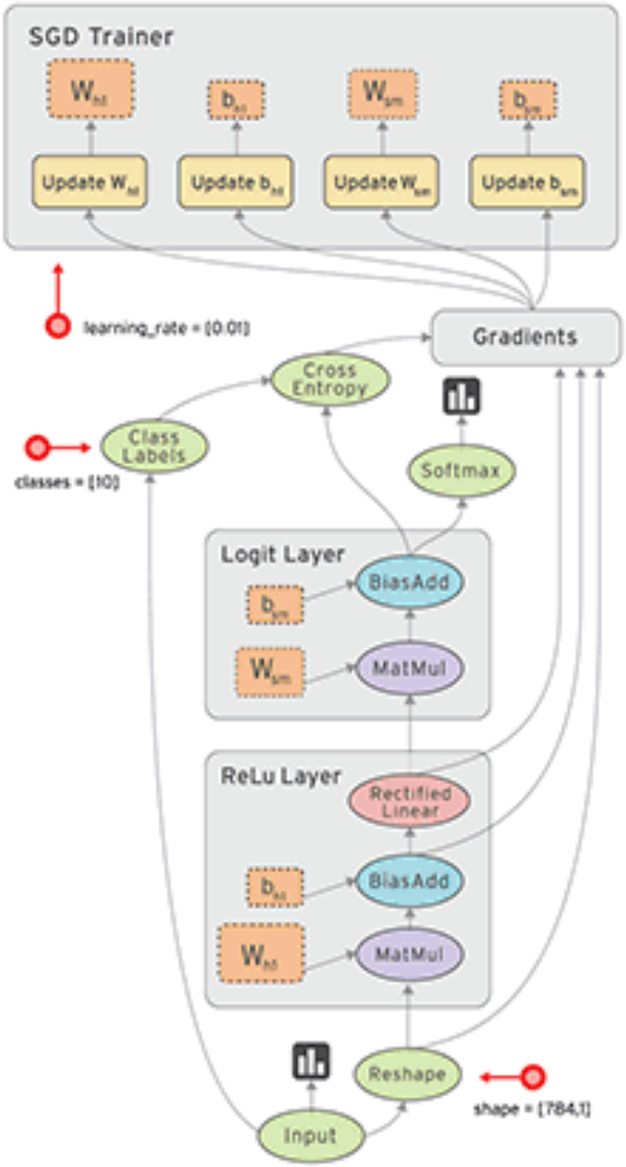
Tensorflow

- Tensors = Multidimensional Arrays



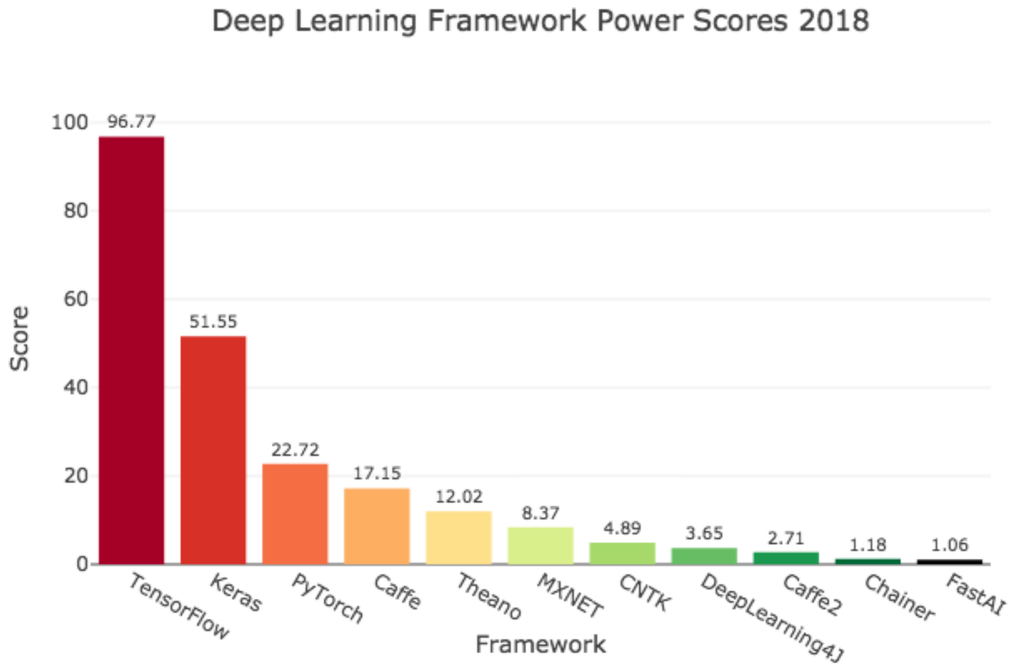
Tensor-"Flow"

- Compilation of static dataflow graph



Keras

- High-level API for Tensorflow, Theano and CNTK
- Part of Tensorflow Core API since v1.4
- User-friendly, extensible and flexible



Keras APIs

SEQUENTIAL MODELS

simple

suitable for most cases

linear order of layers

only one direction from input to output

FUNCTIONAL API

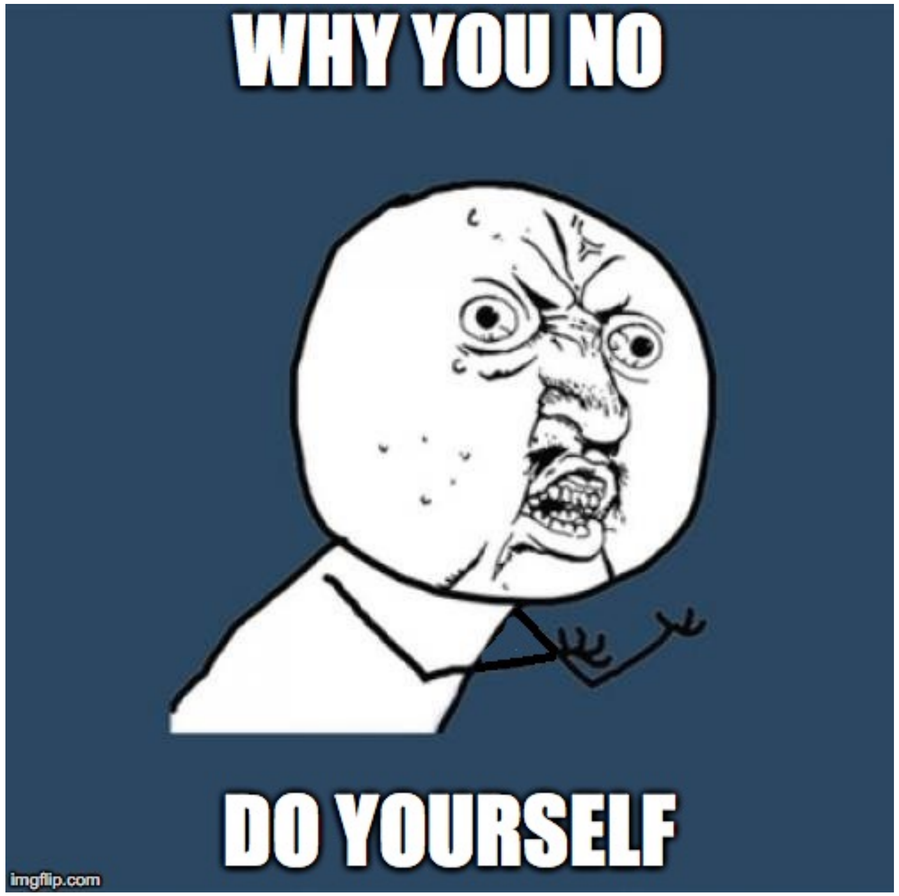
more complex

suitable for complex models

can have multiple in- or outputs

layers can be non-sequential, e.g. LSTM

Let's get Practical: Keras

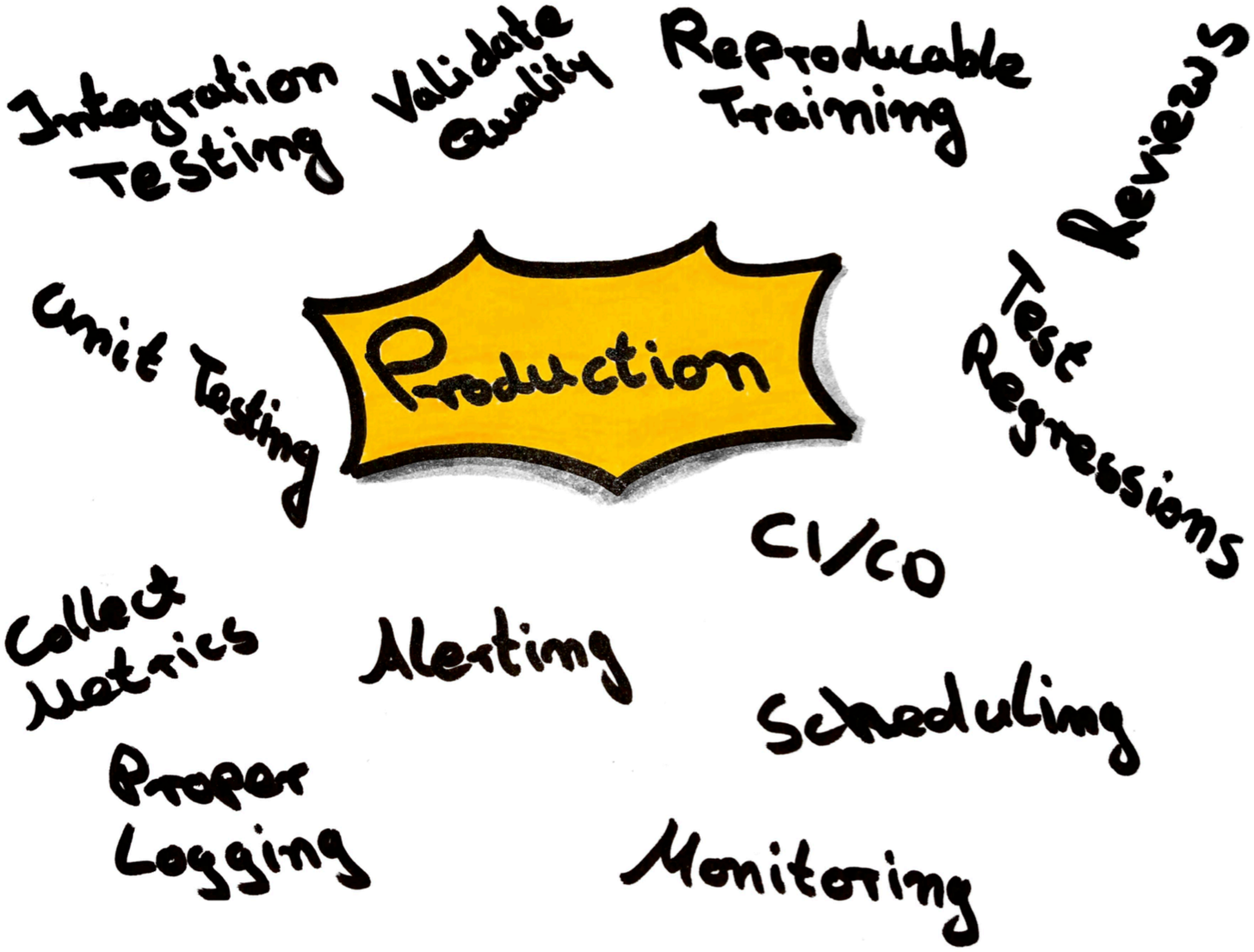


Experiment worked!

And now??

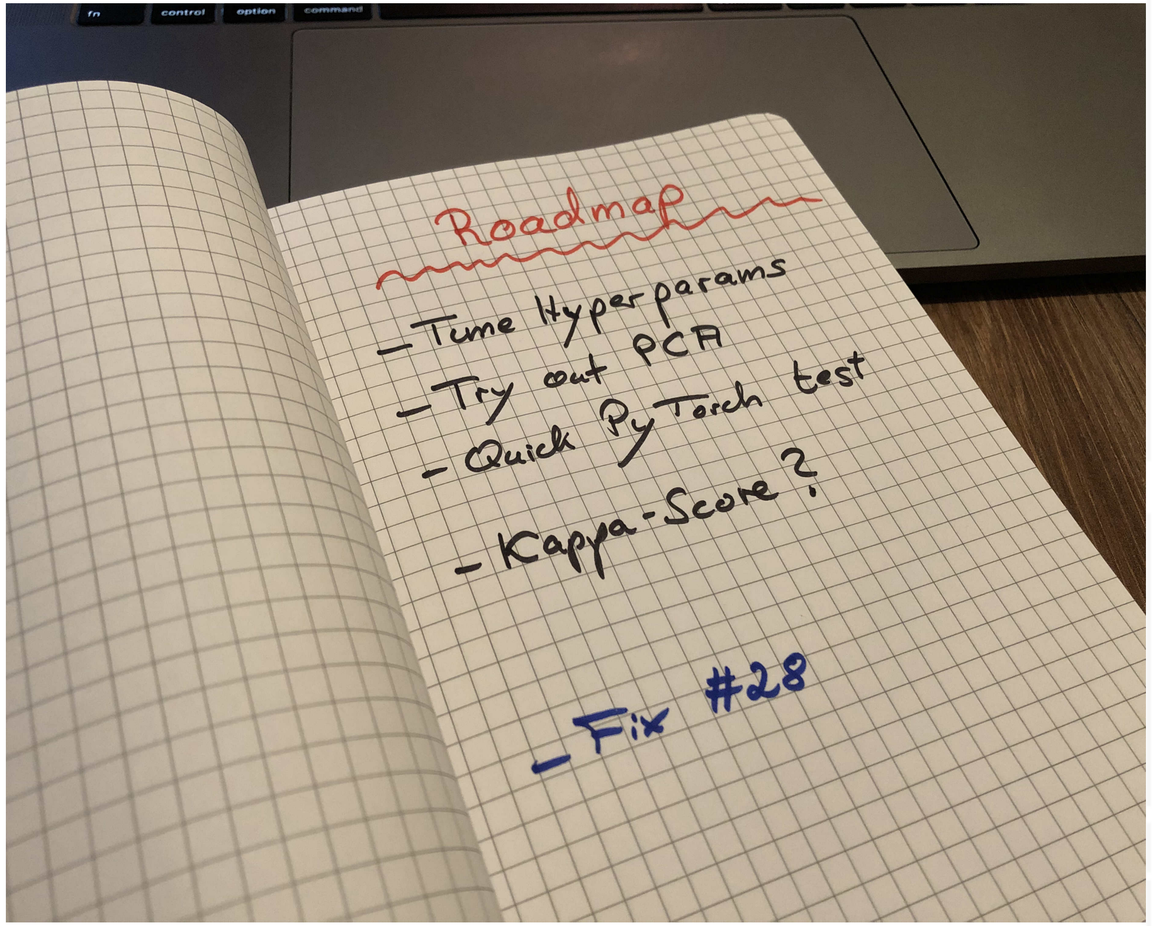
Just Deploy it, Okay?

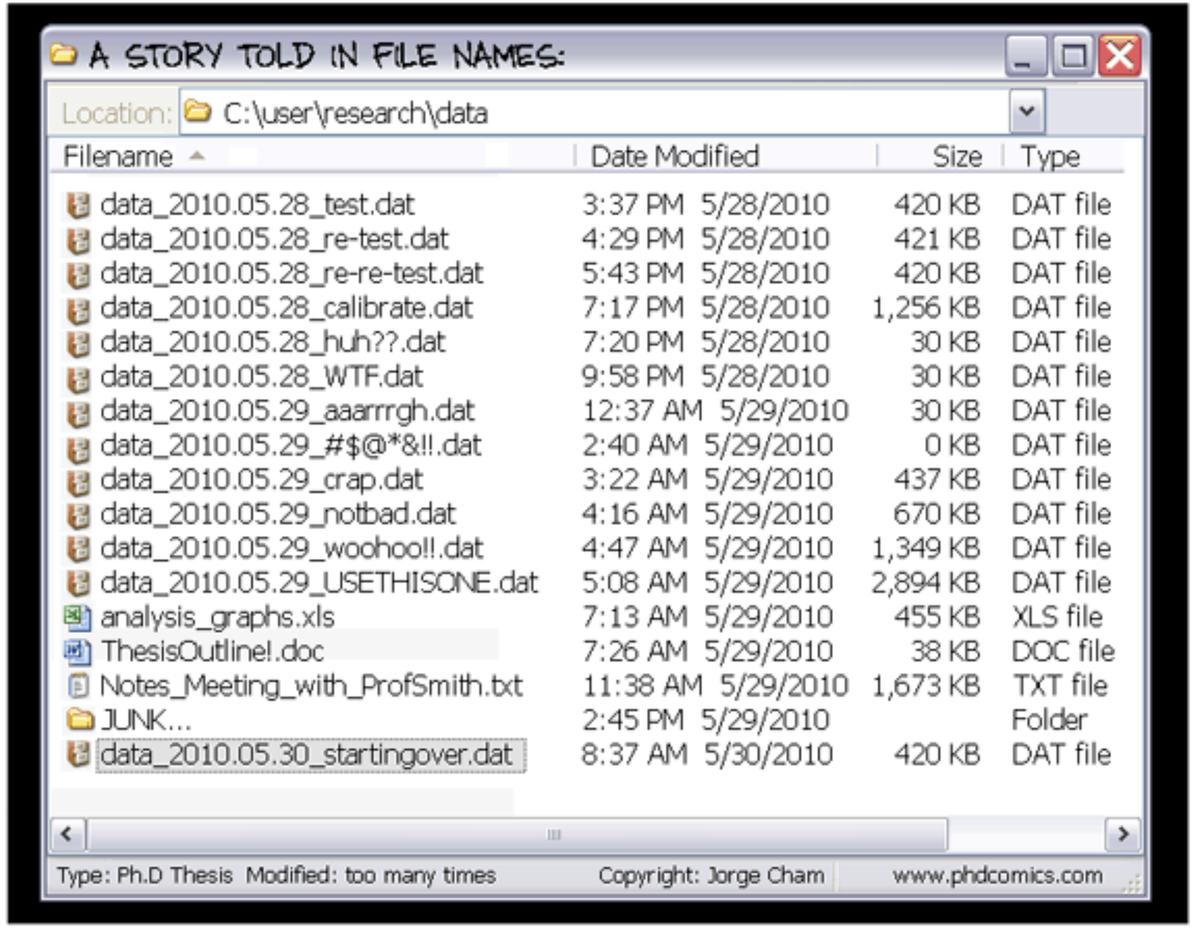
Production-Ready Data Science



Versioning and Reproducable Training

- Data Science is experiment-driven …





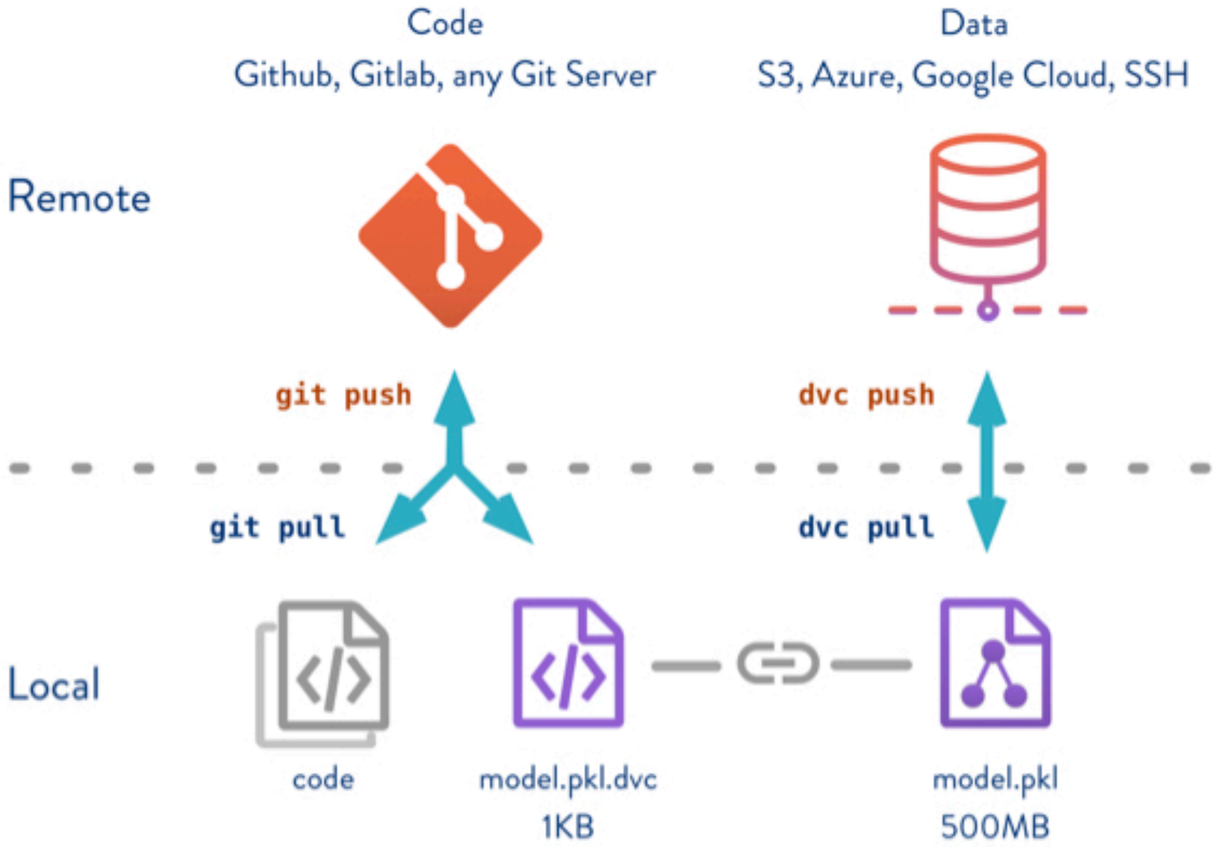
Reproducable Training

- ... and collaborative!

```
Tims-MacBook-Pro:code tsabsch$ git log --graph --decorate --oneline
*   fc88d36 (HEAD -> master) Merge branch 'tim_fix_zerodivisionerror'
|\
| * 2cc23f0 (tim_fix_zerodivisionerror) Fix ZeroDivisionError
* |   e486d76 Merge branch 'mark_batchnorm'
|\ \
| |/
|/|
| * b988f6e (mark_batchnorm) Decrease BN threshold
| * c83736d Add batch normalisation
* | f80b108 Add preprocessing
|/
* a87aa63 Add simple tensorflow model
* 7c0d21e Initial commit
Tims-MacBook-Pro:code tsabsch$
```

DVC to the Rescue!

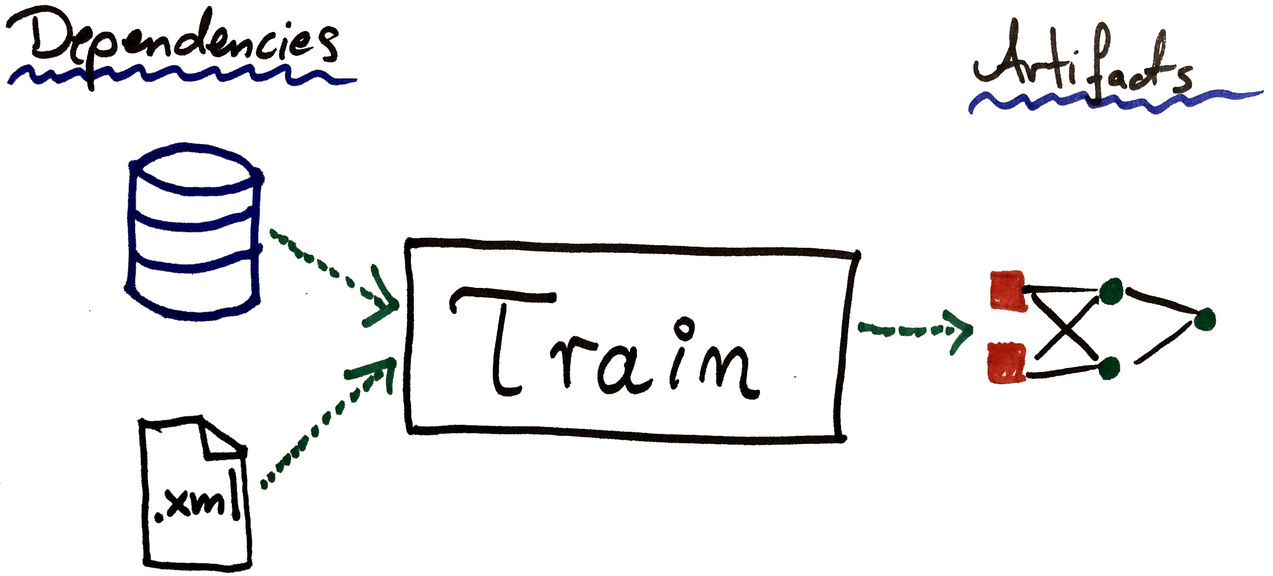
- **Large File Storage**, similar to git-lfs
- Metadata in git (name, hashsum etc.)
- Contents in cache + remote storage



https://dvc.org

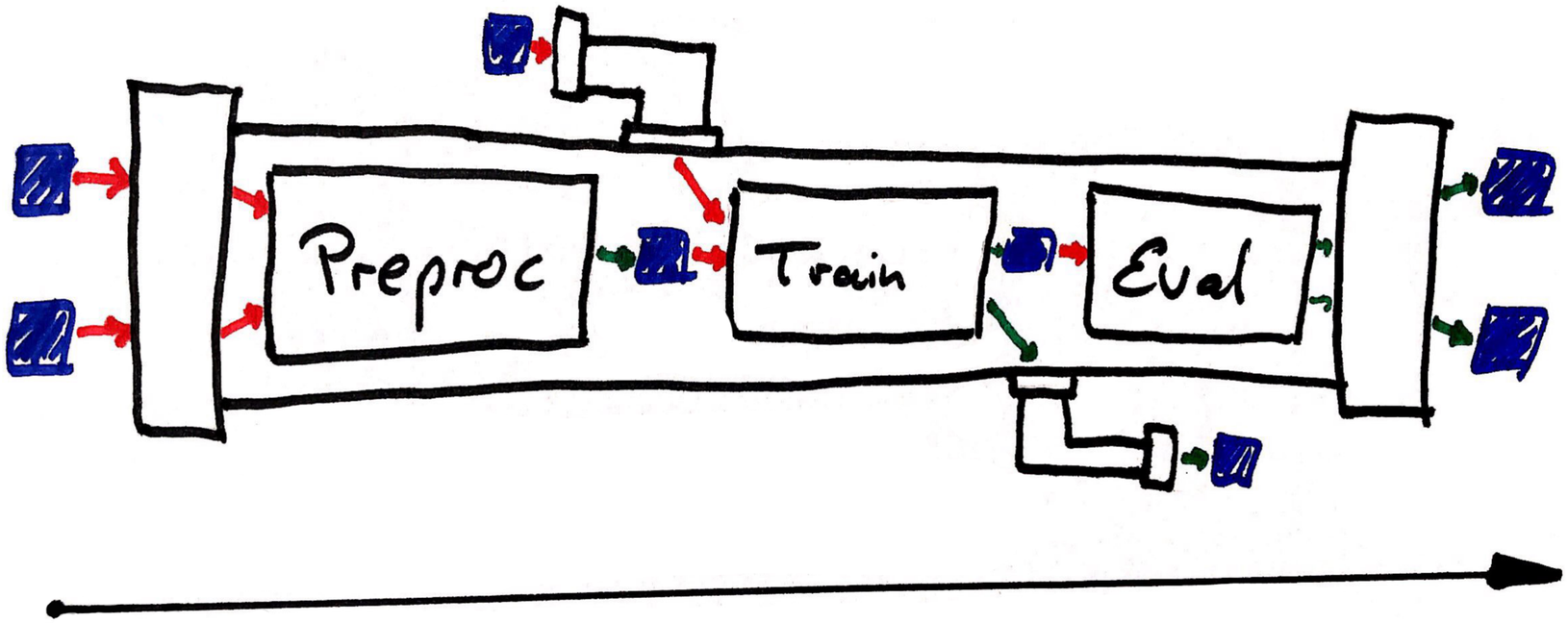
DVC to the Rescue!

- **Pipelines**
- Optimise reproduction by splitting your process into stages
- Stages store dependencies, processing and output artefacts

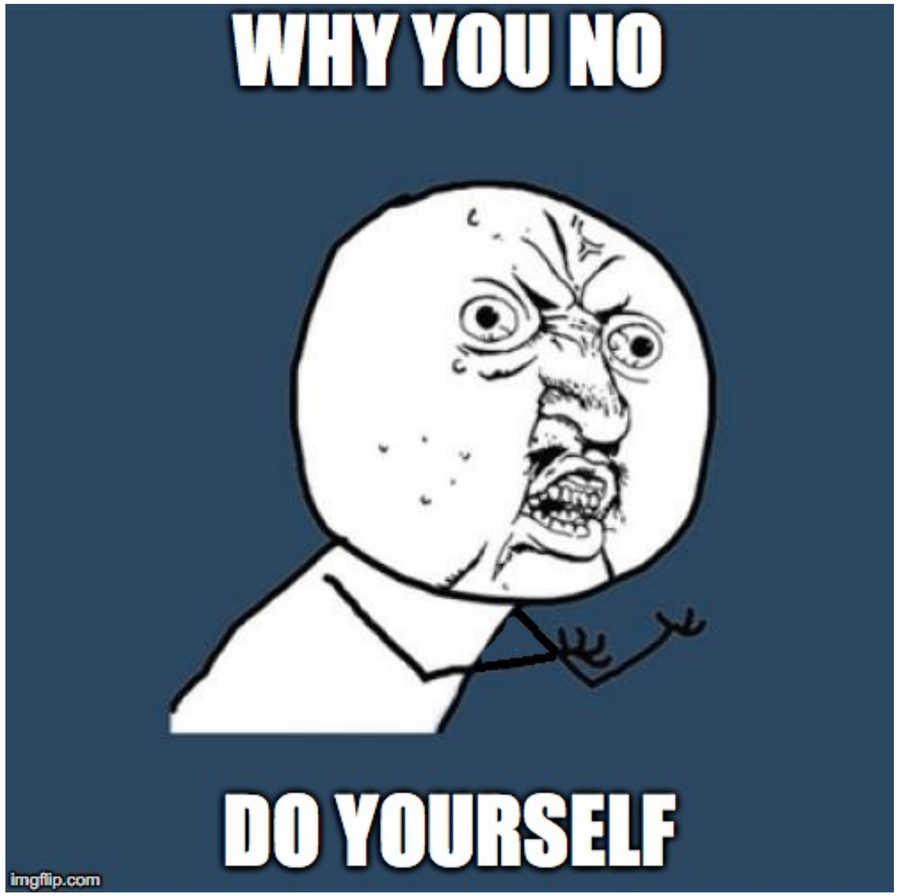


DVC to the Rescue!

- **Pipelines**
- Connect stages to a pipelines via dependencies



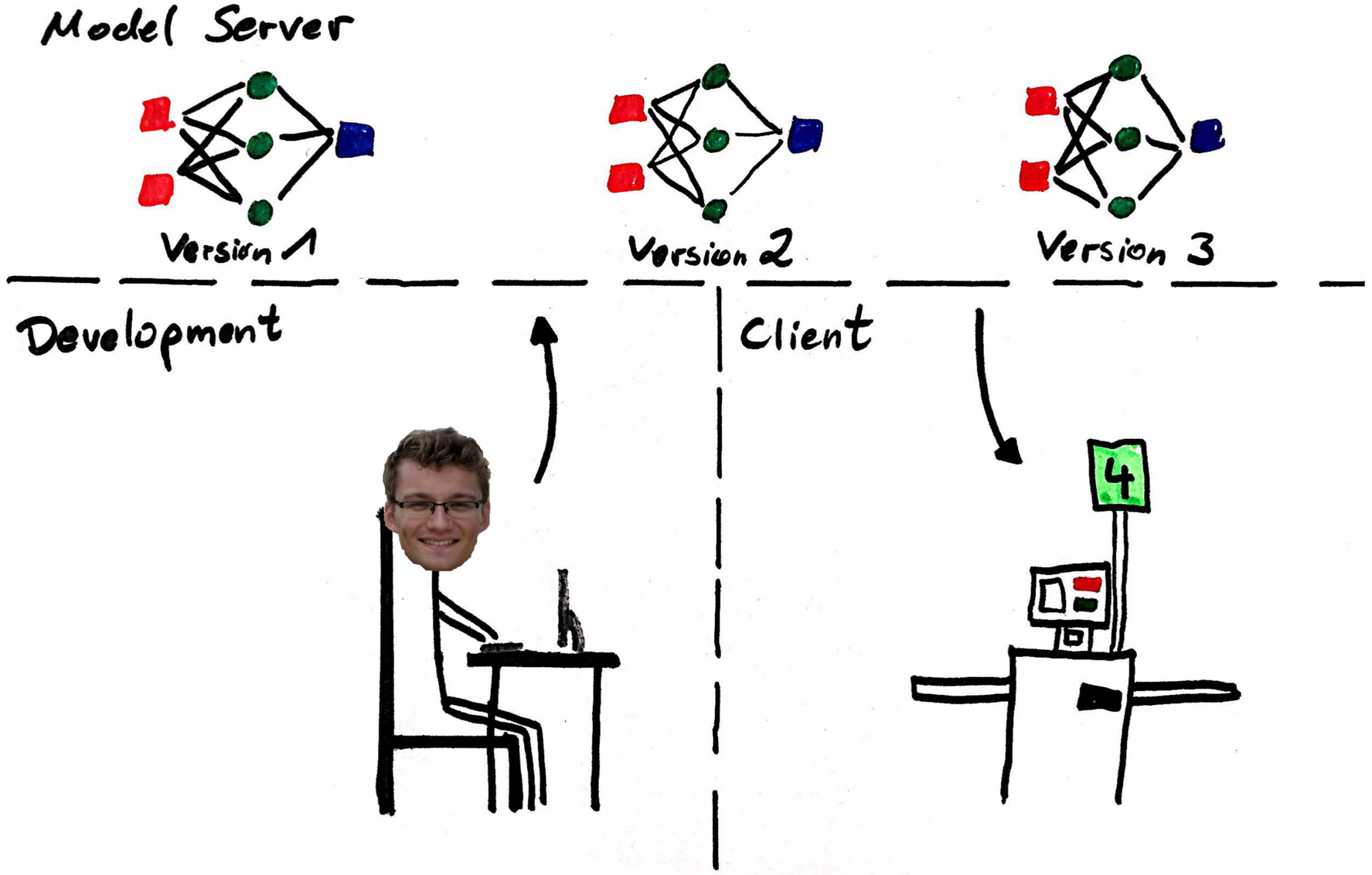
Let's get Practical: DVC



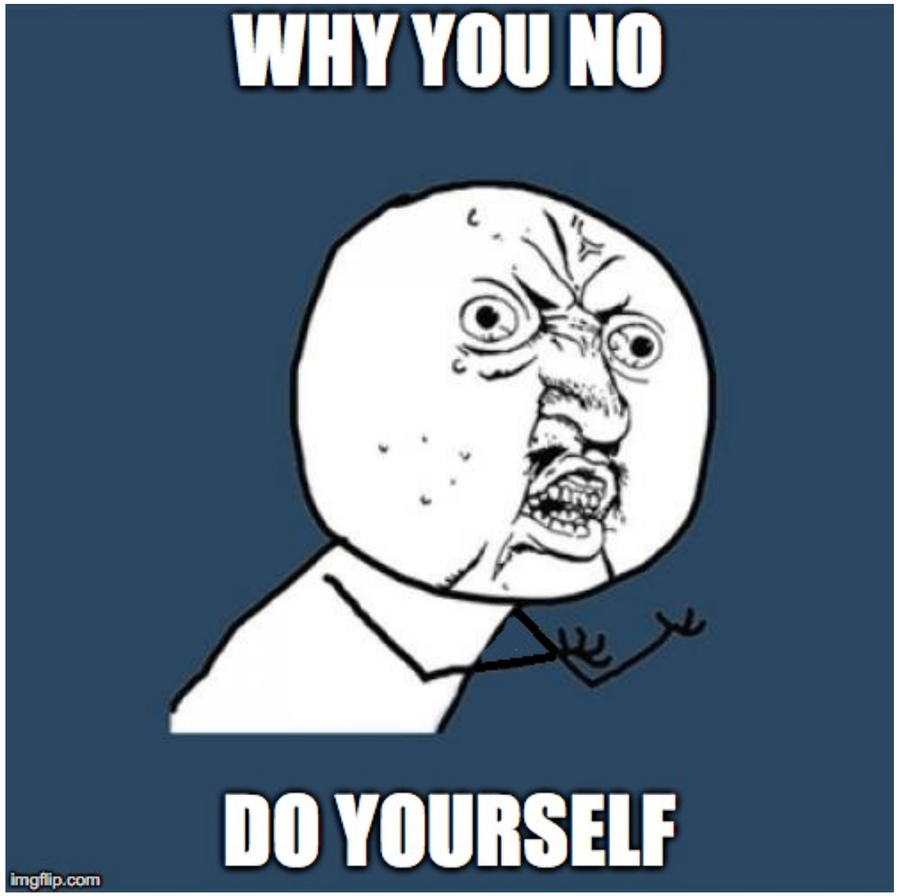
Model Serving with Tensorflow Serving

- Lightweight model server for Tensorflow graphs
- Useful for
  - Distributed model applications
  - Fast and continuous delivery
  - Model versioning

Tensorflow Serving



Let's get Practical: Tensorflow Serving



Preprocessing and the Deployment

- Preprocessing and model are tightly coupled
- How can the client "be sure" to do the right thing?

Possible solutions

- The client just knows
- Wrapper service around model and expose API
- Integrate preprocessing into the model

Preprocessing and the Deployment

```
import tensorflow as tf
from keras.layers import Lambda

def preprocess(x):
  resized = tf.image.resize_images(x, size=[100, 100])
  max_color = tf.constant(255, dtype=tf.float32)
  rescaled = tf.divide(resized, max_color)
  return rescaled

additional = Lambda(preprocess, output_shape=(100, 100, 3), name="preprocessing")
```

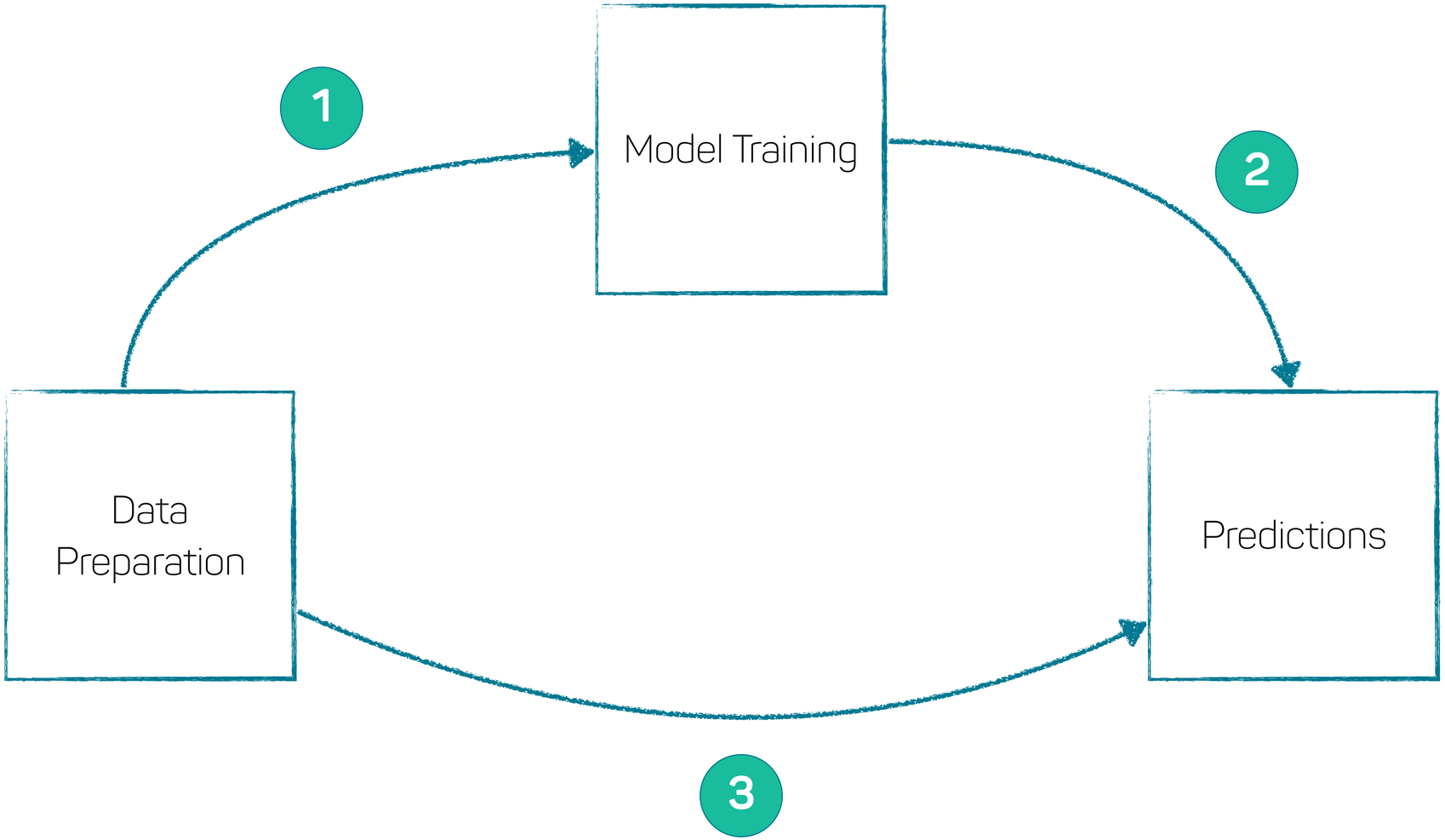
Preprocessing and the Deployment

```
from keras.models import Sequential
from keras.layers import InputLayer

prod_model = Sequential()
prod_model.add(InputLayer(input_shape=(None, None, 3)))
prod_model.add(additional)

for layer in model.layers:
    prod_model.add(layer)
```

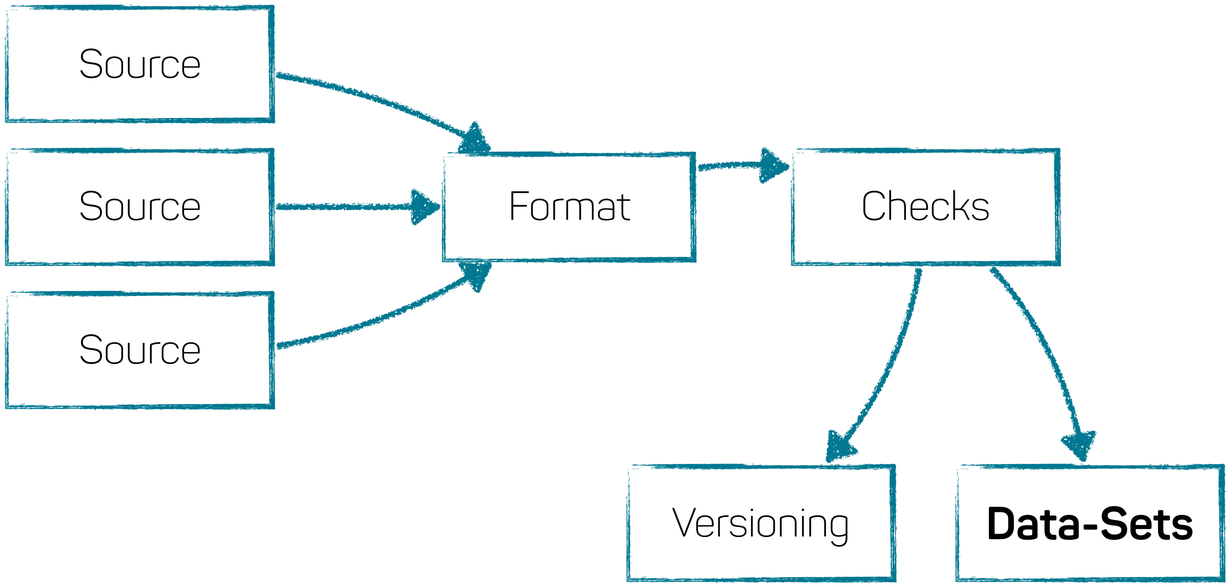
Workflow Management with Luigi/Airflow

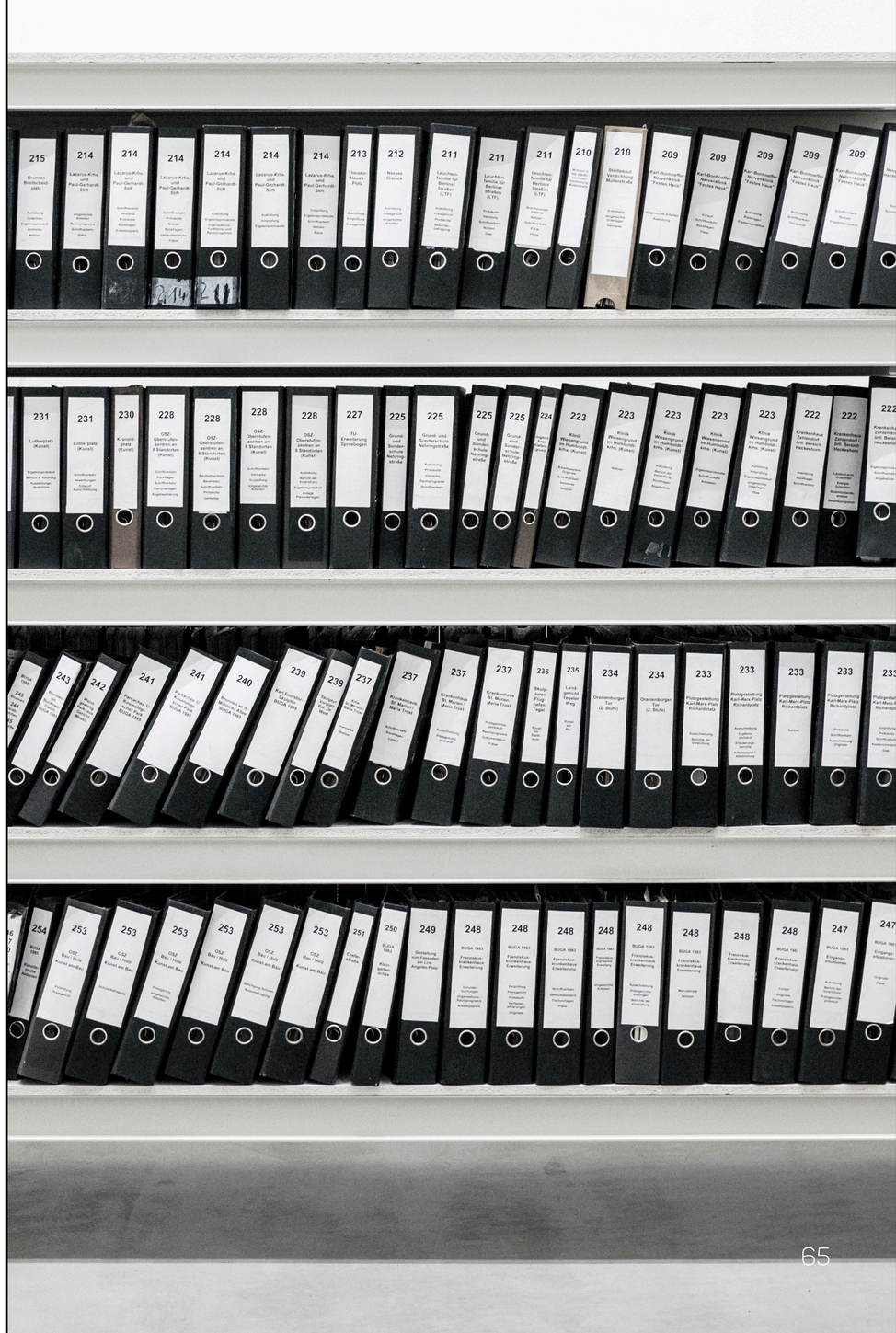
- DVC pipelines are useful for research & development, but intended for manual use
- What about more complex workflows?
- What about an automated & scheduled process?

Data Science Product Delivery

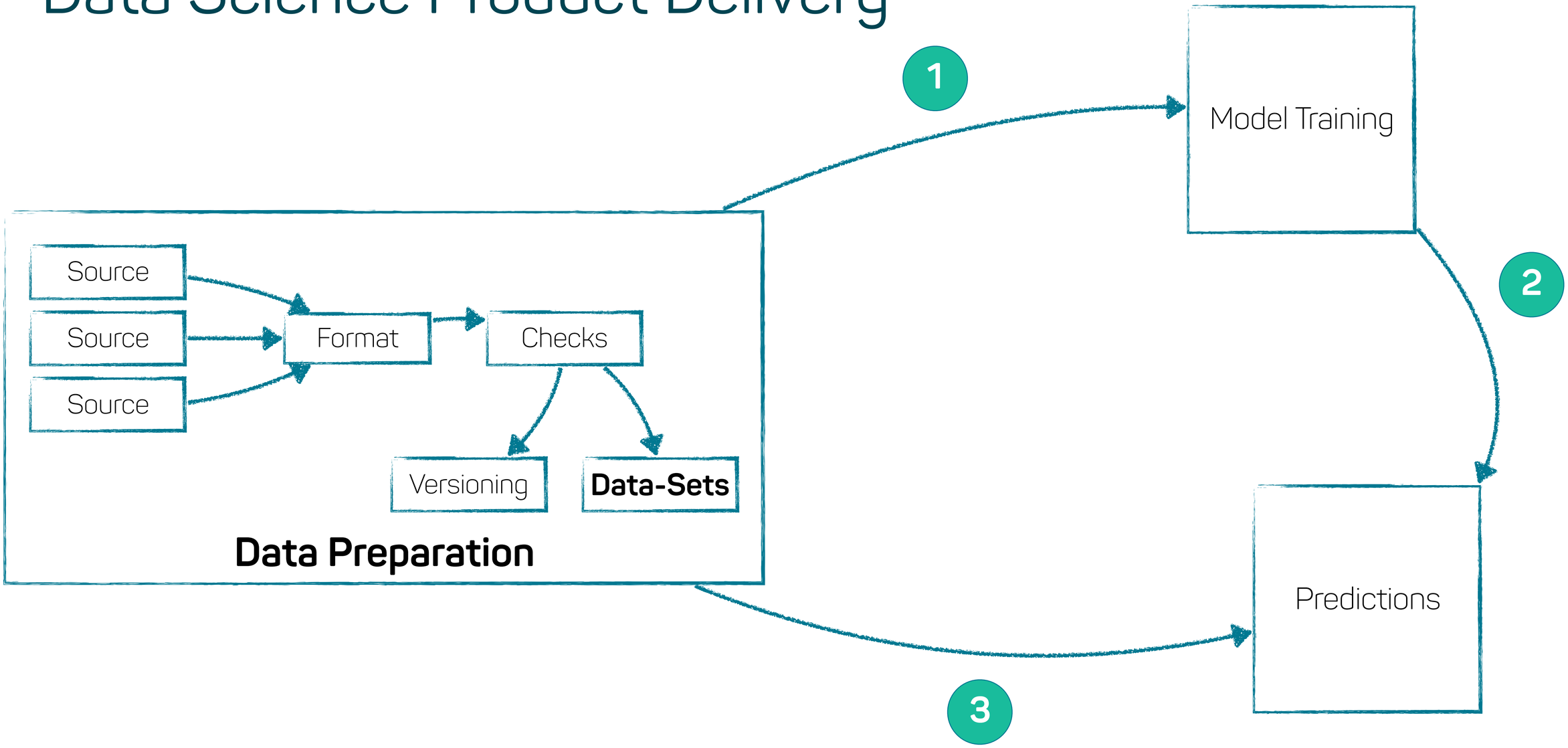


Data Preparation

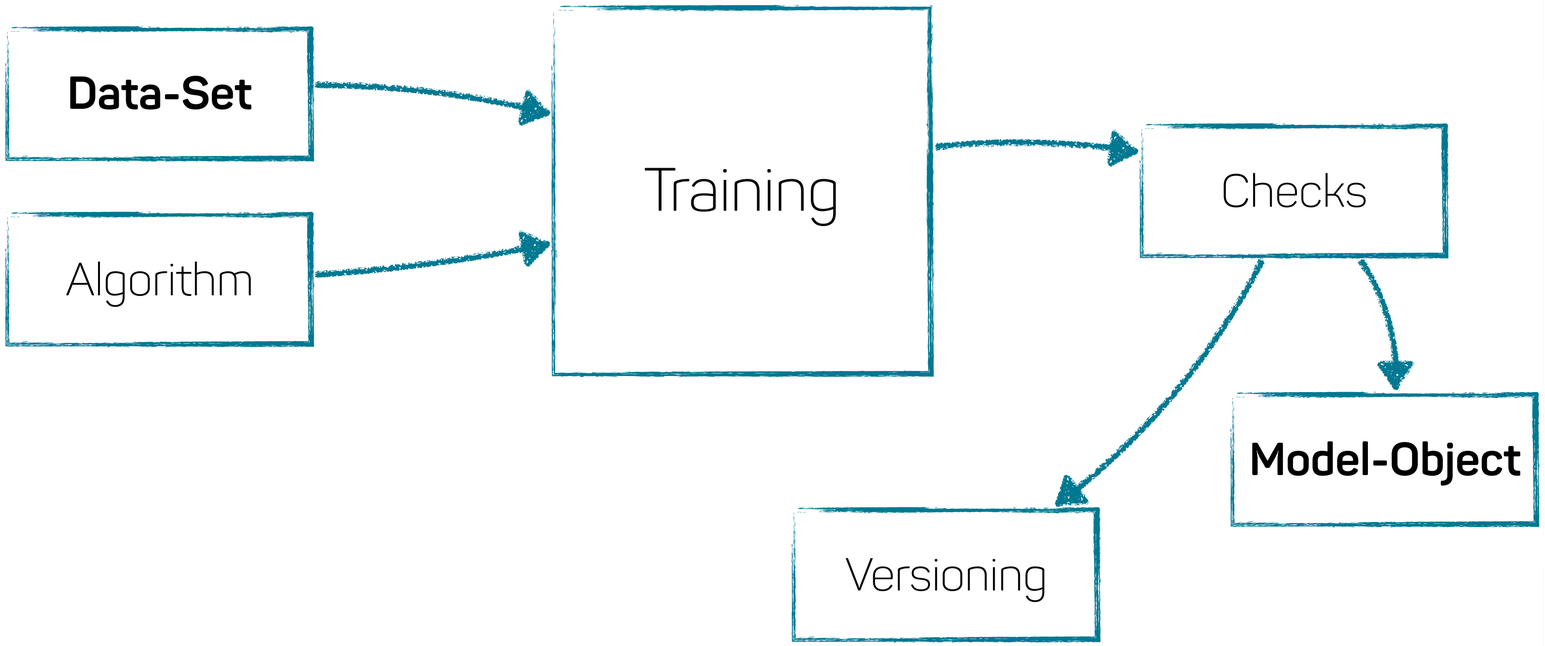


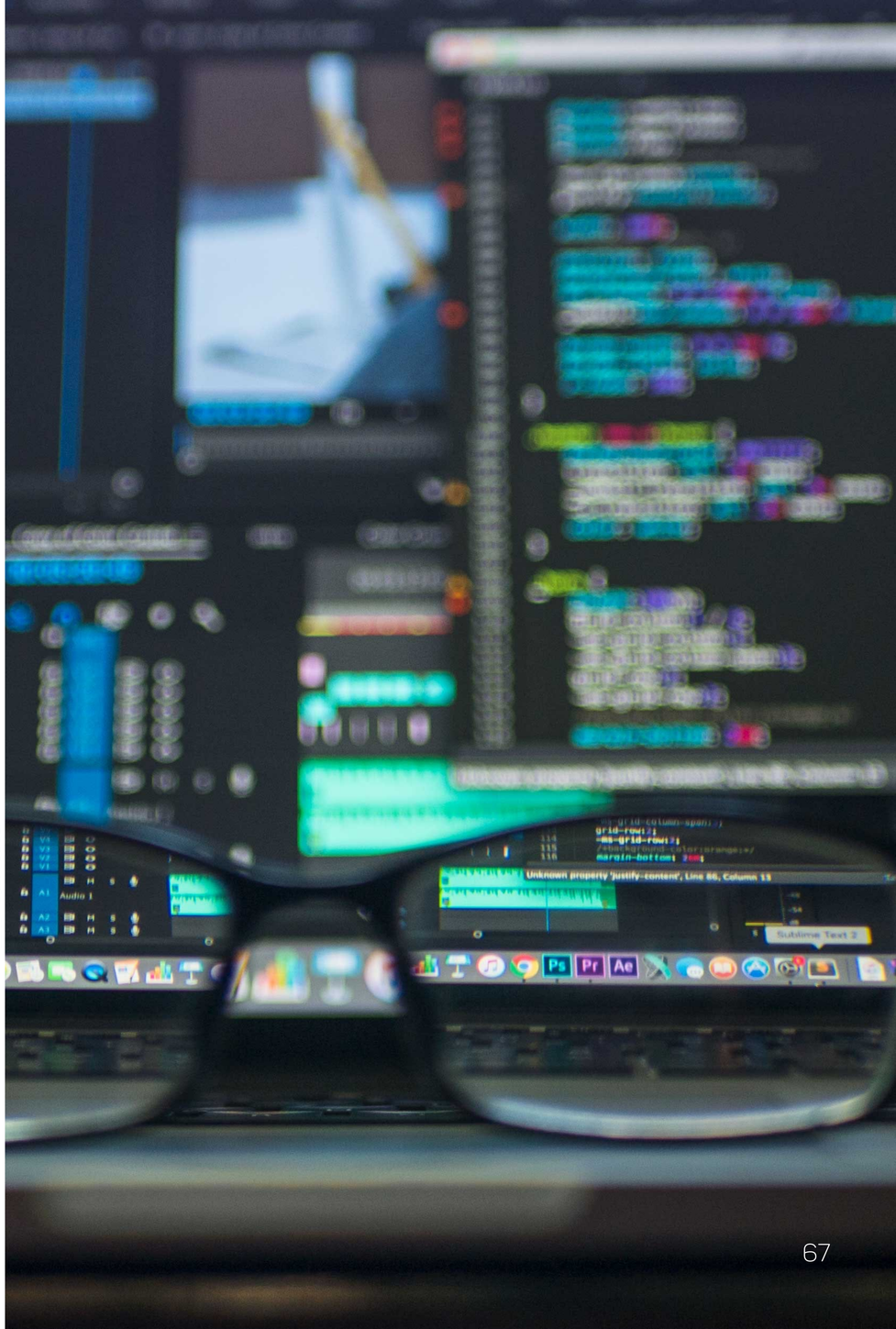


Data Science Product Delivery

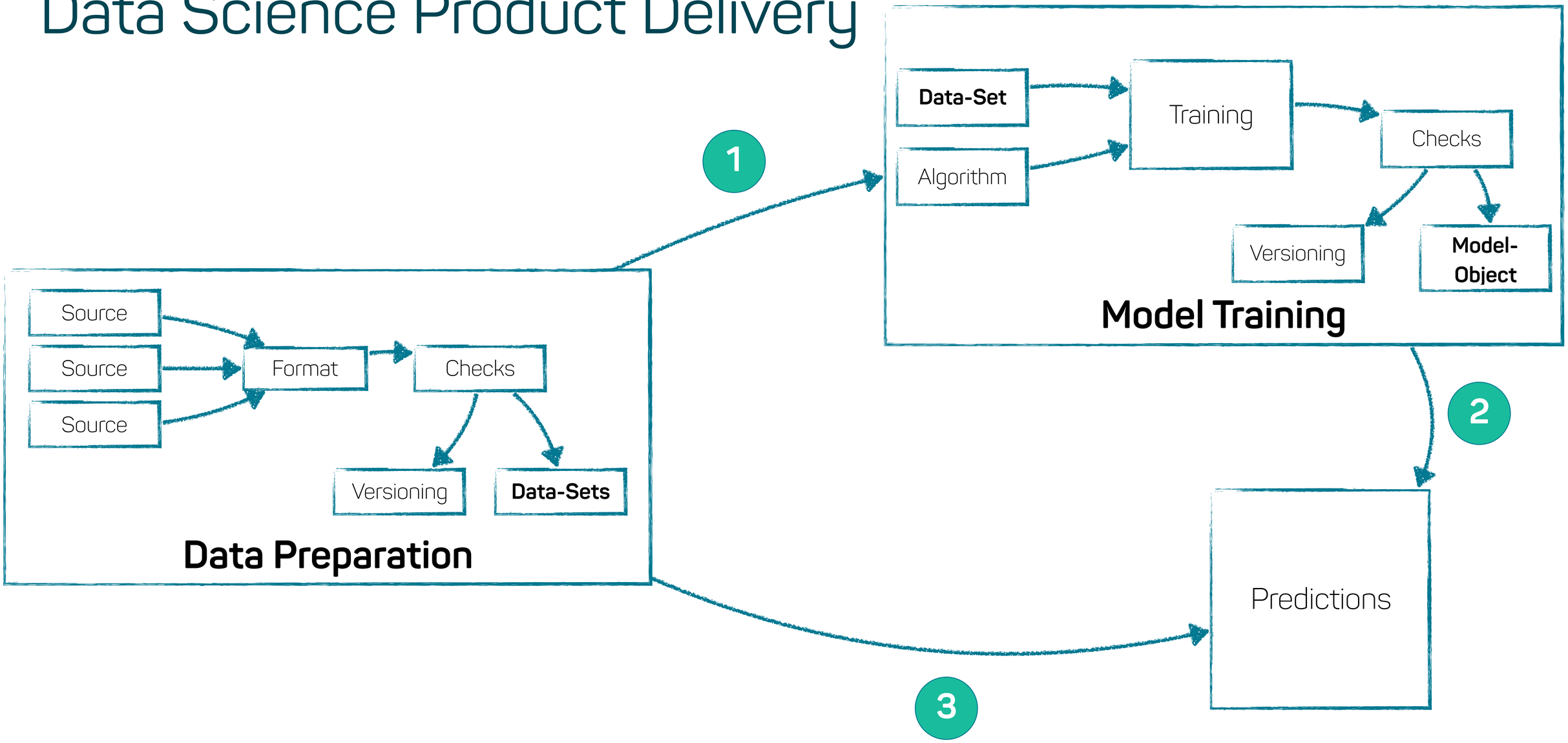


Model Training

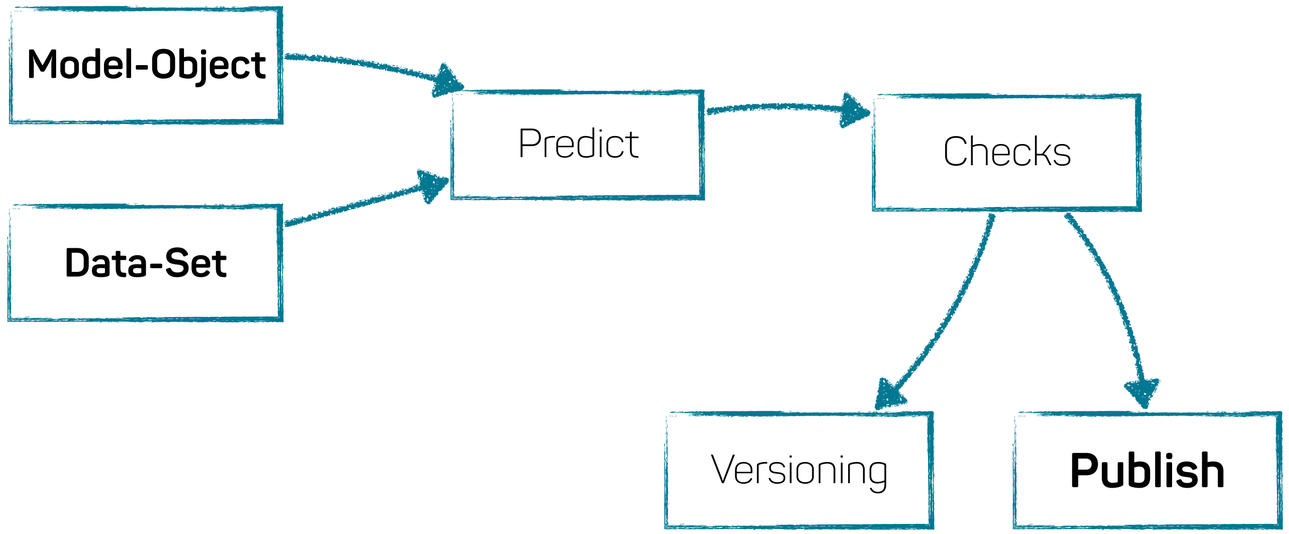




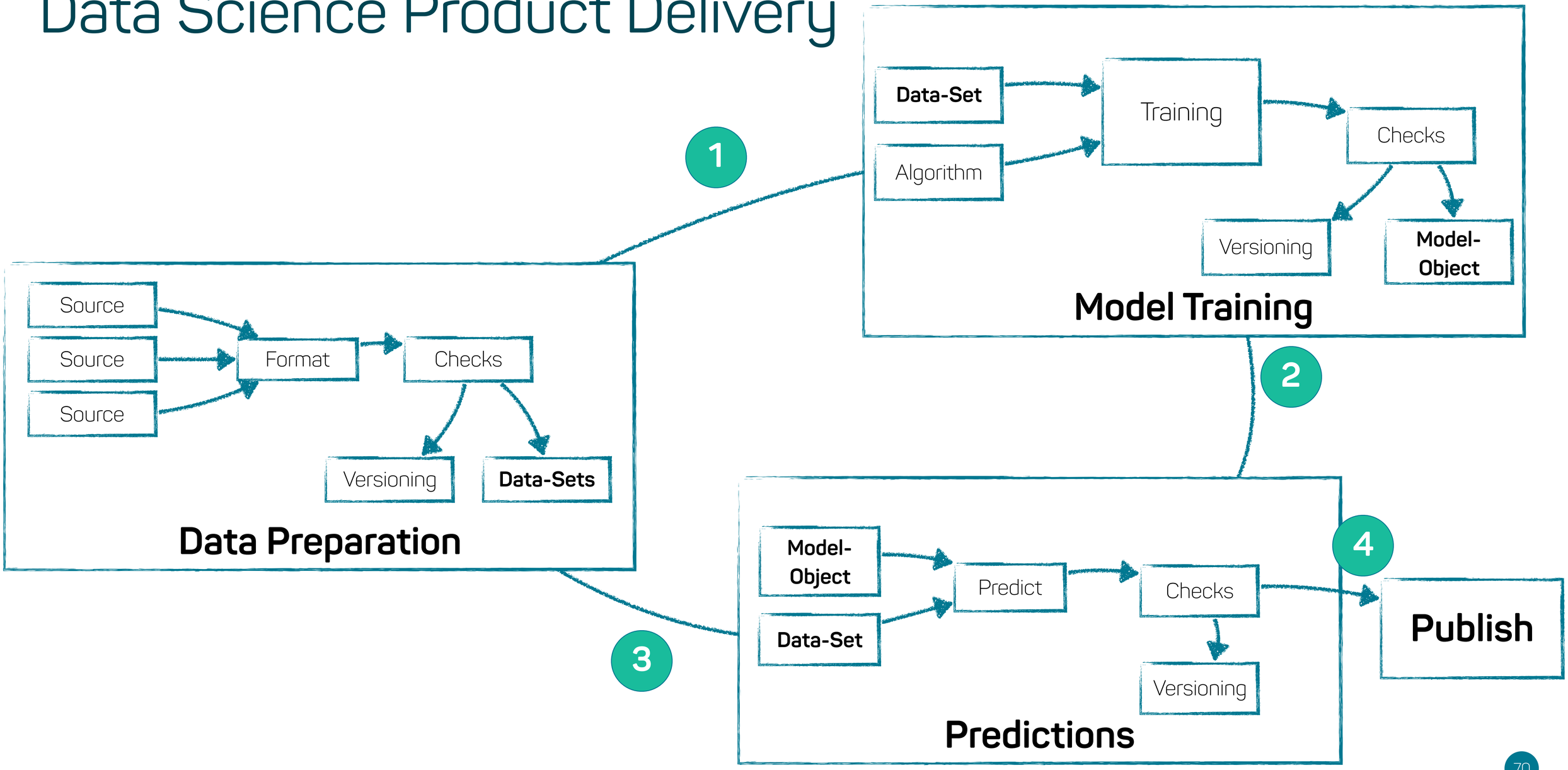
Data Science Product Delivery



Predictions



Data Science Product Delivery



Let's get Practical: Airflow

- Developed by AirBnB, now Apache Foundation

- **Write**, **schedule** and **monitor** workflows
- Write: Workflows are defined as Directed Acyclic Graphs, defined and written in Python
- Schedule: Provides own scheduler, allows for cron-style scheduling
- Monitor: Inspect and interact via webserver / UI