Deep Learning Topics

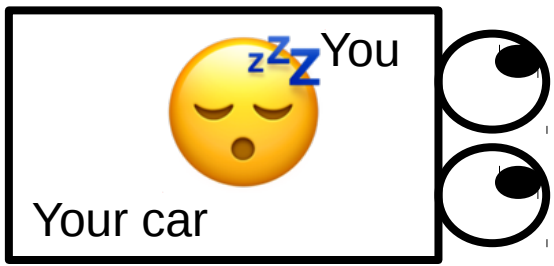
A literature overview

Yinan Yu
2019-10-08 MLFP

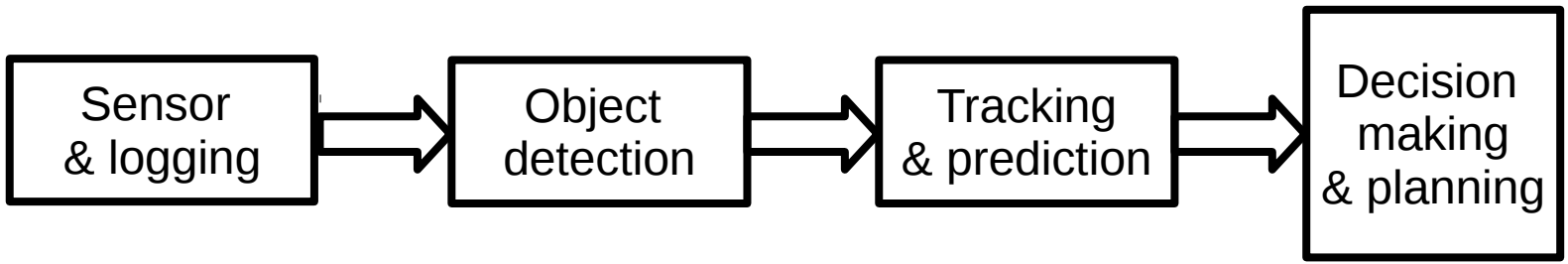
Deep learning!

WHAT FOR???

Self-driving cars

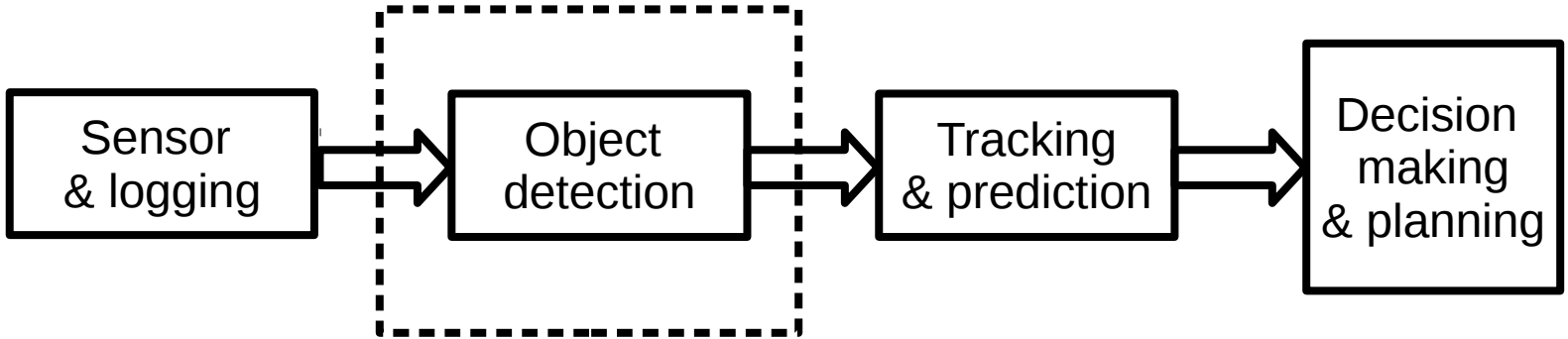


Self-driving cars



Data source: https://www.nuscenes.org/

Self-driving cars

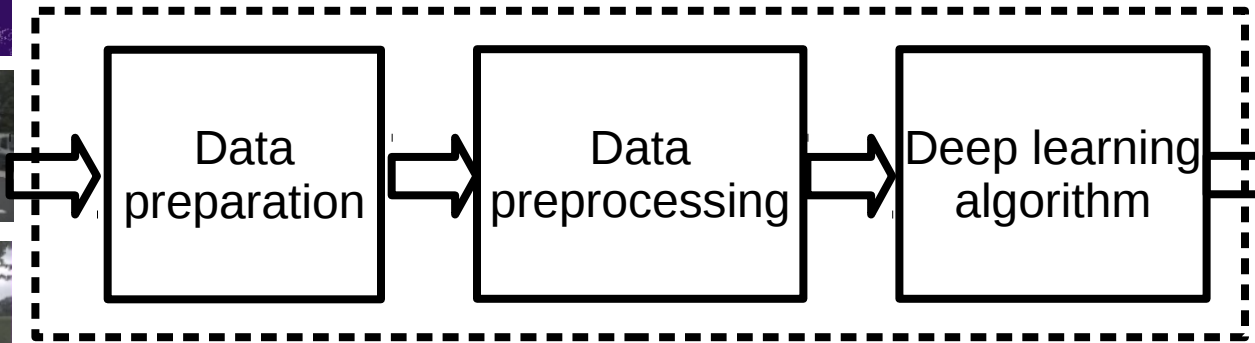


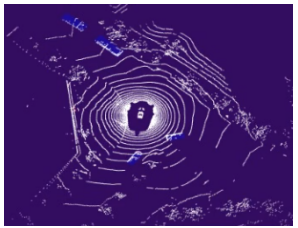
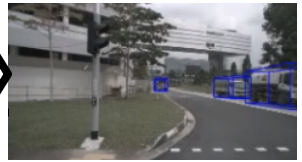
Data source: https://www.nuscenes.org/

Self-driving cars



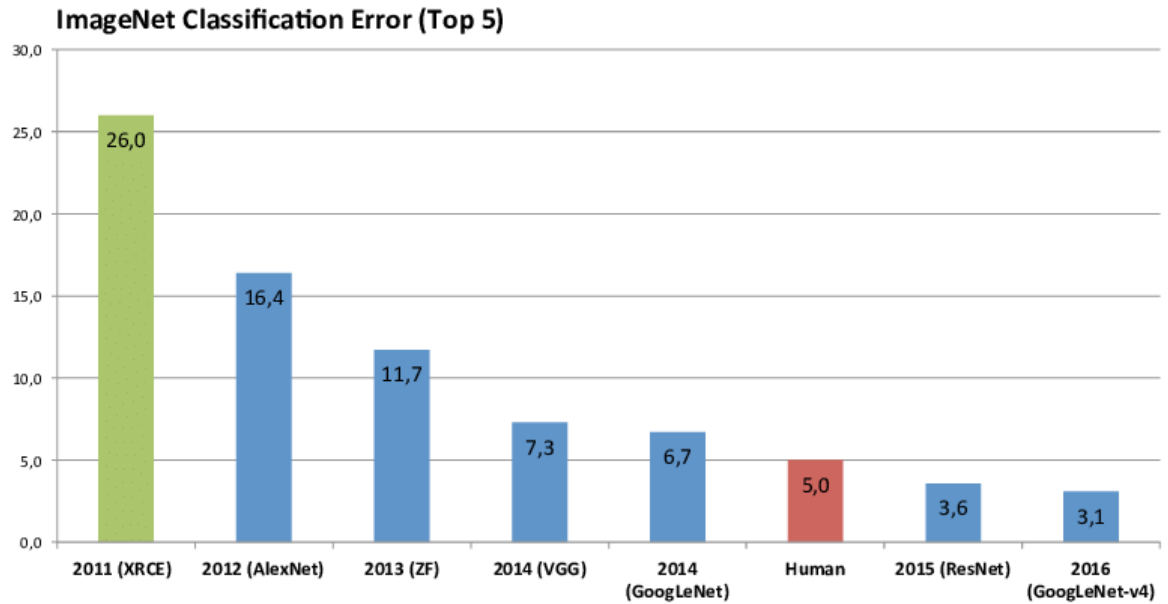

Deep learning




Data source: https://www.nuscenes.org/

Computer vision (ImageNet) challenge



https://devopedia.org/imagenet

Purpose

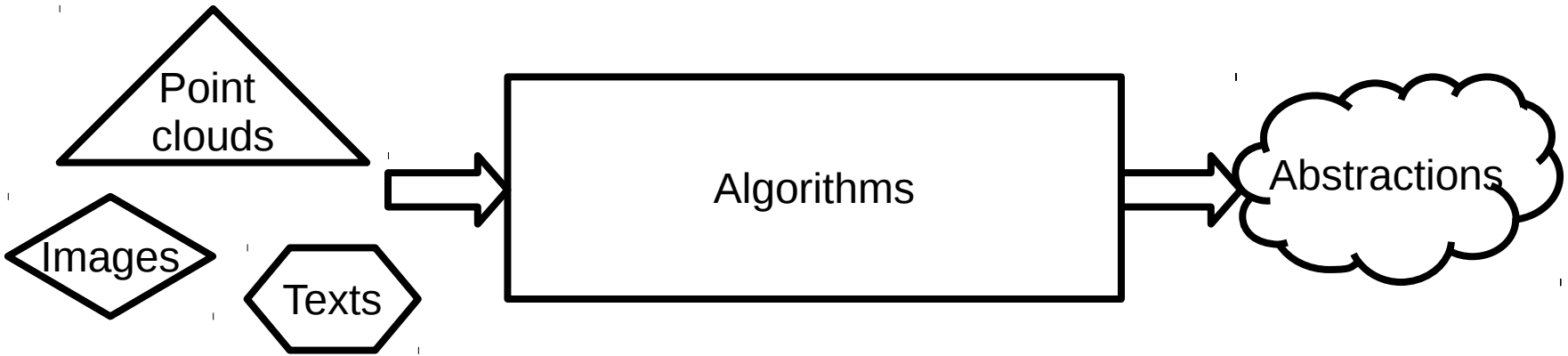
- What are the different building blocks?
- What are the “variations”?
- How are they evaluated?
- Interesting subjects for future MLFP talks?

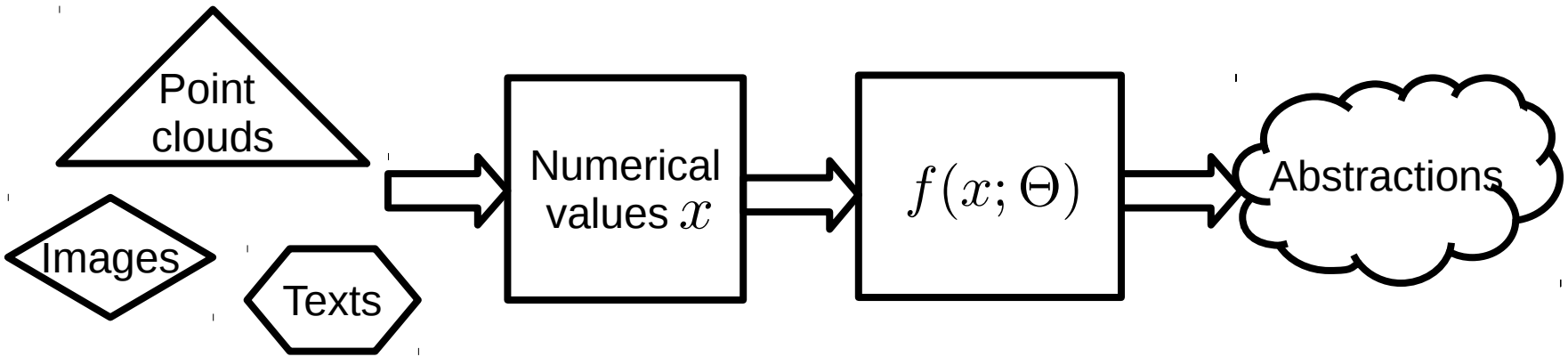
Outline

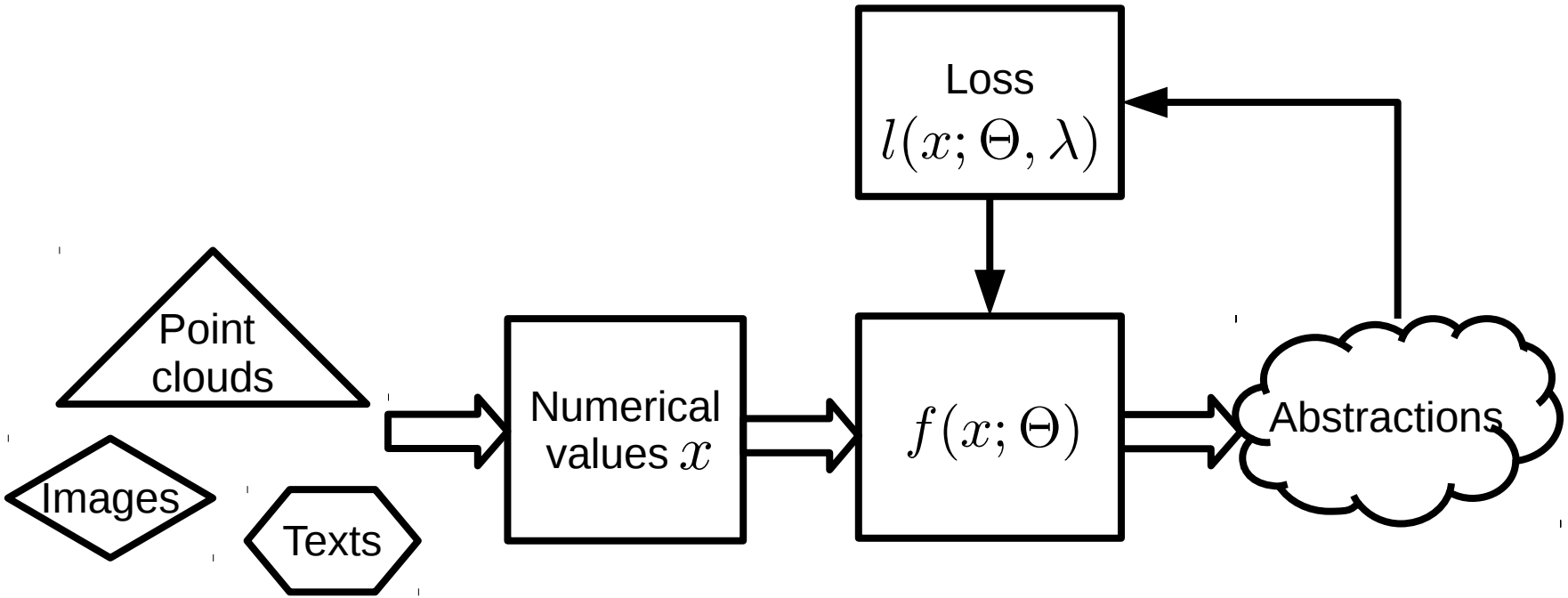
- Overview
  - Deep learning pipeline
  - Evaluation: accuracy, complexity, training time, inference time, memory footprint
- Components
  - Data modeling
  - Architecture
  - Loss, optimization
  - Abstraction and annotation
  - Infrastructure

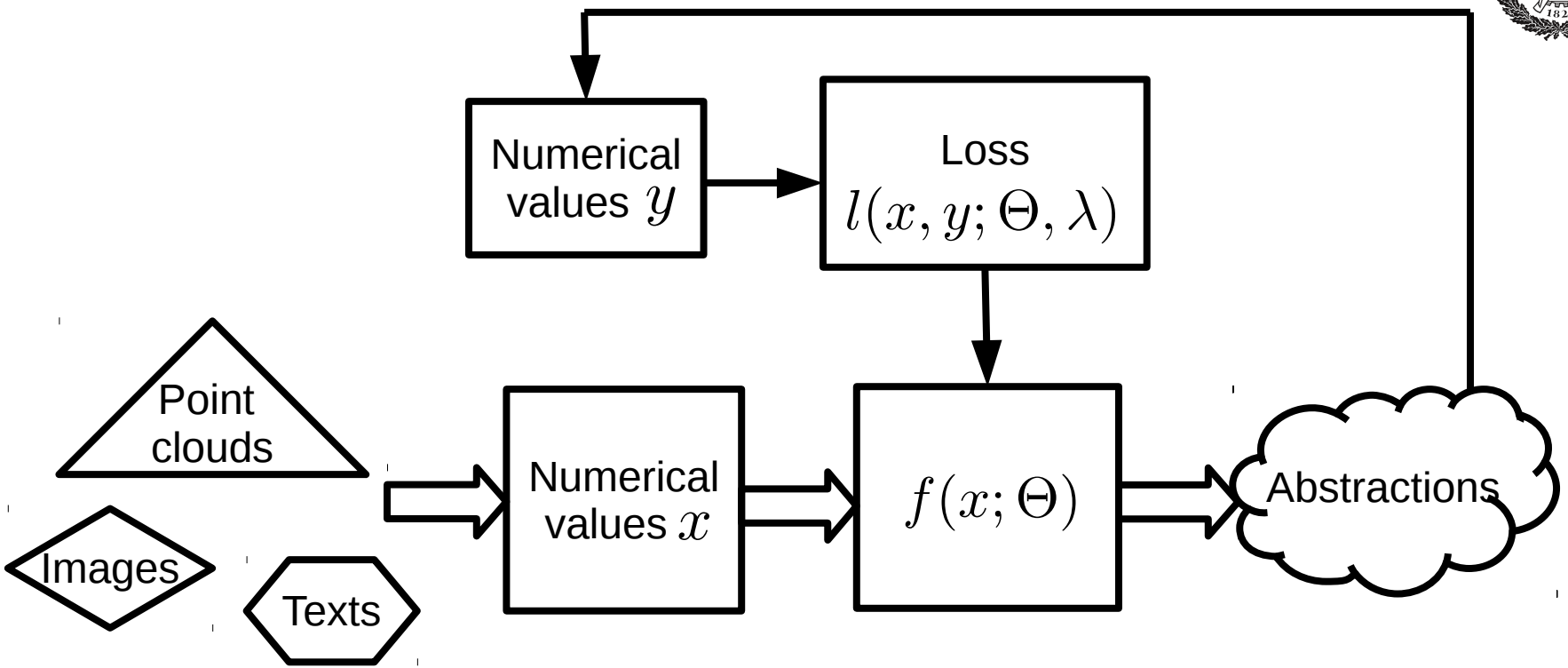
Overview

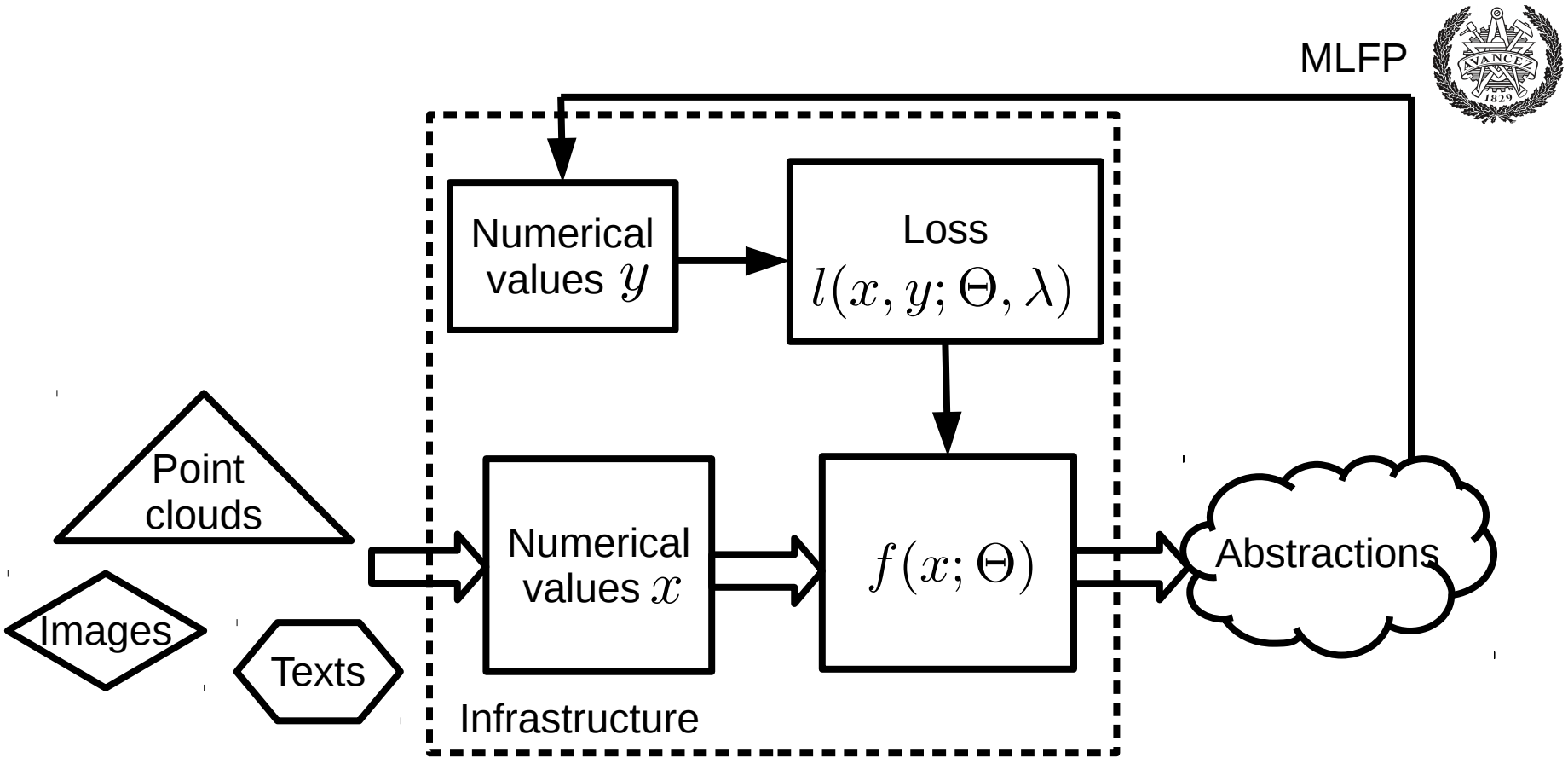
Deep learning: data driven approach




Point clouds
Images
Texts
Numerical values x
$f(x;\Theta)$
Abstractions


Loss
$l(x; \Theta, \lambda)$
Point clouds
Images
Texts
Numerical values x
$f(x; \Theta)$
Abstractions


Numerical values y
Loss $l(x, y; \Theta, \lambda)$
Point clouds
Images
Texts
Numerical values x
$f(x; \Theta)$
Abstractions

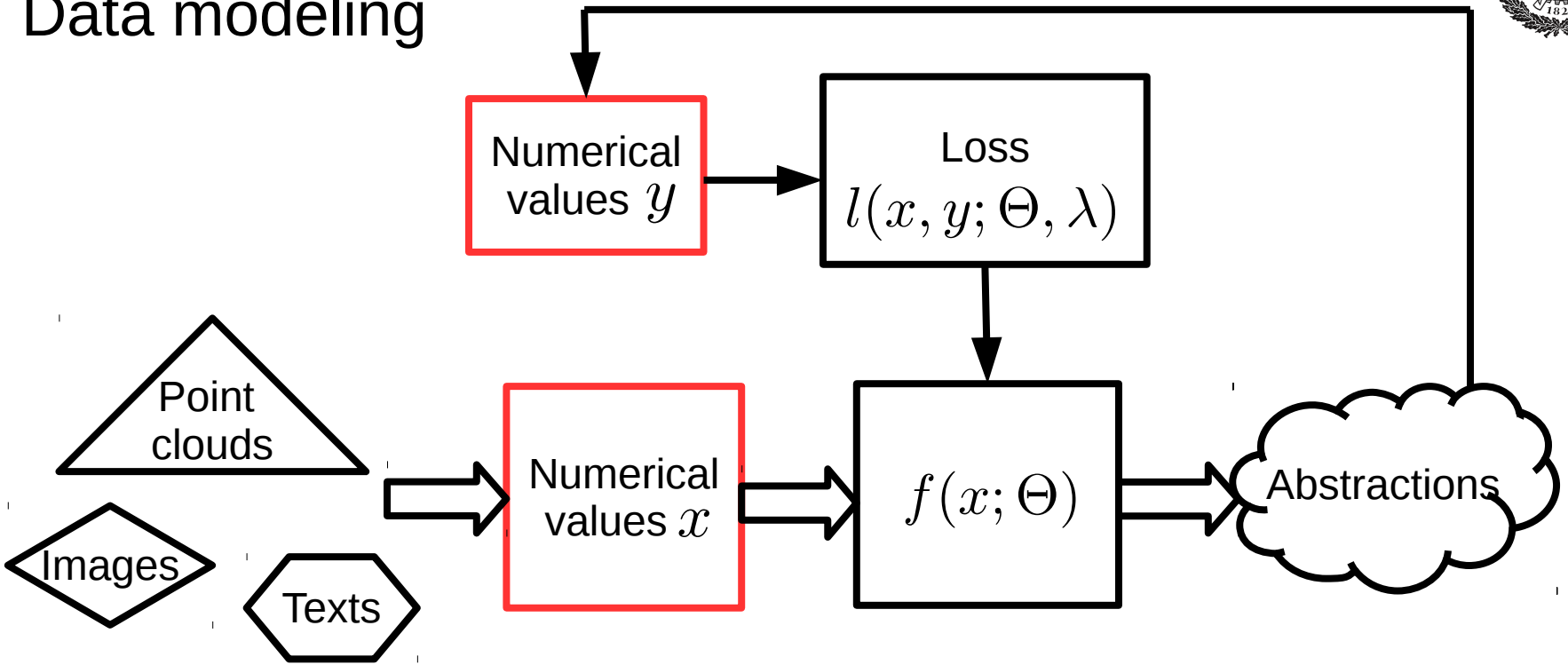

MLFP
Numerical values y
Loss $l(x, y; \Theta, \lambda)$
Point clouds
Images
Texts
Numerical values x
$f(x; \Theta)$
Abstractions
Infrastructure

Evaluation

- Accuracy
  - Quantitative error: classification error, regression error
  - Qualitative error: visualization, human interpretation
  - Performance in the “big picture”
- Complexity measures
  - Number of trainable parameters
  - Model specific complexity
- Training time
  - Theoretical time complexity (number of training data, dimensionality, network architecture, error tolerance)
  - Measured training time for specific infrastructures
- Inference time
  - FLOPs (MACC)
  - Measured inference time for specific infrastructures/target hardware
- Memory footprint

Components

Data modeling

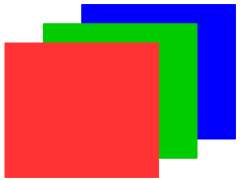


Data modeling

- Goal: transform data (texts, images, point clouds, etc) and their annotations into “meaningful” numerical values
- Challenges:
  - Numerical interpretation from arbitrary data format
  - Meaningful representation of annotations
  - Variable size
  - Unclear contribution to the machine learning task

Data modeling

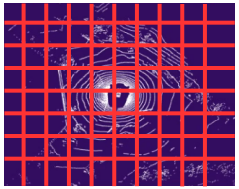
- Images:



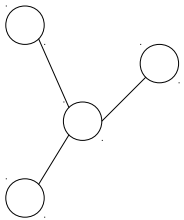
- Text:
  - [“cat”, “cute”, “be”, “I”, “a”, “have”]
  - “Cats are cute.” → [1, 1, 1, 0, 0, 0]
  - “I have a cute cat.” → [1, 1, 0, 1, 1, 1]

Data modeling

- Point cloud:



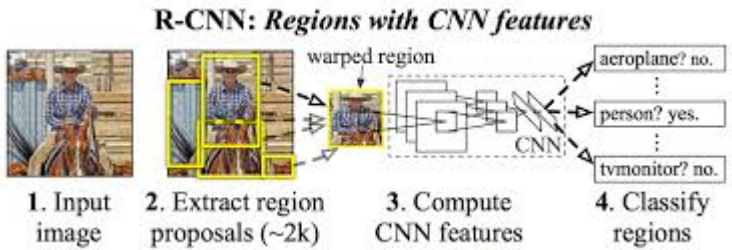
- Drug design:

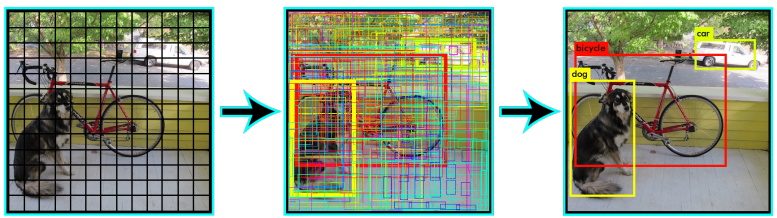


TTACGAAGGA...

Data annotation modeling

- Annotation: “ground truth” of the abstractions
- Use case specific and nontrivial…
- Example (bounding boxes): location + size + class label



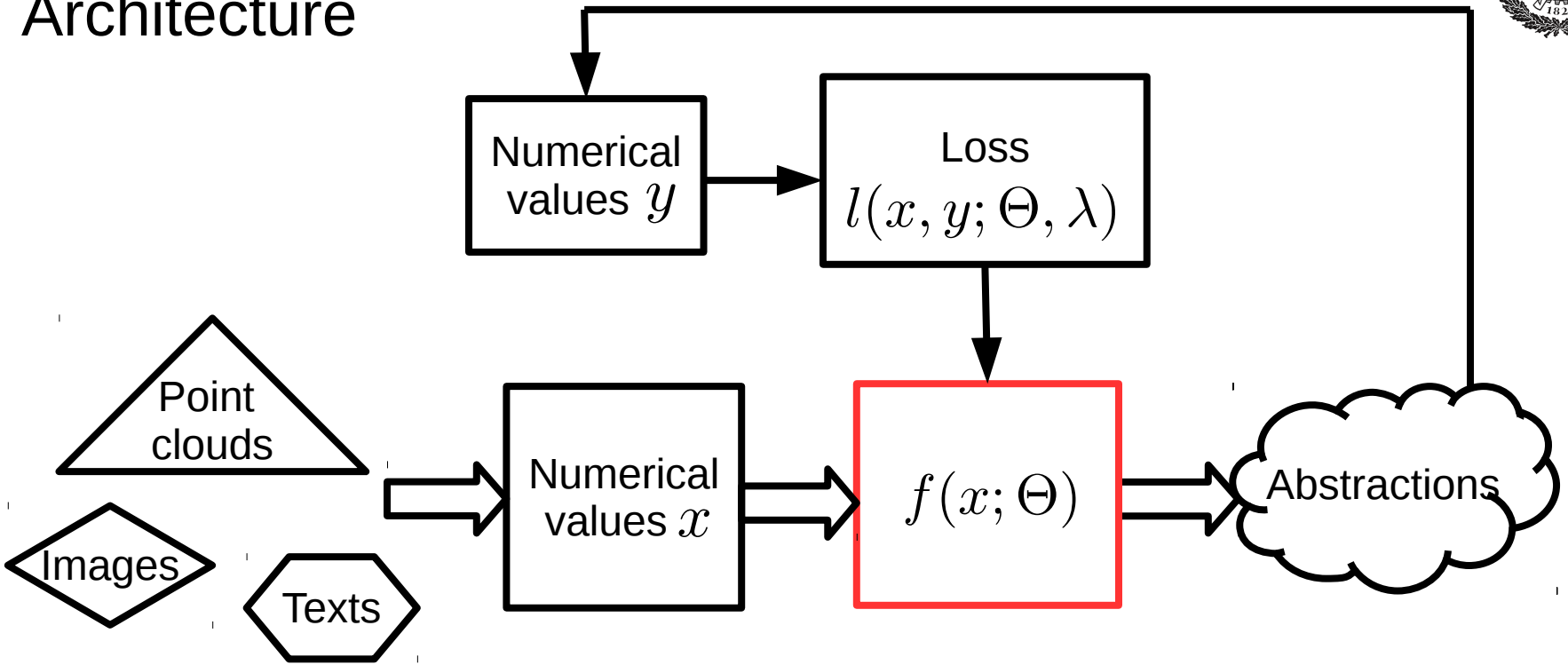


R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. CVPR 2014
Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi, You Only Look Once:unified, real-time object detection, CVPR 2016

Data modeling

- Application specific
- Domain knowledge
- If you want to try out deep learning for your task
  - Be creative with data modeling
  - Apply standard network architectures

Architecture



Architecture

- Types
- Models
- Topology and wiring
- Efficiency improvement

Architecture: types

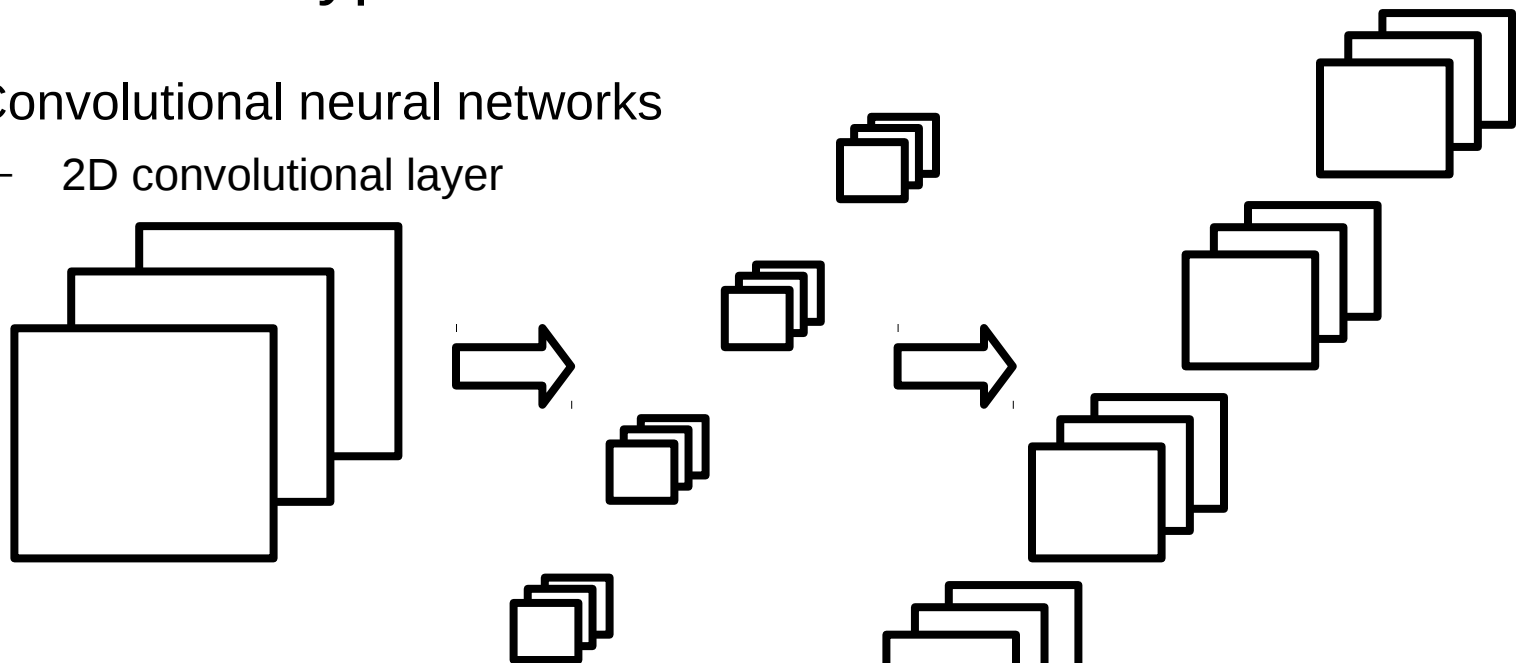
- Different structures
- Handling different inputs and producing different outputs

Architecture: types

- Convolutional neural networks
  - 2D convolutional layers
  - 3D convolutional layers
  - Transposed layers
  - Depth-wise separable conlutional layers
  - More convolutional layers: https://github.com/vdumoulin/conv_arithmetic
  - Pooling layers
  - Activation functions
- Artificial neural networks
- Recurrent neural networks/LSTM/neural Turing machines
- Recursive neural networks
- Graph neural networks
- Spiking neural networks

Architecture: types

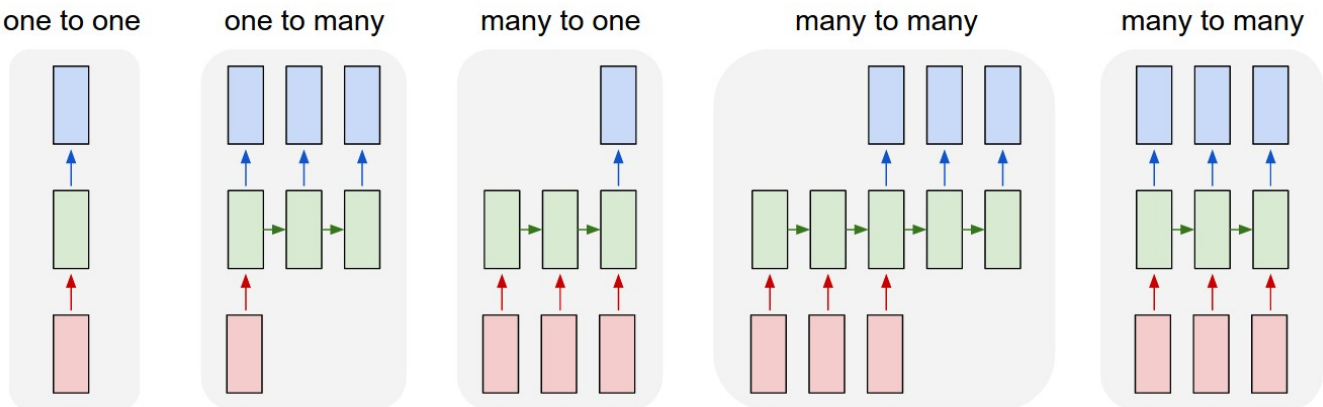
- Convolutional neural networks
  - 2D convolutional layer



$$dim = [c_{in}, H, W, c_{out}]$$

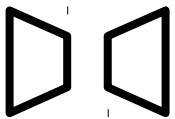
Architecture: types

- Recurrent neural networks



https://karpathy.github.io/2015/05/21/rnn-effectiveness/

Architecture: models

- Generator/discriminator model (GAN)
- Data embedding (metric learning)
- Encoder/decoder ()
- Attention (mainly in recurrent networks, weighted sum)

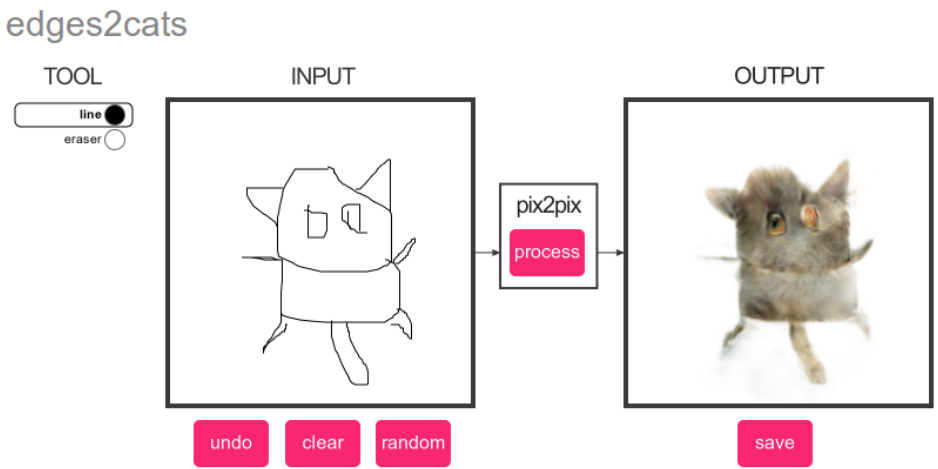
Architecture: models

- Generator/discriminator model (GAN)
- https://affinelayer.com/pixsrv/

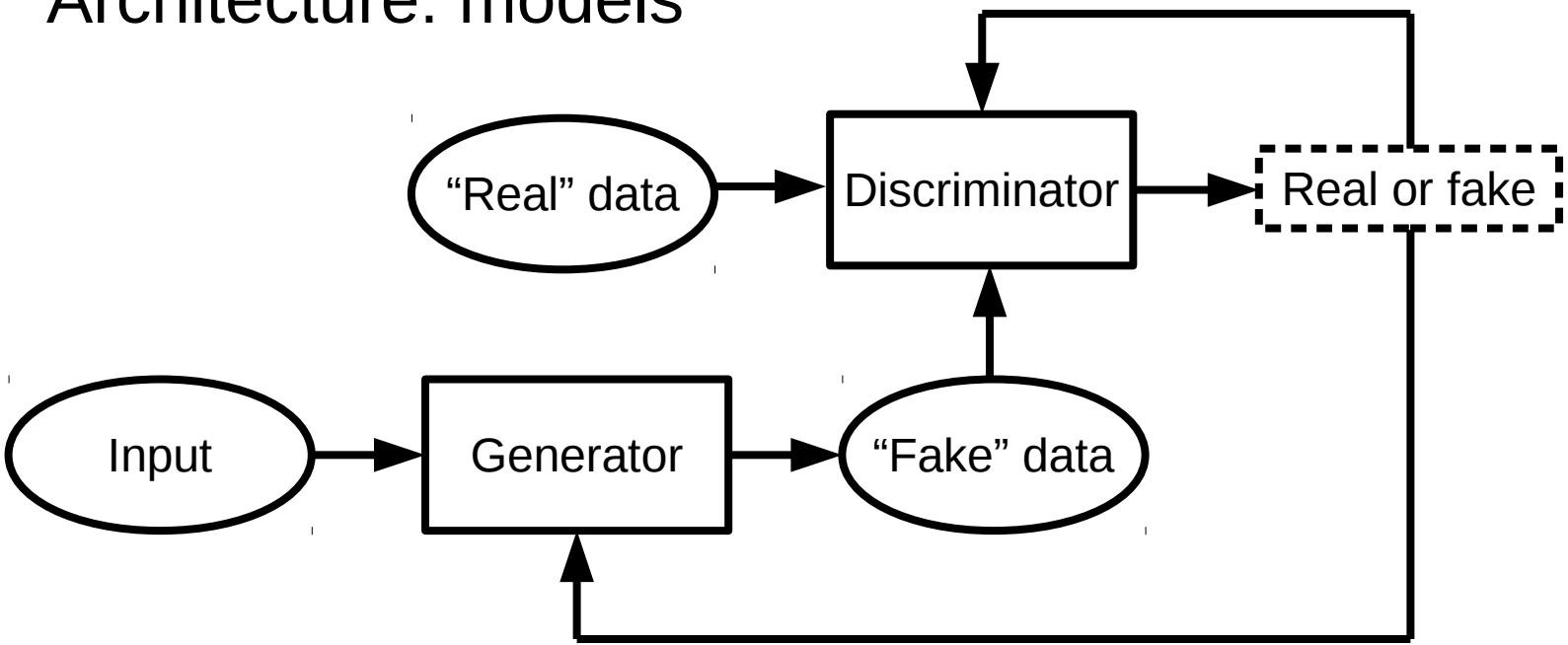
Ian J. Goodfellow, Jonathon Shlens, Christian Szegedy, Explaining and harnessing adversarial examples, ICLR 2014.

Architecture: models

- Generator/discriminator model (GAN)
- https://affinelayer.com/pixsrv/



Architecture: models



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

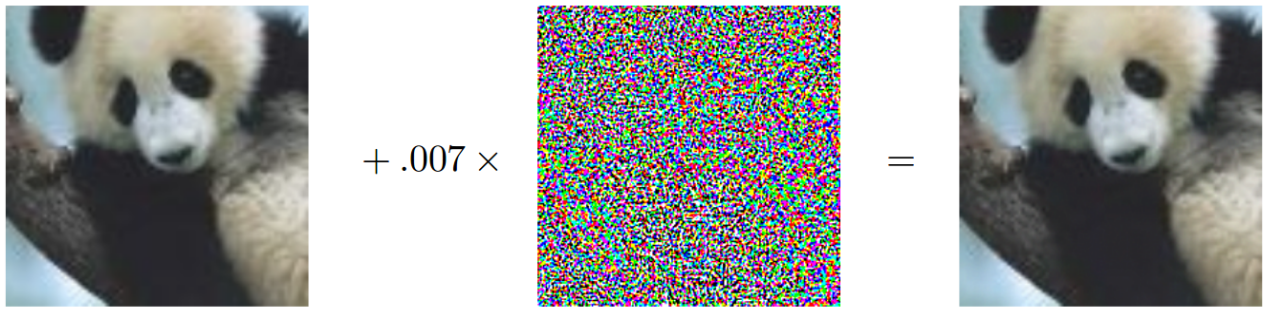
Goodfellow Ian, Pouget-Abadie Jean, Mirza Mehdi, Xu, Bing, Warde-Farley David, Ozair Sherjil, Courville Aaron, Bengio Yoshua, Generative adversarial networks, NIPS 2014.

Architecture: models

- Generator/discriminator model (GAN)
  - “Cool” applications, “impressive” results
  - Many failure cases

Tero Karras, Samuli Laine, Timo Aila, A style-based generator architecture for generative adversarial networks, CVPR 2019
https://github.com/NVlabs/ffhq-dataset

Adversarial attacks



Ian J. Goodfellow, Jonathon Shlens, Christian Szegedy, Explaining and harnessing adversarial examples, ICLR 2014.

Adversarial training

- Deep neural nets are “too linear” → Underfitting
- Adversarial attack construction
- Defense strategies

Adversarial training

	Clean testing data	Adversarial testing data
Clean training data	★★	
Adversarial training data	★★★	★

Architecture: models

- Deep embedding (metric learning)
  - Purpose:
    - find a good metric space: “similar” items should be close, “different” items should be far
  - Definition:
    - $f : \mathbb{R}^N \rightarrow \mathbb{R}^d$
    - f is a differentiable deep network
  - Different loss:
    - Margin Loss
    - Contrastive Loss
    - Triplet Loss
    - Hinge loss

Architecture: topology and wiring

- Evaluation: examples

https://www.cs.toronto.edu/~kriz/cifar.html
http://www.image-net.org

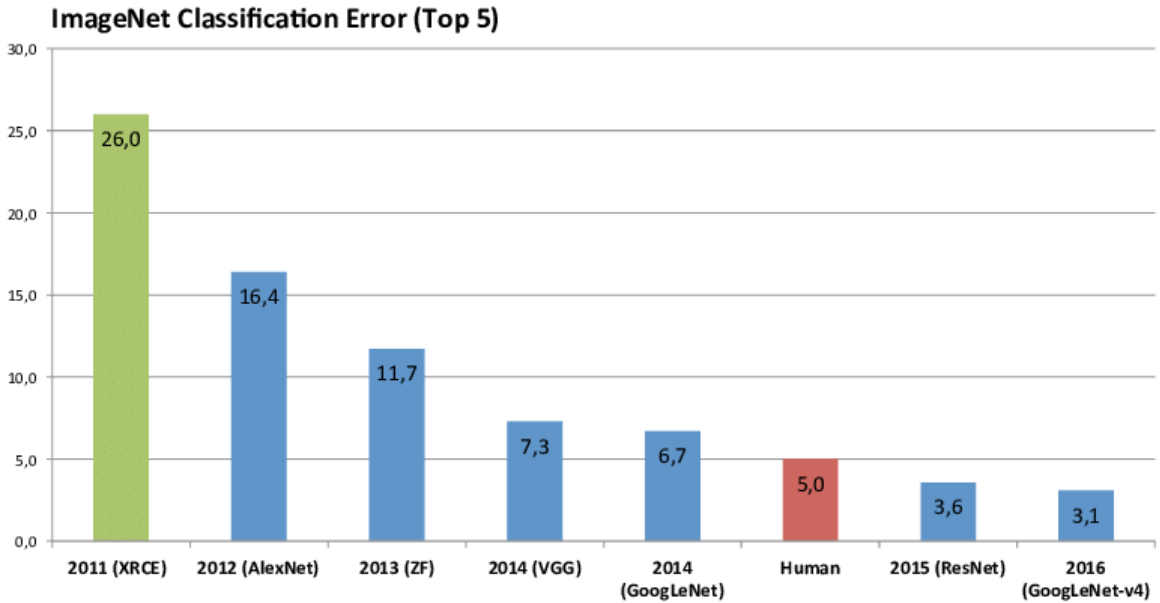
Architecture: topology and wiring

- CIFAR-10
  - 10 classes
  - 60,000 images
  - Image size: 32x32 pixels
  - Accuracy: classification rate
  - Typically takes < an hour to train
- ImageNet
  - 1000 classes
  - 1,500,000 images
  - Image size: approximately 200x200 pixels
  - Accuracy: top-1 & top-5
  - Typically takes days to train

https://www.cs.toronto.edu/~kriz/cifar.html
http://www.image-net.org

Architecture: topology and wiring

- Evaluation result: ImageNet challenge

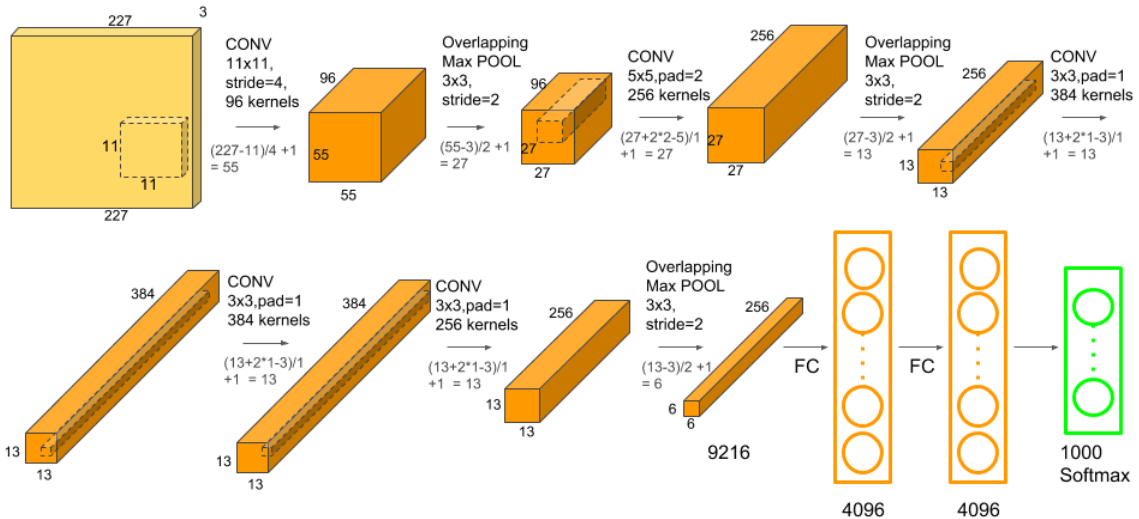


https://devopedia.org/imagenet

Architecture: topology and wiring

- Examples:

  AlexNet

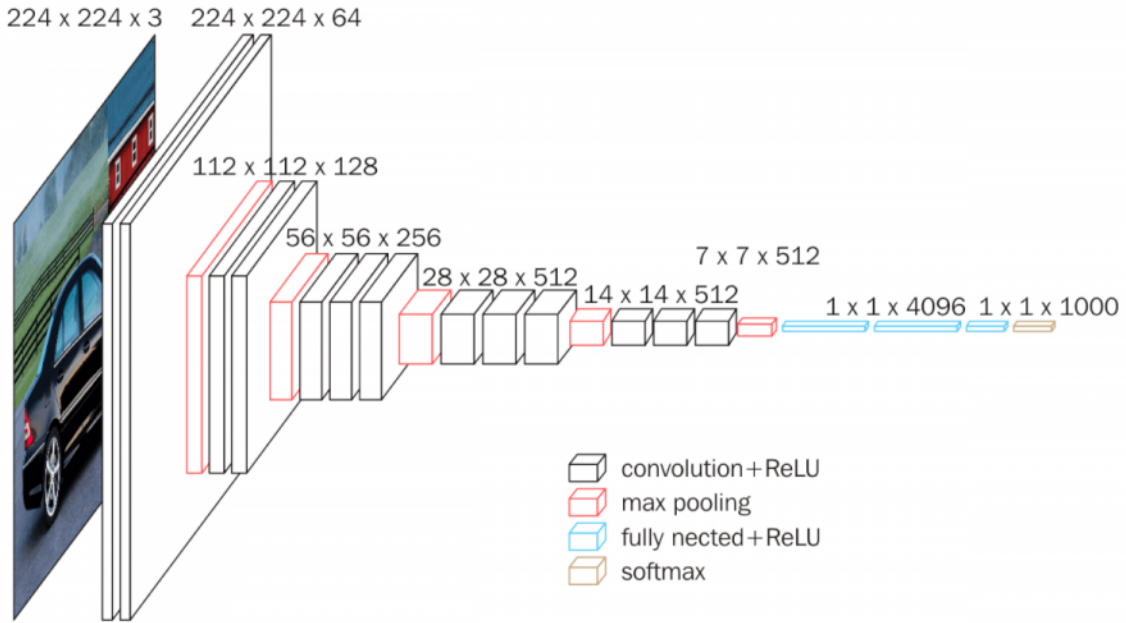


Alex Krizhevsky, ImageNet classification with deep convolutional neural networks, NIPS 2012
https://neurohive.io/en/popular-networks/alexnet-imagenet-classification-with-deep-convolutional-neural-networks/

Architecture: topology and wiring

- Examples:

  VGG

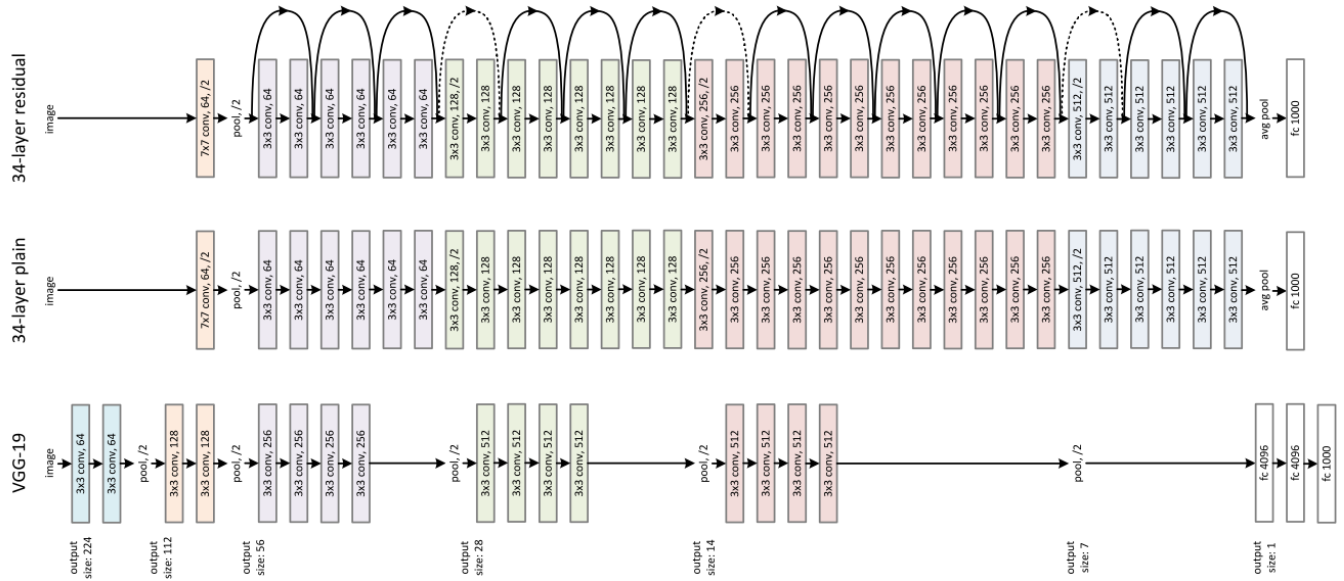


Karen Simonyan, Andrew Zisserman, Very deep convolutional networks for large-scale image recognition, ICLR 2015
https://neurohive.io/en/popular-networks/vgg16/

Architecture: topology and wiring

- Examples:

  ResNet

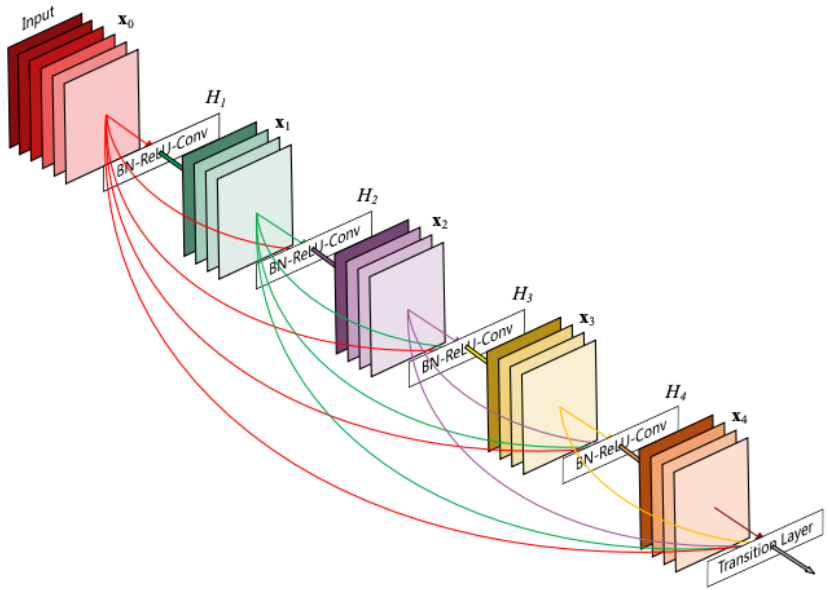


Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, Deep residual learning for image recognition, CVPR 2016

Architecture: topology and wiring

- Examples:

  DenseNet



Gao Huang, Zhuang Liu, Laurens van der Maaten, Kilian Q. Weinberger, Densely connected convolutional networks, CVPR 2017

Architecture: topology and wiring

- Examples:

  Inception/Xception

Figure 1: A canonical Inception module (Inception V3).

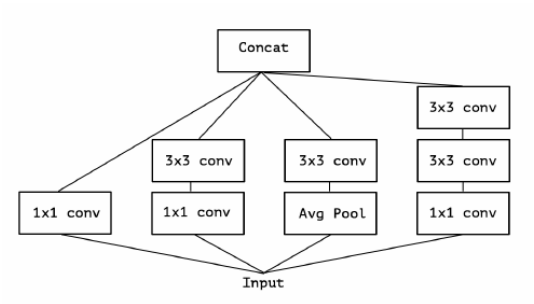


Figure 2: A simplified Inception module.

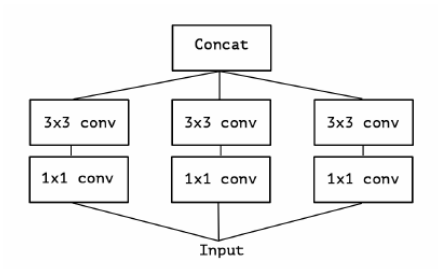
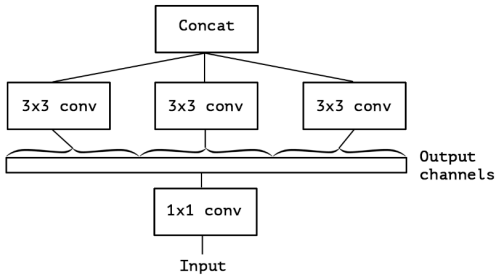
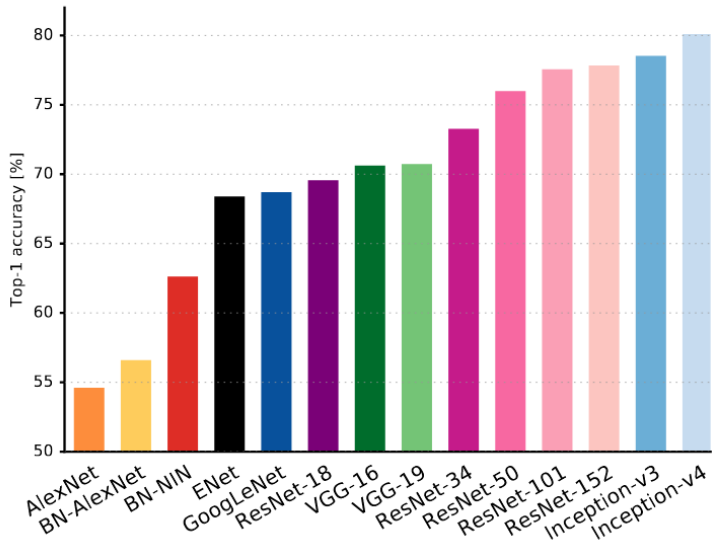


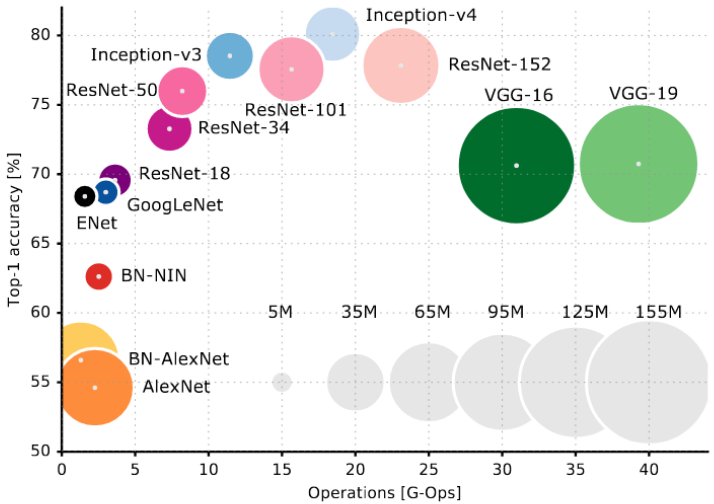
Figure 3. A strictly equivalent reformulation of the simplified Inception module.



Francois Chollet, Xception: Deep learning with depthwise separable convolutions, CVPR 2017

Architecture: topology and wiring





A Canziani, A Paszke, E Culurciello, An analysis of deep neural network models for practical applications, ICLR 2017

Architecture: topology and wiring

Q: What are my choices?

Architecture: topology and wiring

Q: What are my choices?

A: Connections, filter size, number of filters, strides, activation functions, dropout, number of neurons

Architecture: topology and wiring

Q: What are my choices?

A: Connections, filter size, number of filters, strides, activation functions, dropout, number of neurons

Q: How do I know what is a good choice?

Architecture: topology and wiring

Q: What are my choices?

A: Connections, filter size, number of filters, strides, activation functions, dropout, number of neurons

Q: How do I know what is a good choice?

A: Performance on benchmark datasets

Architecture: topology and wiring

Q: What are my choices?

A: Connections, filter size, number of filters, strides, activation functions, dropout, number of neurons

Q: How do I know what is a good choice?

A: Performance on benchmark datasets

Q: Why did I choose that architecture?

Architecture: topology and wiring

Q: What are my choices?

A: Connections, filter size, number of filters, strides, activation functions, dropout, number of neurons

Q: How do I know what is a good choice?

A: Performance on benchmark datasets

Q: Why did I choose that architecture?

A: ¯_(ツ)_/¯

Architecture: topology and wiring

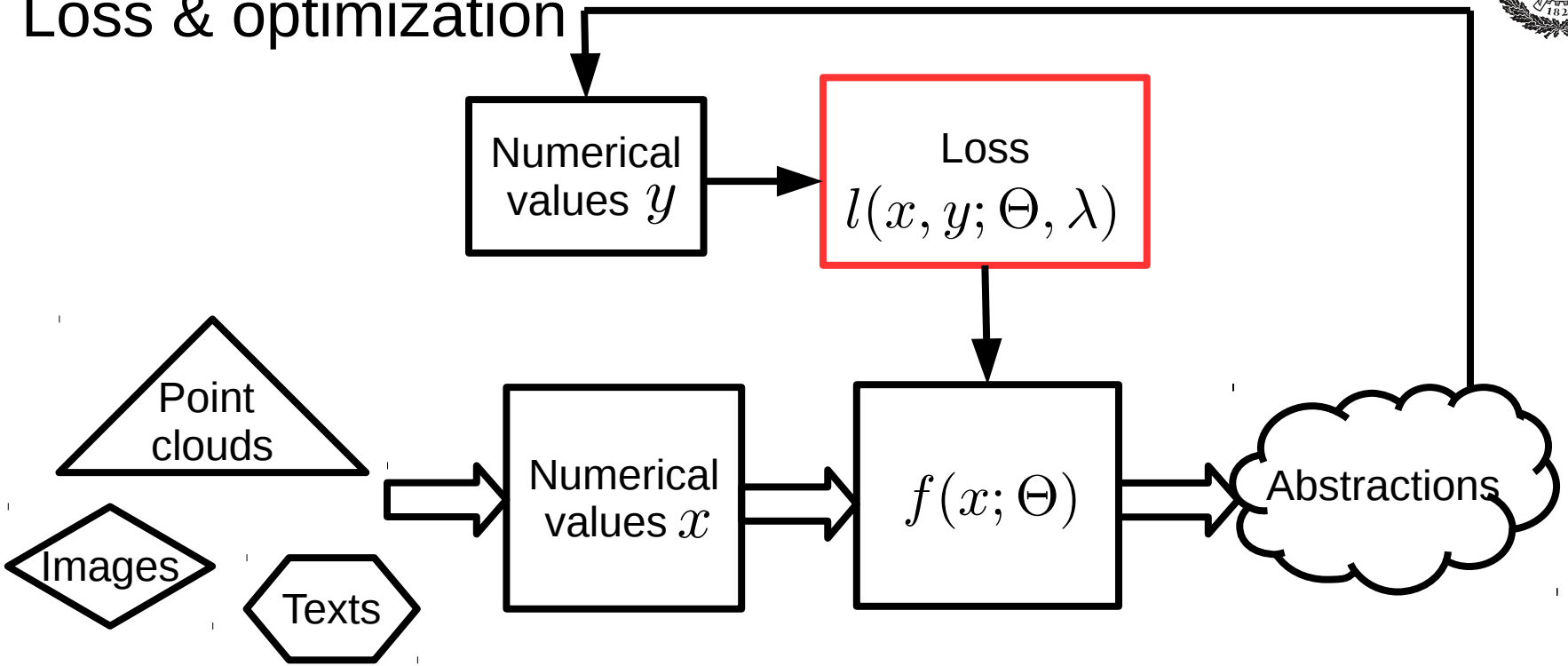
- Network Architecture Search (NAS)
  - Reinforcement learning
  - Progressive network search
  - Differentiable architecture search
- Macro vs micro search
  - Macro search: “give me the best network”
  - Micro search: “given a skeleton of the network, give me the best wiring/components for each block”

https://en.wikipedia.org/wiki/Neural_architecture_search
https://www.automl.org/automl/literature-on-neural-architecture-search/

Architecture: efficiency improvement

- Pruning (remove insignificant weights/connections)
- Compression (quantization, Huffman encoding, binarization)
- Weight sharing (use the same weights to reduce the dof)
- Low rank approximation (tensor/matrix factorization)

Loss & optimization



Loss functions

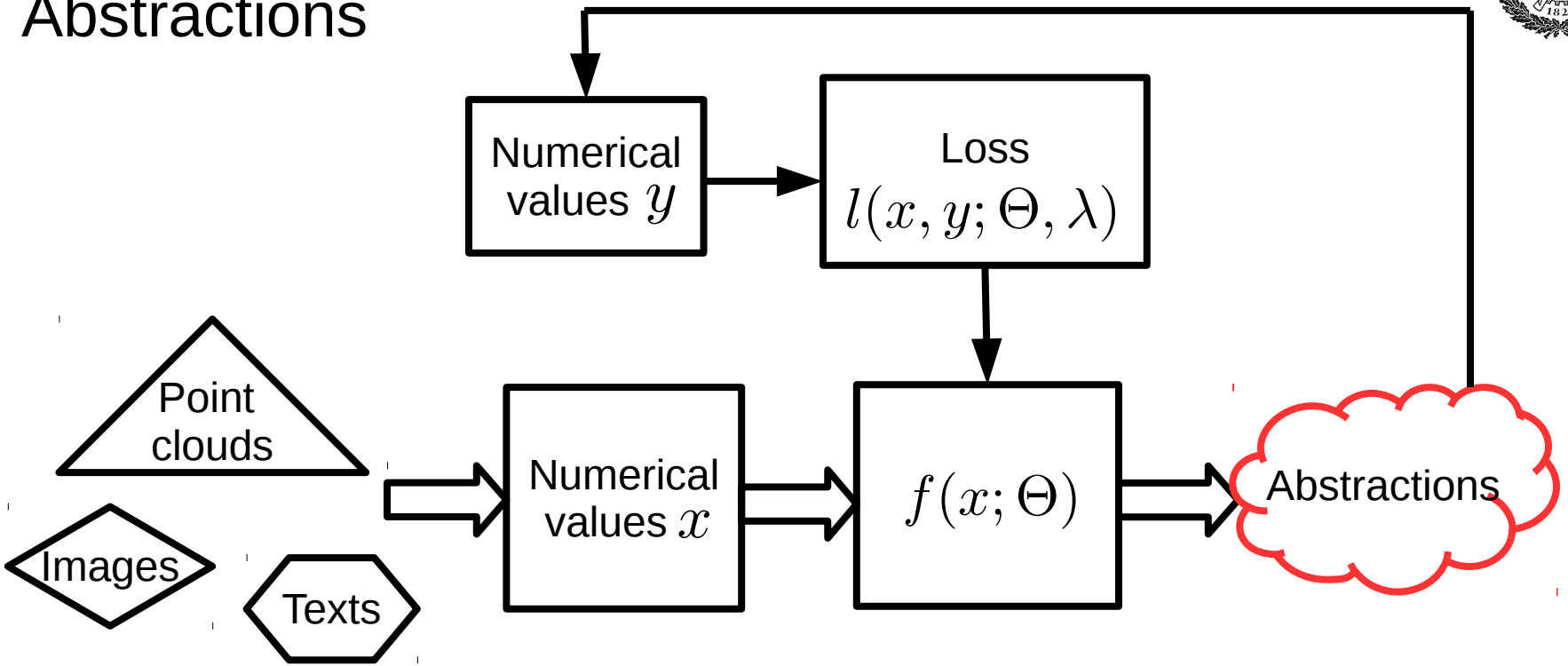
- Cross entropy loss: $l_i = -(y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i))$
- Hinge loss: $l_i = \sum_{j \neq y_i} \max(0, \hat{y}_j - \hat{y}_{y_i} + 1)$
- Mean Absolute Error (L1 Loss): $l_i = |\hat{y}_i - y_i|$
- Mean Square Error/Quadratic Loss (L2 Loss): $l_i = (\hat{y}_i - y_i)^2$

i : data point index
y : label
$\hat{y}$: prediction from the network

Optimization

- Optimizer
  - Stochastic Gradient Descent
  - Momentum (“moving average”)
  - ADAM
- Learning rate decay (mainly for first order optimization)
- Regularization (“specificity”→ “generalization”)
  - Empirical risk minimization & statistical learning theory
  - Drop out (set activation to zero)
  - Batch normalization
  - Data augmentation
  - Ensemble learning

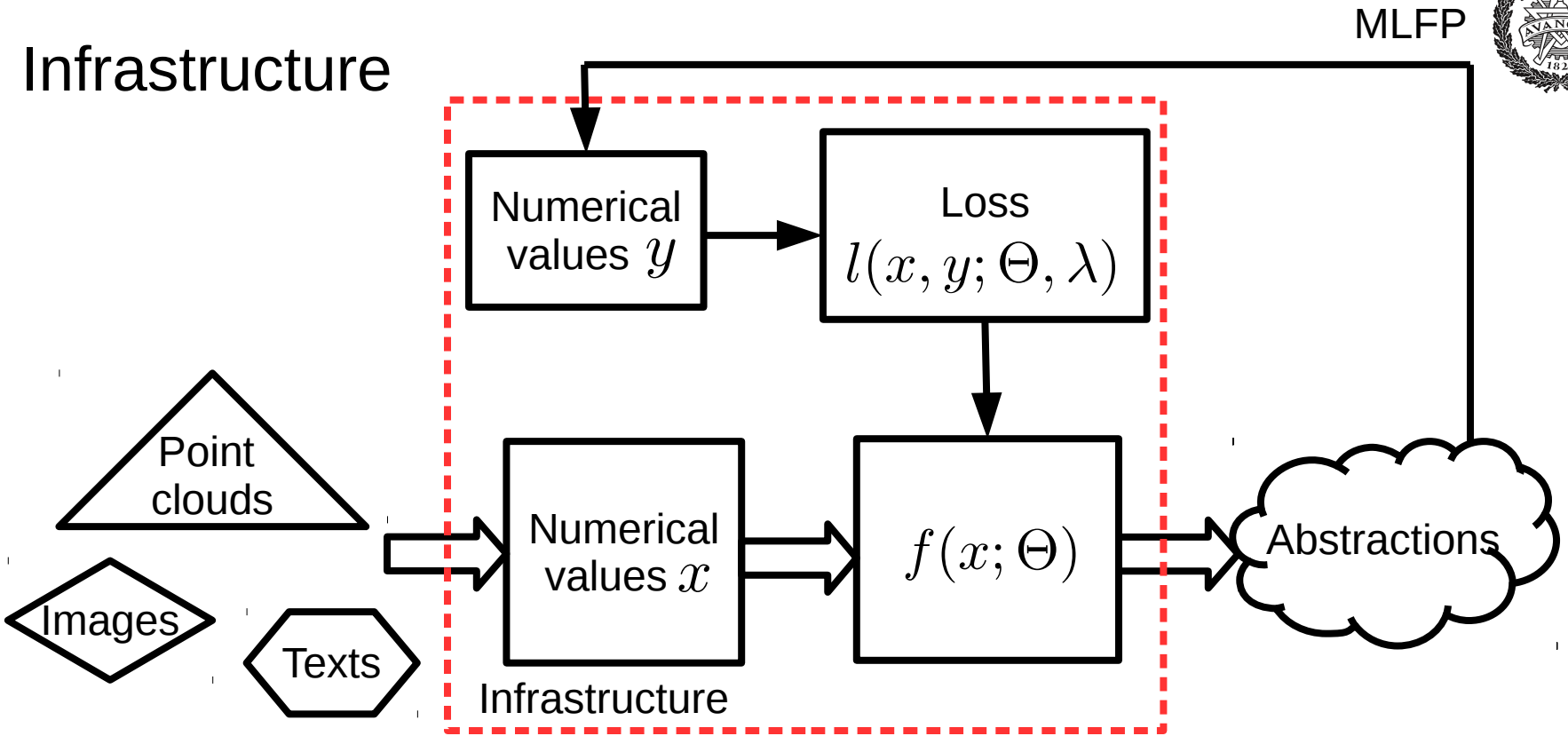
Abstractions



Abstractions

- Annotations for supervised learning
  - Annotation guideline
  - Manual annotation
  - Automation
  - Clustering, semi-supervised learning, crowd sourcing and queries
  - KPI of use cases

Infrastructure



Infrastructure

- CPU/GPU/storage
- CPU: preparing data
- GPU: backprops and inference
  - Laptop/workstation
  - GPU servers
- Storage: fast random access
- Hosting/cooling

Conclusion

- Deep learning works
- Deep learning evolves fast
- Deep learning brings everything together
  - “Big data” is here
  - Modern hardware + low level libraries enable scalability
  - Data modeling for fancy applications
  - Highly complex models: low bias, high variance
  - Regularization to balance bias/variance trade-off
  - Optimizers: faster and better
- Deep learning is mysterious
  - Adversarial attacks
  - Automation, analysis and verification

Thank you