Deep Learning on Java

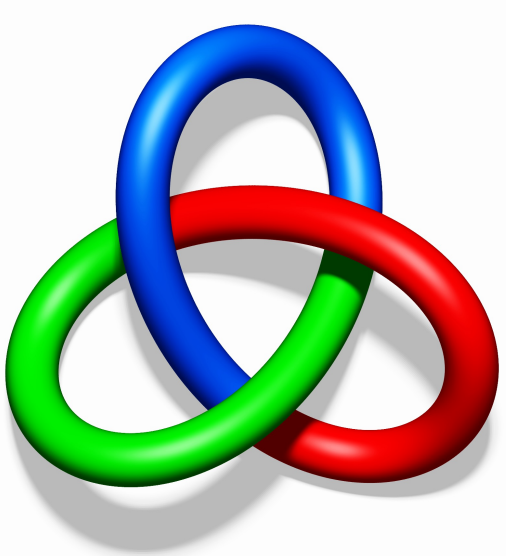
Breandan Considine

JFokus 2017

Who am I?

- Background in Computer Science, Machine Learning
- Worked for a small ad-tech startup out of university
- Spent two years as Developer Advocate @JetBrains
- Interested in machine learning and speech recognition
- Enjoy writing code, traveling to conferences, reading
- Say hello! @breandan | breandan.net | bre@ndan.co

What is “three”?



Size

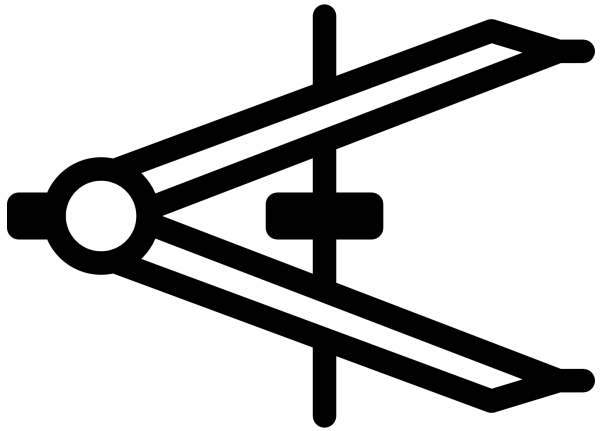
Shape

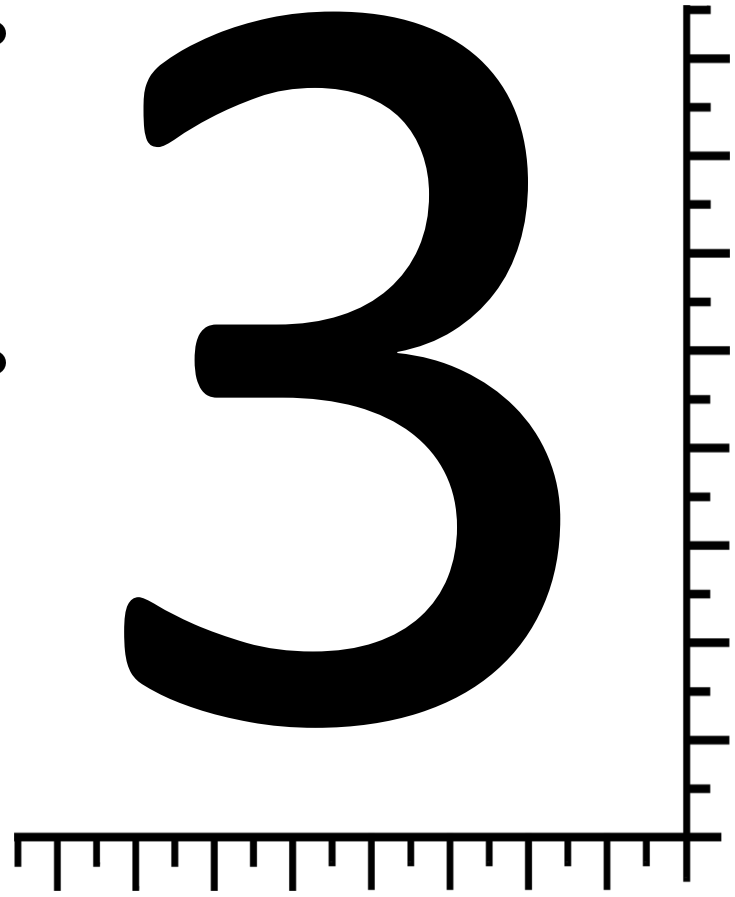
Distance

Similarity

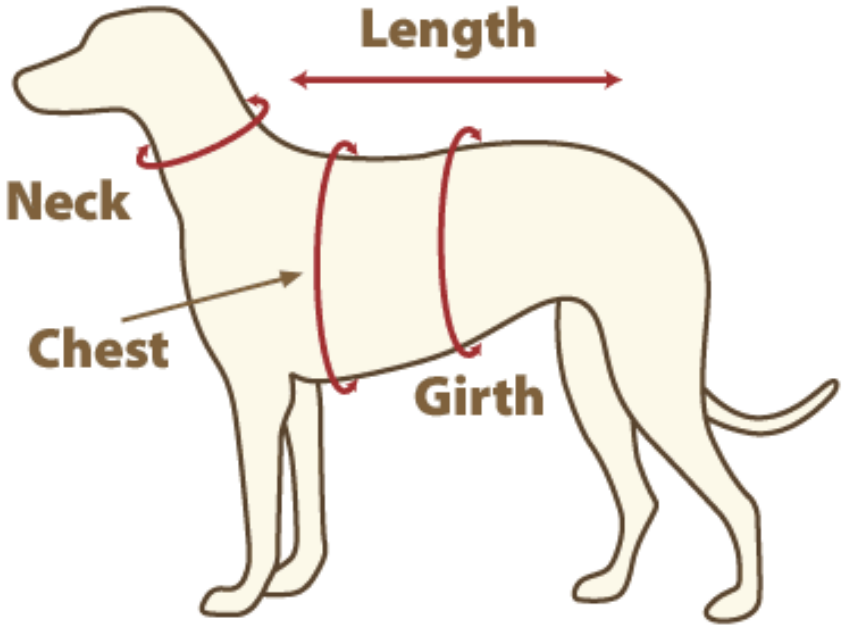
Separation

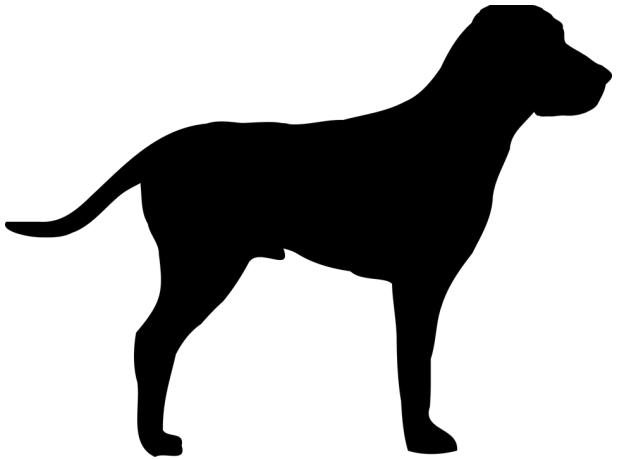
Orientation

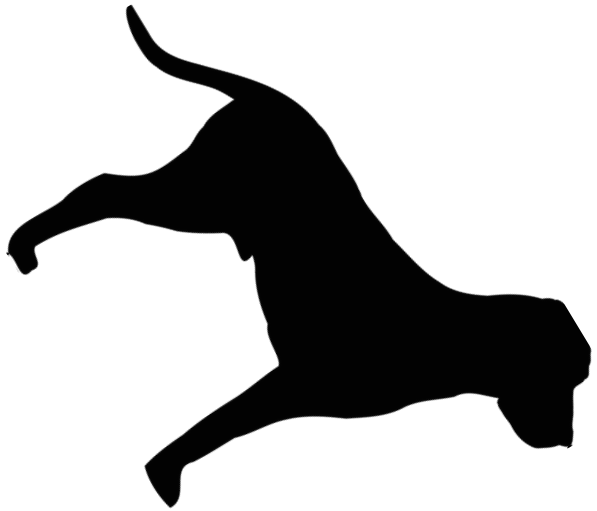




What is “dog”?

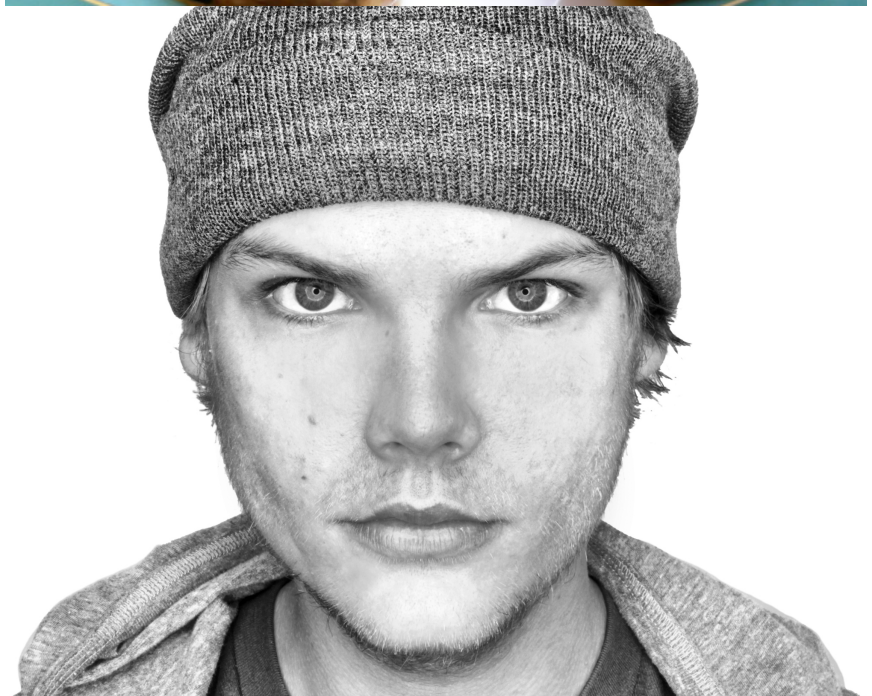






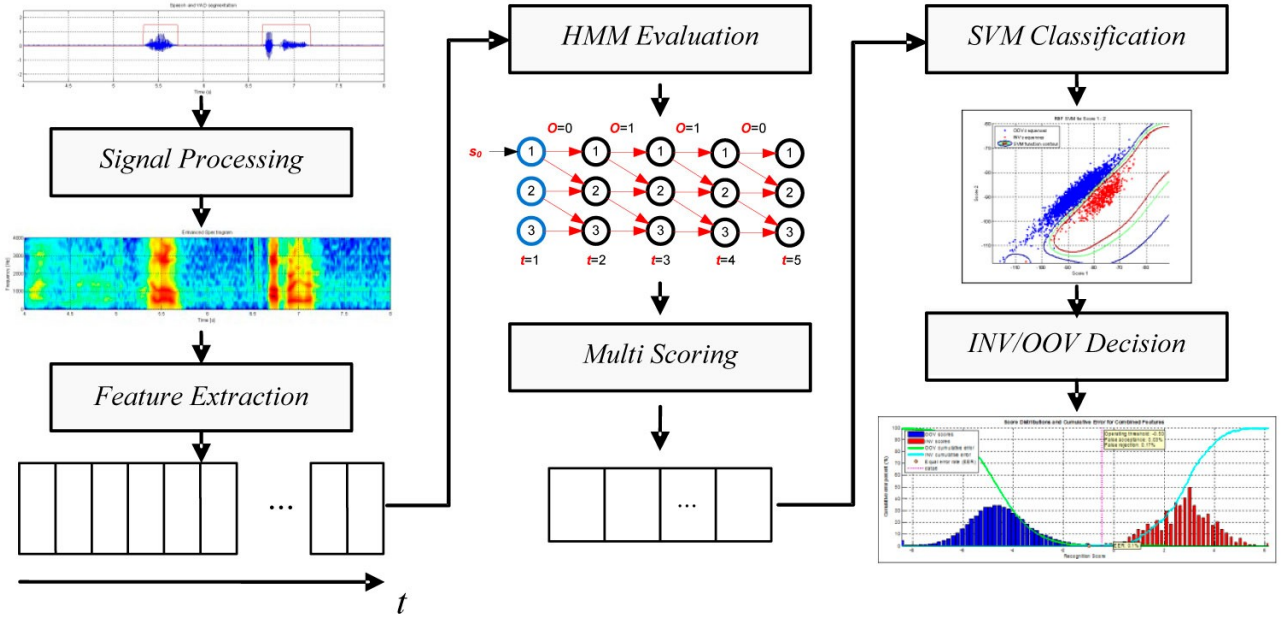
DOG SIZE	DOG BREED
Extra Small	Chihuahua Miniature Breed Yorkie
Small	Cavalier Spaniel Miniature Poodle Dachshund Jack Russell Westie Whippet
Medium	Beagle Cocker Spaniel Springer Spaniel Staffordshire Bull Terrier Standard Poodle
Large	Boxer Doberman German Shepherd Labrador Retriever Setter
Extra Large	Rottweiler Wolfhound Pyrenean Bloodhound Great Dane St. Bernard

What is “Swedish”?

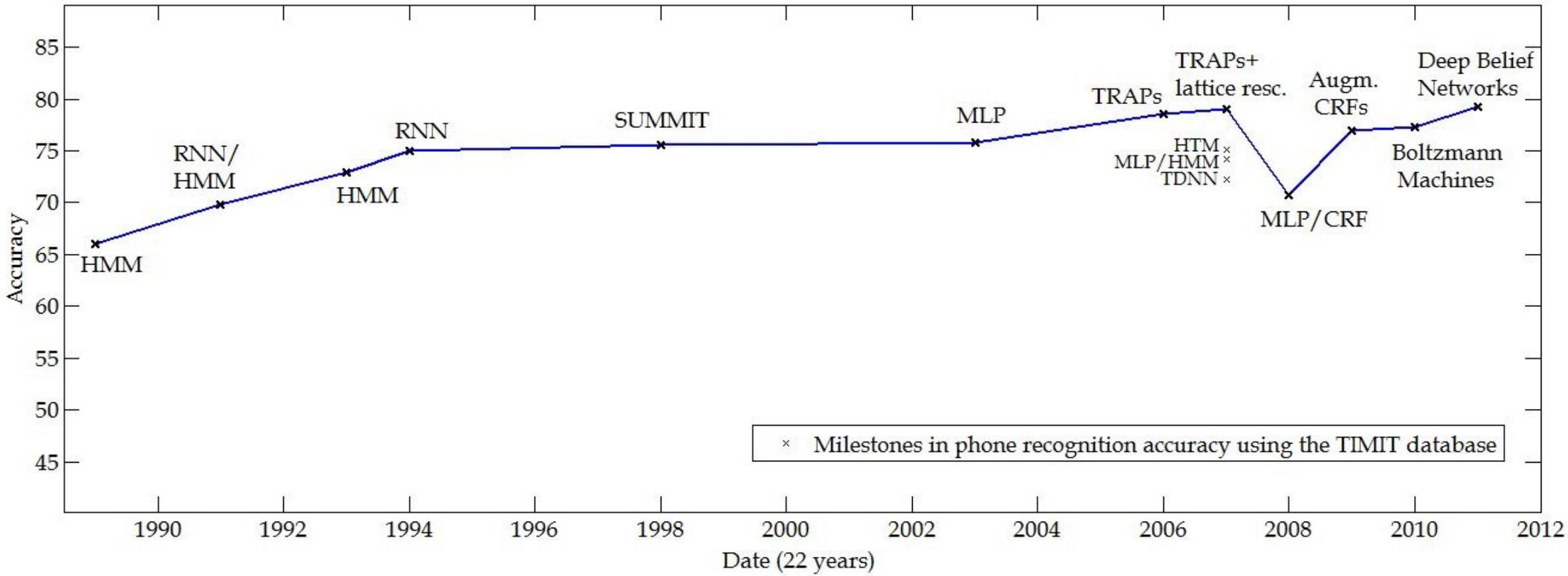


Early Speech Recognition

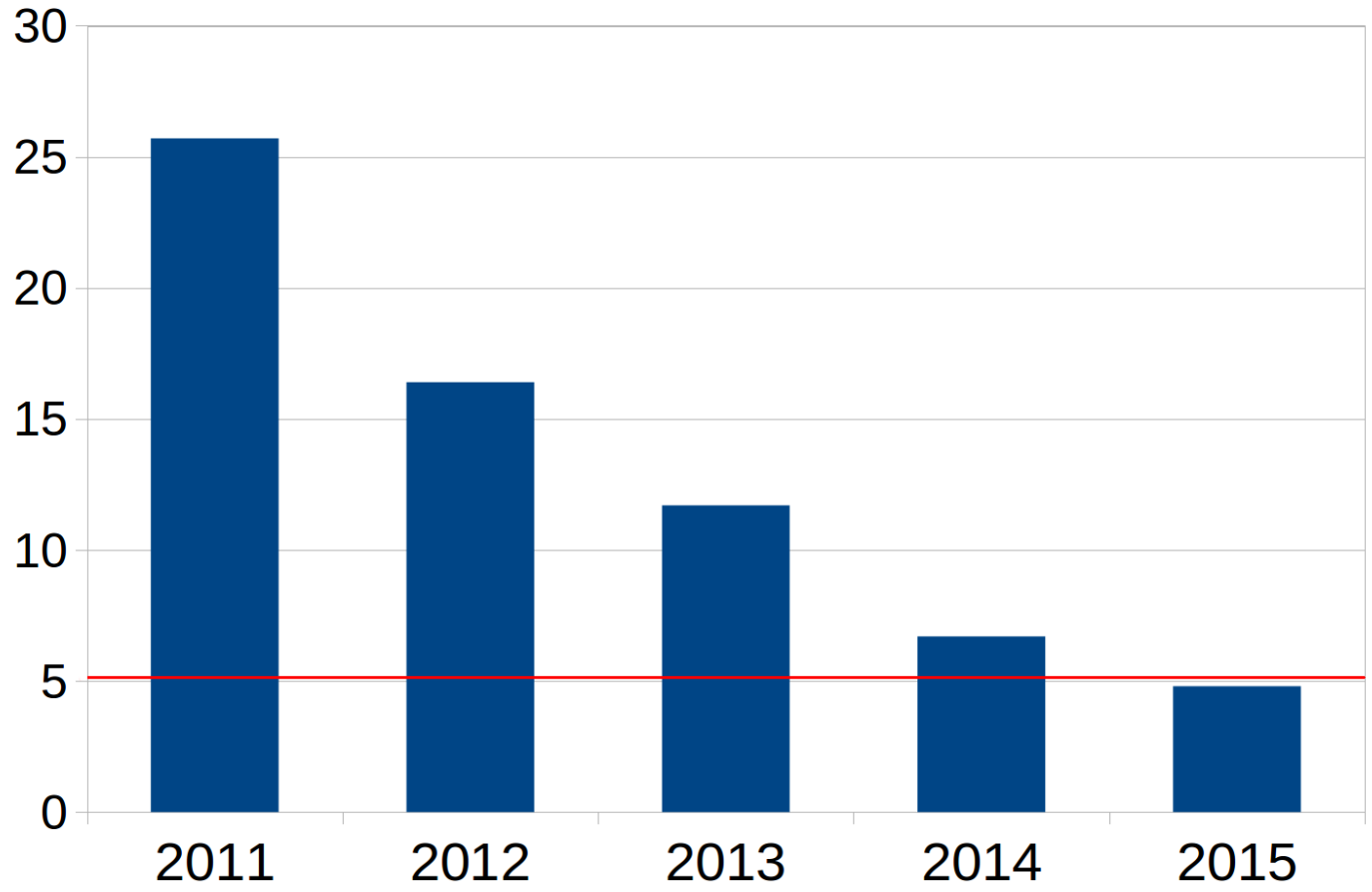
- Requires lots of handmade feature engineering
- Poor results: >25% WER for HMM architectures



Automatic speech recognition in 2011

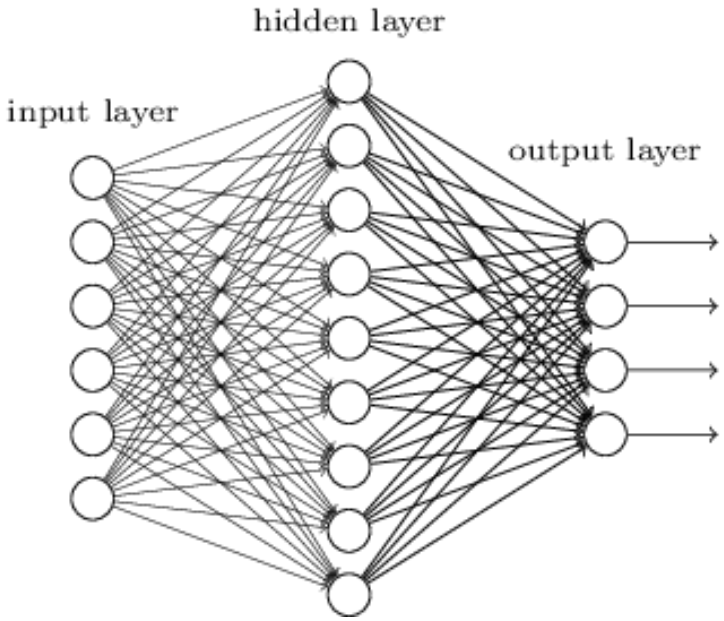


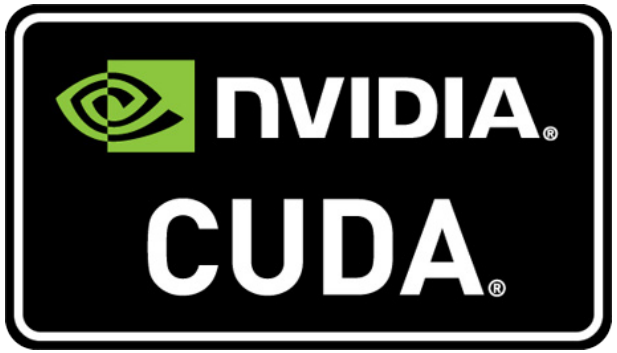
Year over year Top-5 Recognition Error



What happened?

- Bigger data
- Faster hardware
- Smarter algorithms





What is machine learning?

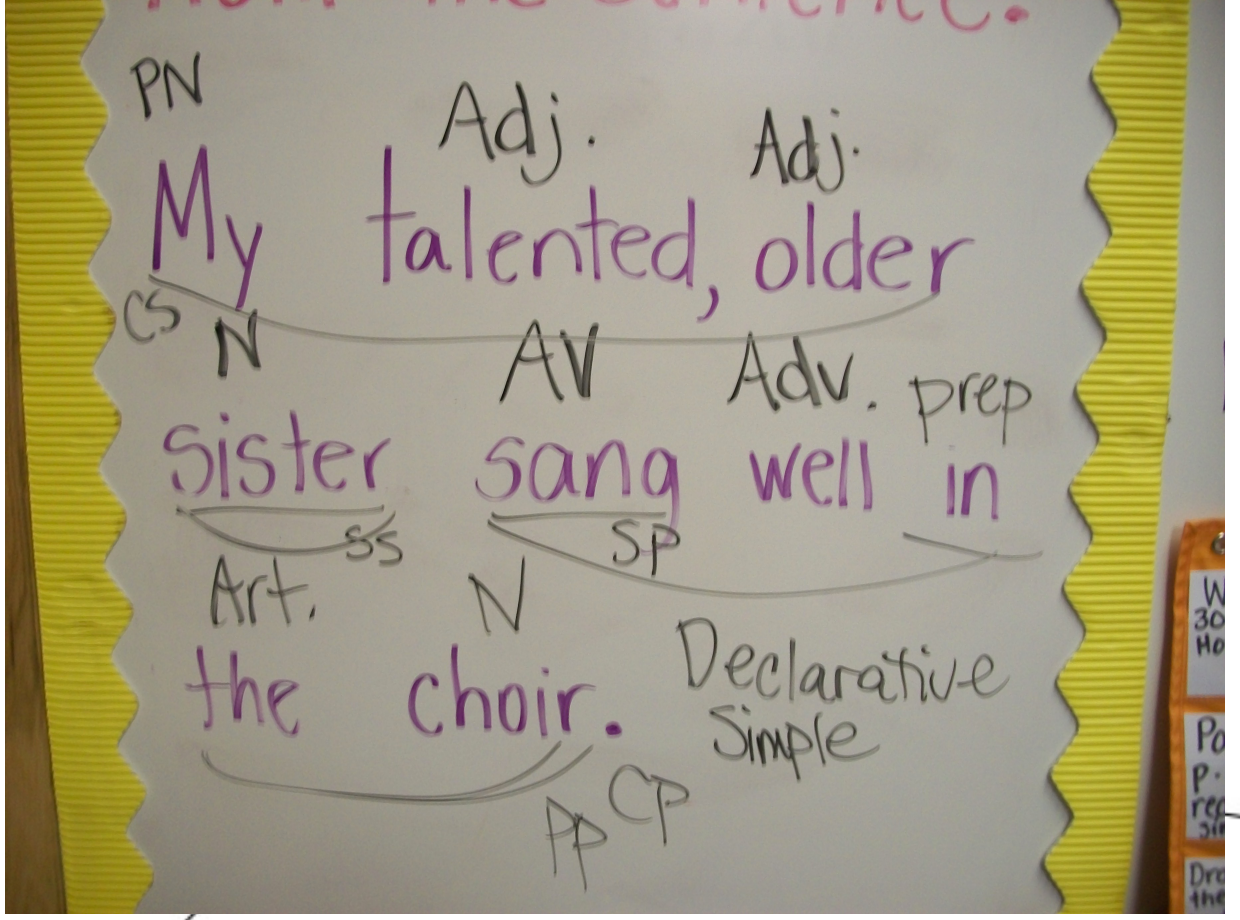
- Prediction
- Categorization
- Anomaly detection
- Personalization
- Adaptive control
- Playing games

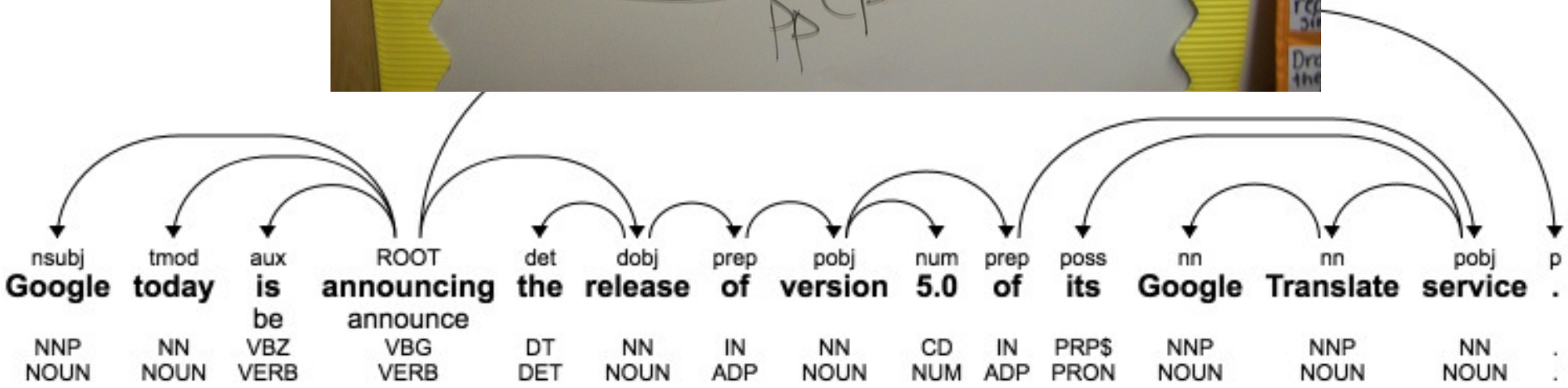
Traditional education

- One-size-fits-all curriculum
- Teaching process is repetitive
- Students are not fully engaged
- Memorization over understanding
- Encouragement can be inconsistent
- Teaches to the test (not the real world)

How can we improve education?

- Personalized learning
- Teaching assistance
- Adaptive feedback
- Active engagement
- Spaced repetition
- Assistive technology


PN
Adj.
Adj.
My talented, older
CS
N
AV
Adv.
prep
sister sang well in
SS
SP
Art.
N
the choir.
Declarative
Simple
PP
CP


nsubj tmod aux ROOT det dobj prep pobj num prep poss nn nn pobj p
Google today is announcing the release of version 5.0 of its Google Translate service .
be
announce
NNP NN VBZ VBG DT NN IN NN CD IN PRP$ NNP NNP NN .
NOUN NOUN VERB VERB DET NOUN ADP NOUN NUM ADP PRON NOUN NOUN NOUN .

Original

Had Walter Alvarez not asked his father, the Nobel Prize-winning physicist Luis Alvarez, how long the clay had taken to deposit, the younger Alvarez may not have thought to use iridium, an element rarely found on earth but more plentiful in meteorites, to answer this question. Iridium, in the form of microscopic grains of cosmic dust, is constantly raining down on the planet. The Alvarezes reasoned that if the clay layer had taken a significant amount of time to deposit, it would contain detectable levels of iridium.

Translated

Had Walter Alvarez not asked his father, the Nobel Prize-winning physicist [1] Luis Alvarez, how long the Clay had taken to deposit, the younger Alvarez may not have thought to use iridium [2], a part uncommonly discovered on earth but more great amount in metal, stones from outer space, to answer this question. iridium [2], in the form of small sized grains of all space dust, is constantly raining down on the moving body moving round sun. The Alvarezes reasoned that if the Clay level had taken an important amount of time to deposit, it would have within measurable levels of iridium [2].

---

physicist expert in physics. Continue reading.

iridium chemical element, atomic number 77, atomic weight 193.1, a very hard steel-grey metal with a high melting-point, having a number of important uses in industry, for example, in watch-making and pen making. Continue reading.

Proof. Omitted. □

Lemma 0.1. *Let $\mathcal{C}$ be a set of the construction.*

Let $\mathcal{C}$ be a gerber covering. Let $\mathcal{F}$ be a quasi-coherent sheaves of $\mathcal{O}$-modules. We have to show that

$$\mathcal{O}_{\mathcal{O}_X} = \mathcal{O}_X(\mathcal{L})$$

.

Proof. This is an algebraic space with the composition of sheaves $\mathcal{F}$ on $X_{\acute{e}tale}$ we have

$$\mathcal{O}_X(\mathcal{F}) = \{morph_1 \times_{\mathcal{O}_X} (\mathcal{G}, \mathcal{F})\}$$

where $\mathcal{G}$ defines an isomorphism $\mathcal{F} \to \mathcal{F}$ of $\mathcal{O}$-modules. □

Lemma 0.2. *This is an integer $\mathcal{Z}$ is injective.*

Proof. See Spaces, Lemma ??. □

Lemma 0.3. *Let S be a scheme. Let X be a scheme and X is an affine open covering. Let $\mathcal{U} \subset \mathcal{X}$ be a canonical and locally of finite type. Let X be a scheme. Let X be a scheme which is equal to the formal complex.*

The following to the construction of the lemma follows.

Let X be a scheme. Let X be a scheme covering. Let

$$b : X \to Y' \to Y \to Y \to Y' \times_X Y \to X.$$

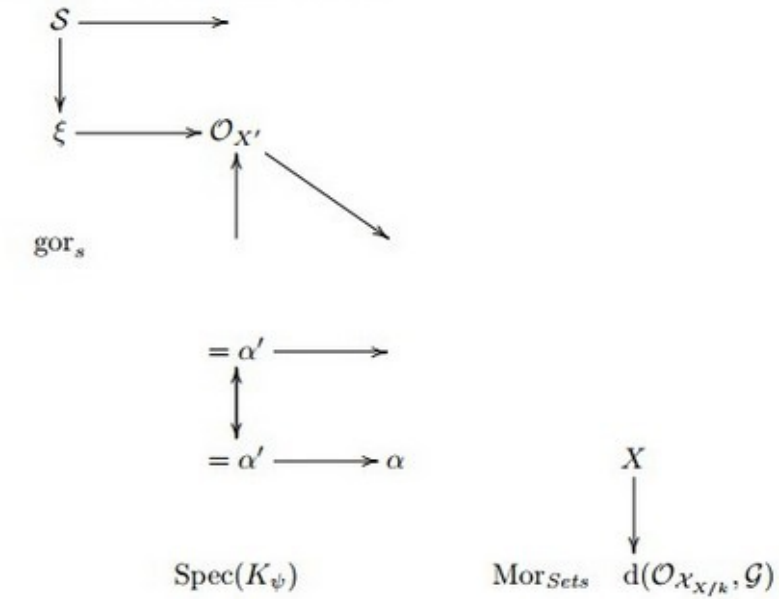
be a morphism of algebraic spaces over S and Y.

Proof. Let X be a nonzero scheme of X. Let X be an algebraic space. Let $\mathcal{F}$ be a quasi-coherent sheaf of $\mathcal{O}_X$-modules. The following are equivalent

(1) $\mathcal{F}$ is an algebraic space over S.
(2) If X is an affine open covering.

Consider a common structure on X and X the functor $\mathcal{O}_X(U)$ which is locally of finite type. □

This since $\mathcal{F} \in \mathcal{F}$ and $x \in \mathcal{G}$ the diagram



is a limit. Then $\mathcal{G}$ is a finite type and assume S is a flat and $\mathcal{F}$ and $\mathcal{G}$ is a finite type f_*. This is of finite type diagrams, and

- the composition of $\mathcal{G}$ is a regular sequence,
- $\mathcal{O}_{X'}$ is a sheaf of rings.

□

Proof. We have see that $X = \mathrm{Spec}(R)$ and $\mathcal{F}$ is a finite type representable by algebraic space. The property $\mathcal{F}$ is a finite morphism of algebraic stacks. Then the cohomology of X is an open neighbourhood of U. □

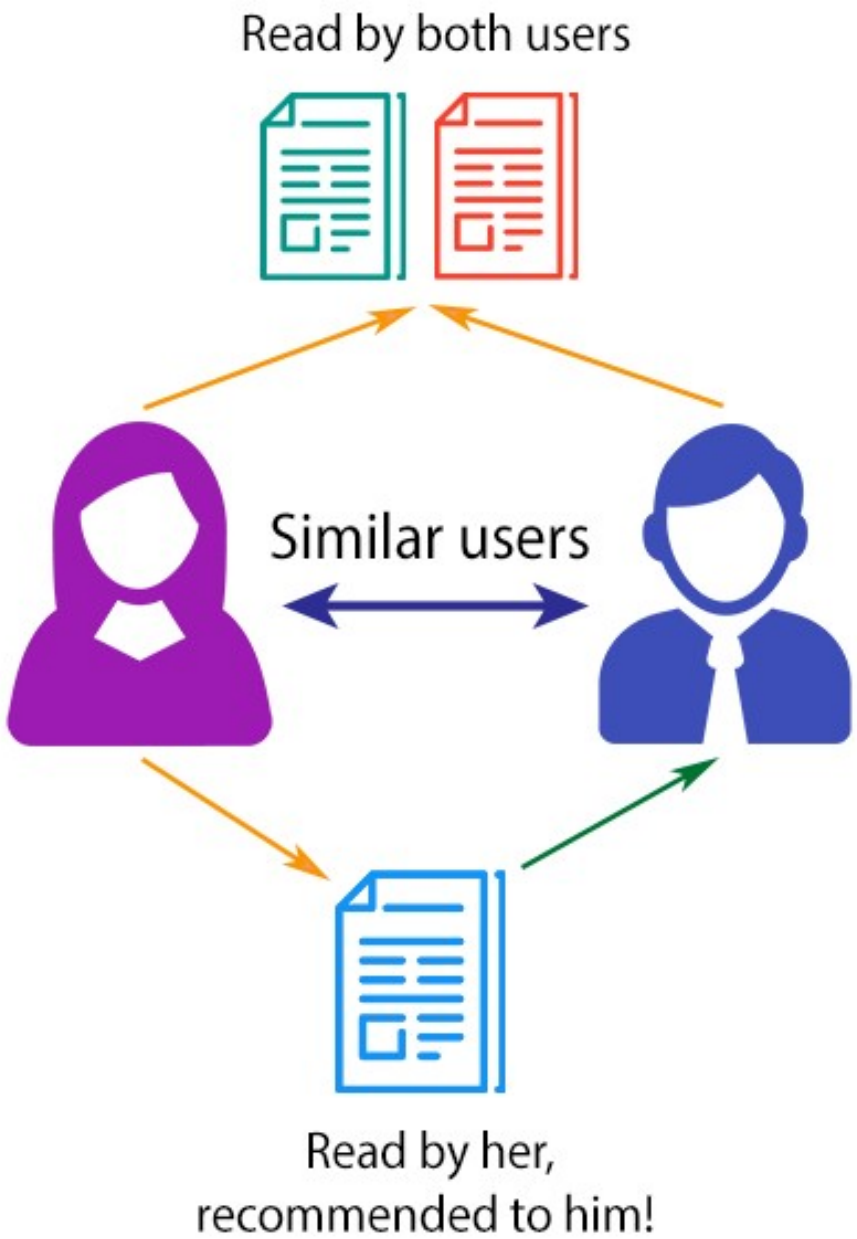
Proof. This is clear that $\mathcal{G}$ is a finite presentation, see Lemmas ??.
A *reduced above* we conclude that U is an open covering of $\mathcal{C}$. The functor $\mathcal{F}$ is a "field

$$\mathcal{O}_{X,x} \longrightarrow \mathcal{F}_{\overline{x}} \quad \text{-1}(\mathcal{O}_{X_{\acute{e}tale}}) \longrightarrow \mathcal{O}_{X_\ell}^{-1}\mathcal{O}_{X_\lambda}(\mathcal{O}_{X_\eta}^{\overline{v}})$$

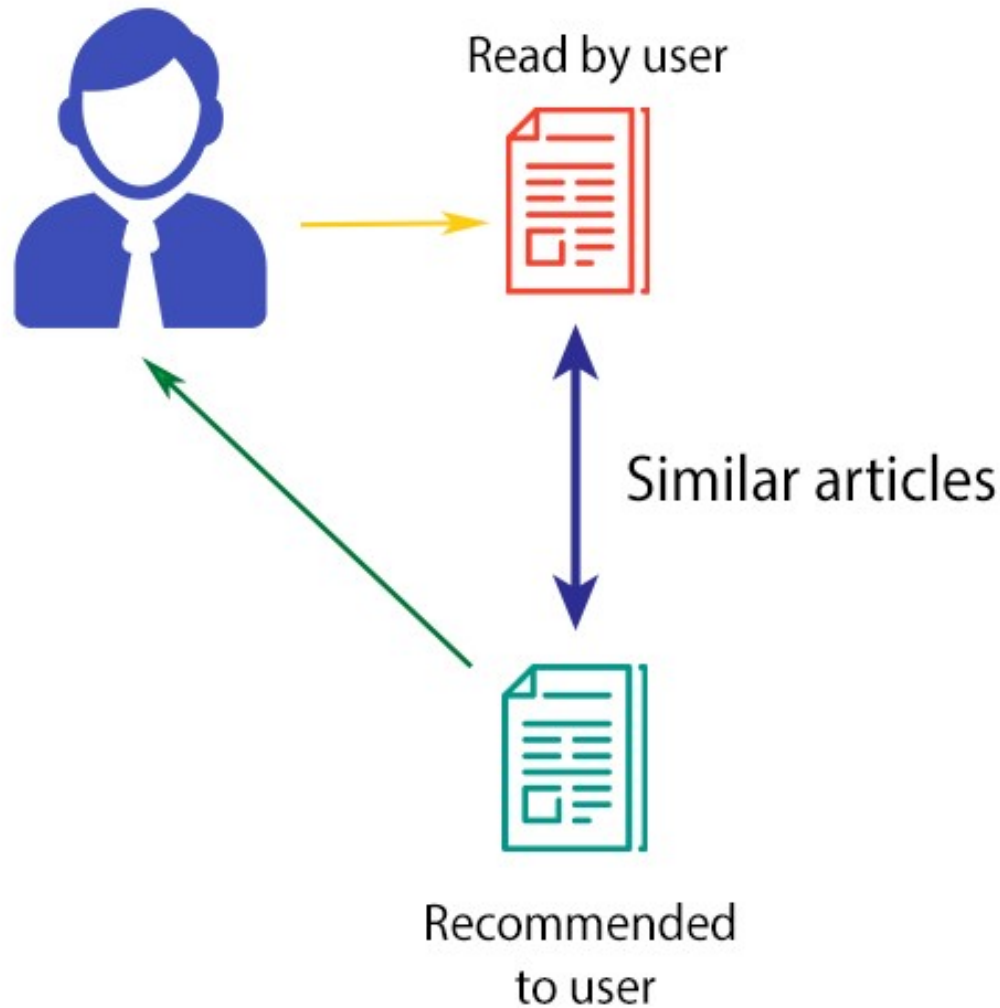
is an isomorphism of covering of $\mathcal{O}_{X_i}$. If $\mathcal{F}$ is the unique element of $\mathcal{F}$ such that X is an isomorphism.
The property $\mathcal{F}$ is a disjoint union of Proposition ?? and we can filtered set of presentations of a scheme $\mathcal{O}_X$-algebra with $\mathcal{F}$ are opens of finite type over S.
If $\mathcal{F}$ is a scheme theoretic image points. □

If $\mathcal{F}$ is a finite direct sum $\mathcal{O}_{X_\lambda}$ is a closed immersion, see Lemma ??. This is a sequence of $\mathcal{F}$ is a similar morphism.

COLLABORATIVE FILTERING



CONTENT-BASED FILTERING



‹ Back to search results for "halloween costume"

Roll over image to zoom in

Rasta Imposta

Rasta Imposta Lightweight Penguin Costume

4.5 out of 5 stars ▾ 341 customer reviews | 27 answered questions

Price: $21.50 - $99.69 & FREE Returns. Details

Sale: Lower price available on select options

Fit: As expected (80%) ▾

Size:

Select ⇕ Size Chart

Color: Black/white

- 100% Polyester
- Imported
- Hand Wash
- Rasta imposta's lightweight penguin is both affordable and comfortable
- This costume comes with a tunic with an attached head and shoe covers

Share

To buy, select **Size**

Add to Cart

Add to List

Customers who viewed this item also bought

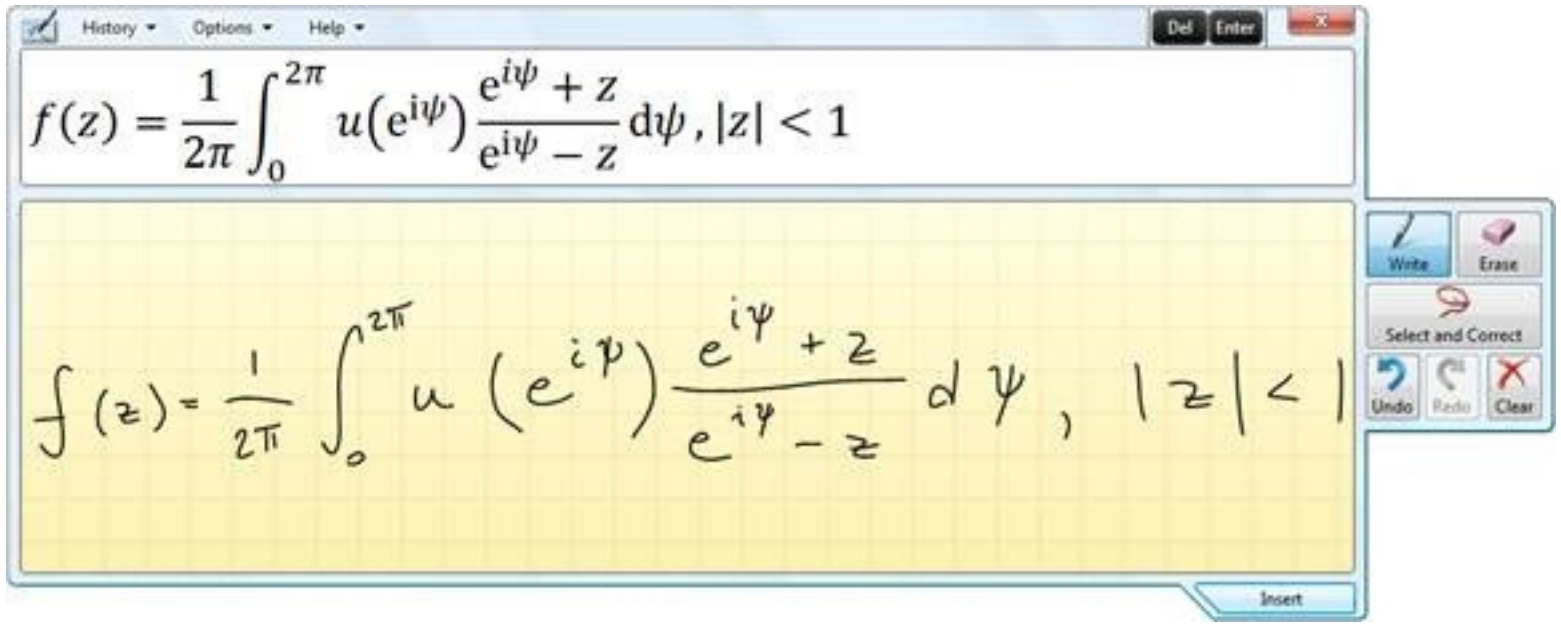
Rasta Imposta
$9.00 - $49.98

Forum Novelties
$13.67 - $52.49

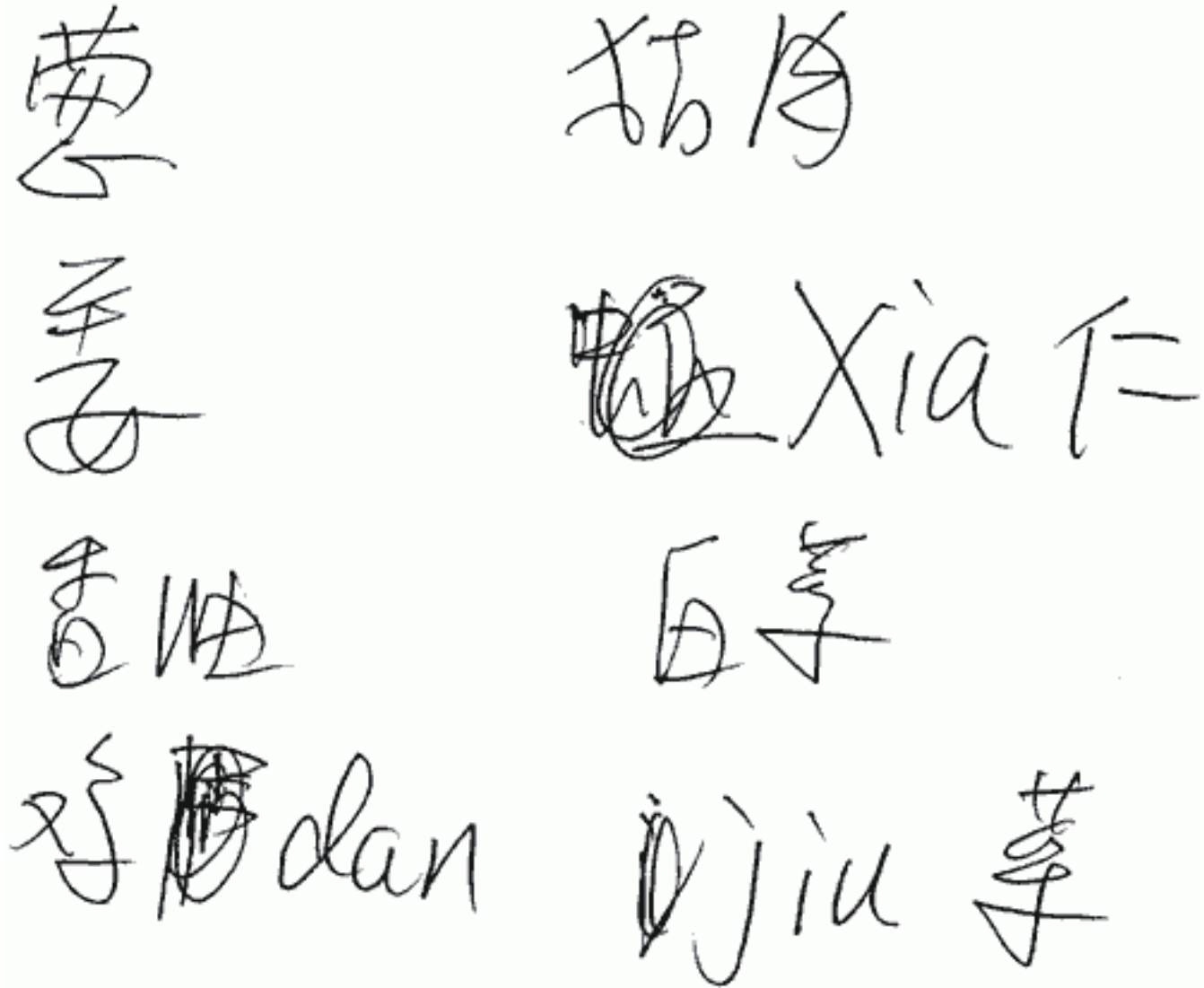
Rasta Imposta
$19.89 - $50.88

Rasta Imposta
$54.99 - $116.99

Handwriting recognition



Handwriting recognition



Speech recognition

colorless green ideas sleep furiously

Speech Verification / Recitation

Machine learning, for humans

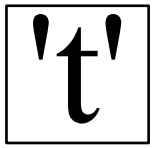
- Self-improvement
- Language learning
- Computer training
- Special education
- Reading comprehension
- Content generation

What’s a Tensor?

- A “tensor’ is just an n-dimensional array
- Useful for working with complex data
- We use (tiny) tensors every day!

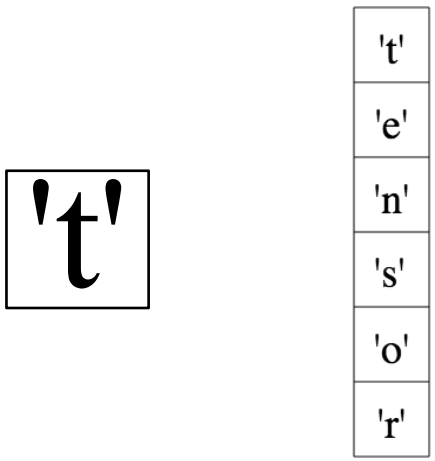
What’s a Tensor?

- A “tensor’ is just an n-dimensional array
- Useful for working with complex data
- We use (tiny) tensors every day!



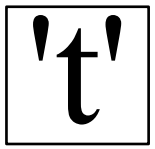
What's a Tensor?

- A "tensor' is just an n-dimensional array
- Useful for working with complex data
- We use (tiny) tensors every day!



What's a Tensor?

- A "tensor' is just an n-dimensional array
- Useful for working with complex data
- We use (tiny) tensors every day!



't'
'e'
'n'
's'
'o'
'r'

3	1	4	1
5	9	2	6
5	3	5	8
9	7	9	3
2	3	8	4
6	2	6	4

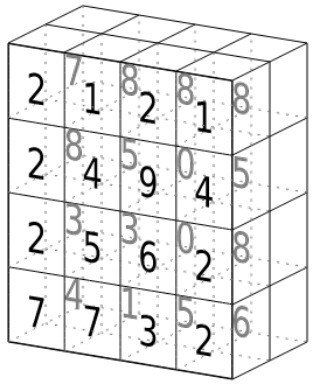
What's a Tensor?

- A "tensor' is just an n-dimensional array
- Useful for working with complex data
- We use (tiny) tensors every day!

't'

't'
'e'
'n'
's'
'o'
'r'

3	1	4	1
5	9	2	6
5	3	5	8
9	7	9	3
2	3	8	4
6	2	6	4



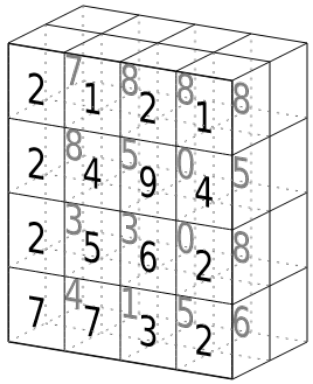
What's a Tensor?

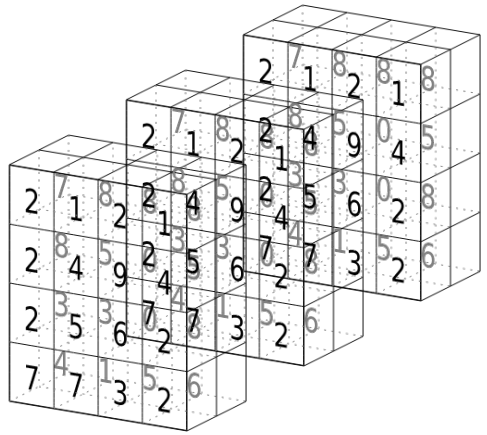
- A "tensor' is just an n-dimensional array
- Useful for working with complex data
- We use (tiny) tensors every day!

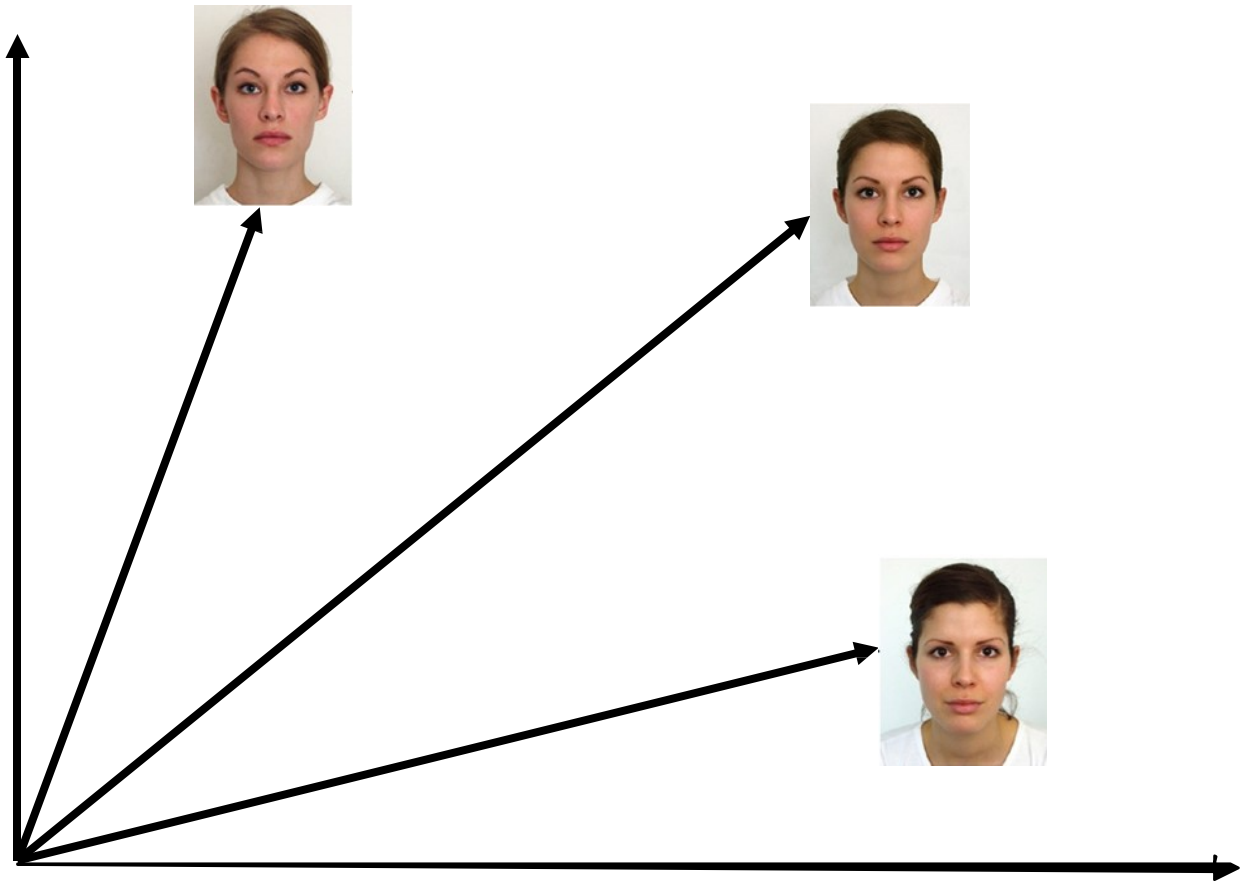
't'

't'
'e'
'n'
's'
'o'
'r'

3	1	4	1
5	9	2	6
5	3	5	8
9	7	9	3
2	3	8	4
6	2	6	4

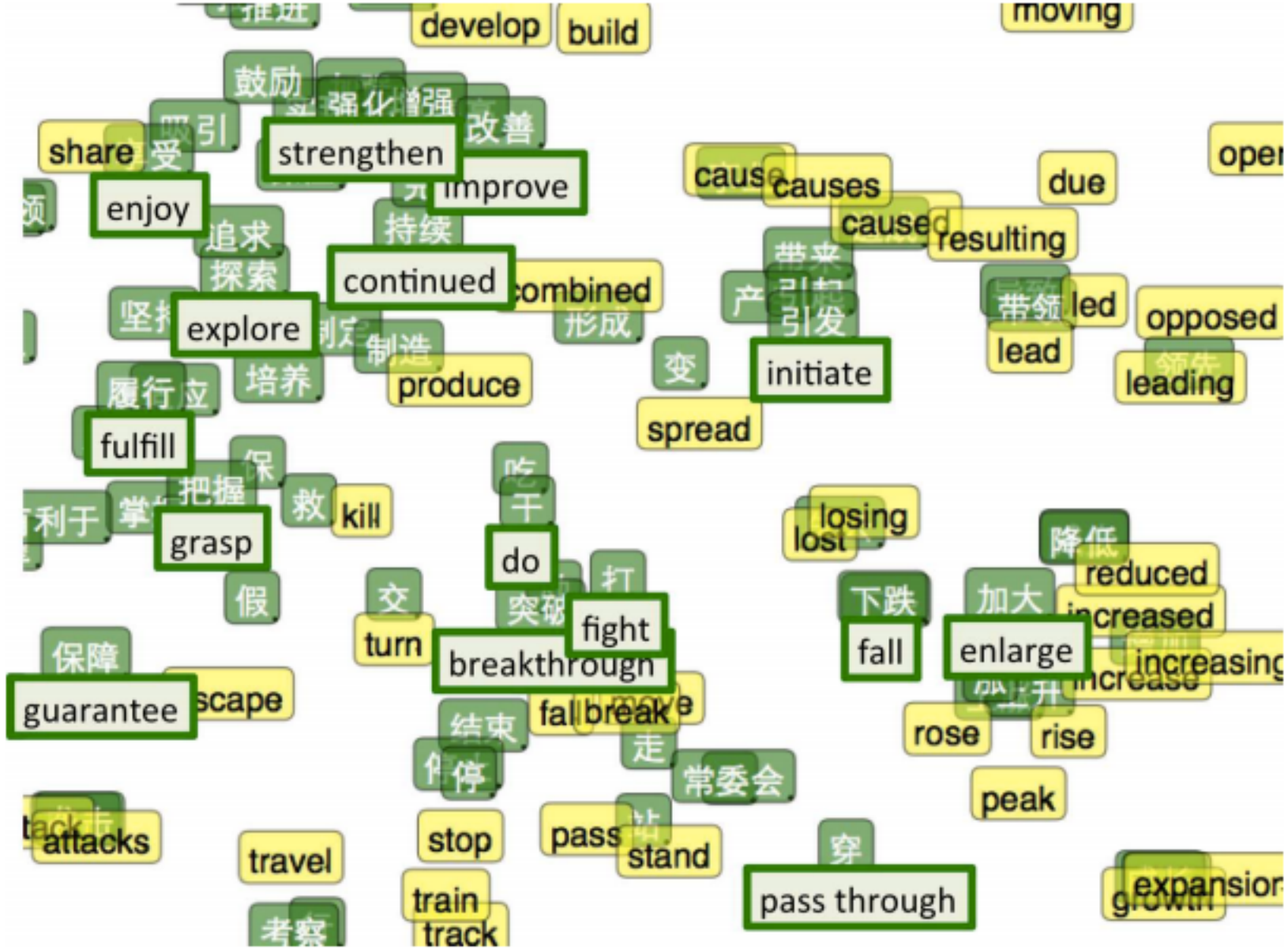




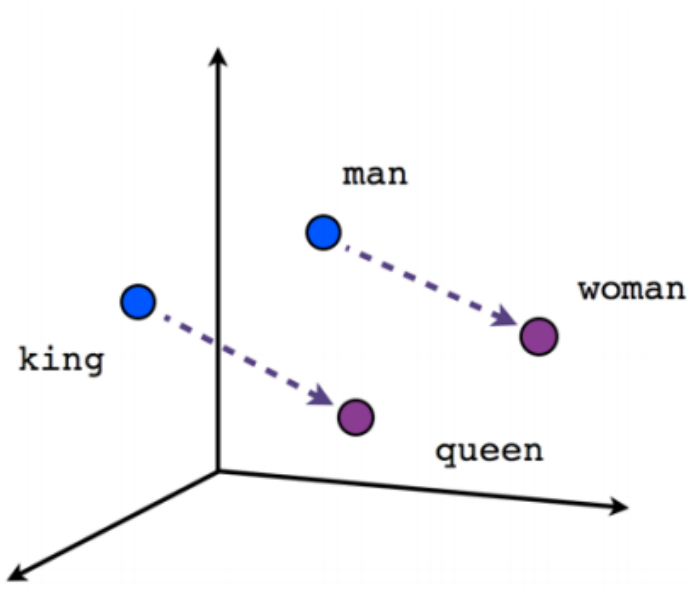


NxM image is a point in R^{NM}

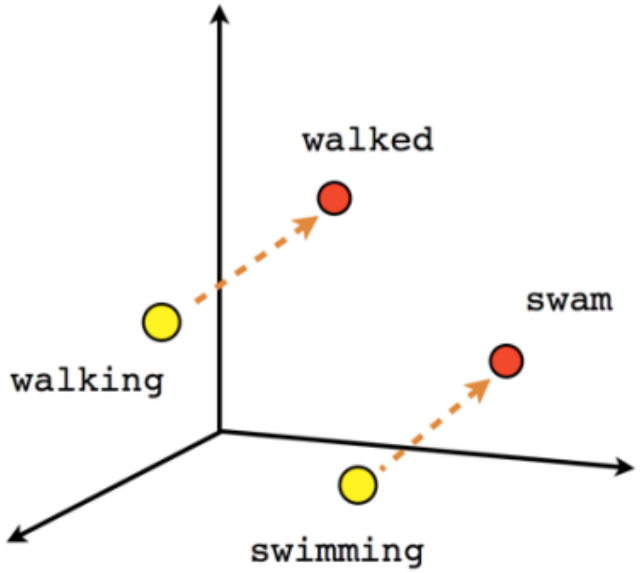
https://inst.eecs.berkeley.edu/~cs194-26/fa14/upload/files/proj5/cs194-dm/



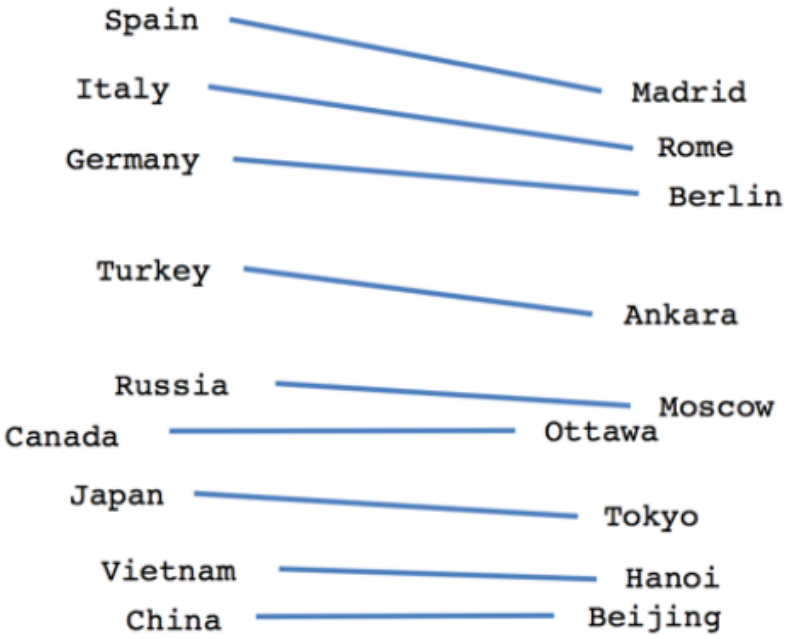
http://ai.stanford.edu/~wzou/emnlp2013_ZouSocherCerManning.pdf



Male-Female



Verb tense



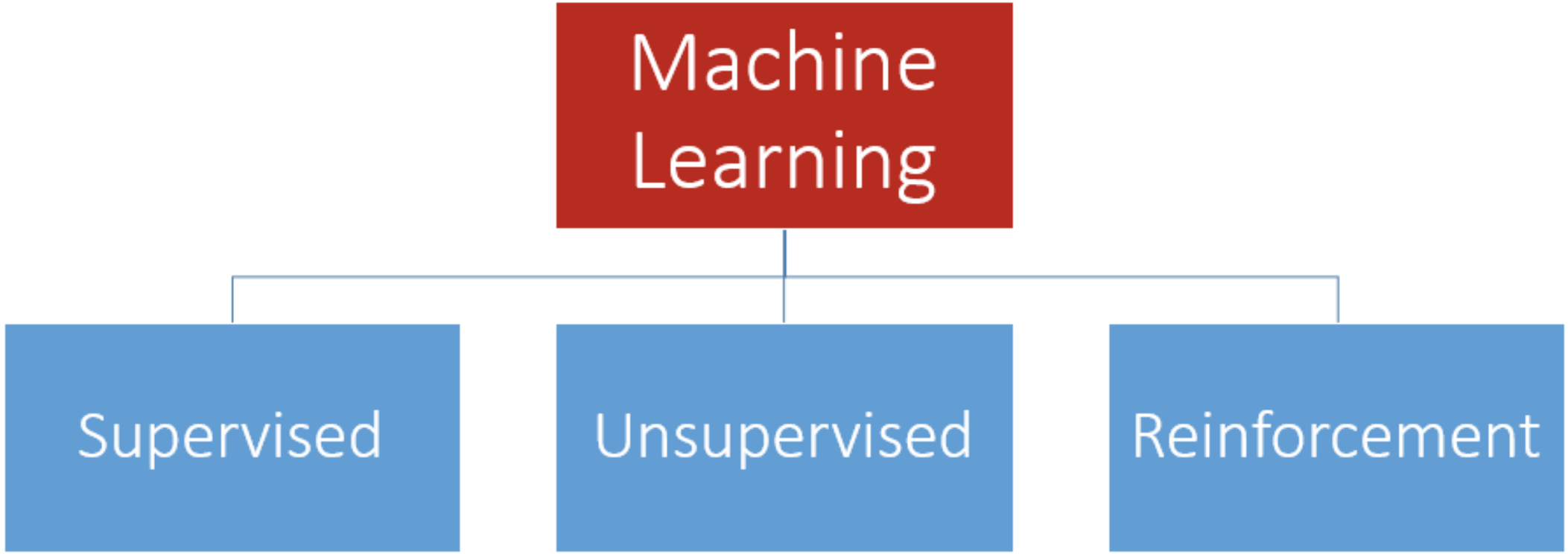
Country-Capital

http://www.snee.com/bobdc.blog/2016/09/semantic-web-semantics-vs-vect.html

Relationship	Example 1	Example 2	Example 3
France - Paris	Italy: Rome	Japan: Tokyo	Florida: Tallahassee
big - bigger	small: larger	cold: colder	quick: quicker
Miami - Florida	Baltimore: Maryland	Dallas: Texas	Kona: Hawaii
Einstein - scientist	Messi: midfielder	Mozart: violinist	Picasso: painter
Sarkozy - France	Berlusconi: Italy	Merkel: Germany	Koizumi: Japan
copper - Cu	zinc: Zn	gold: Au	uranium: plutonium
Berlusconi - Silvio	Sarkozy: Nicolas	Putin: Medvedev	Obama: Barack
Microsoft - Windows	Google: Android	IBM: Linux	Apple: iPhone
Microsoft - Ballmer	Google: Yahoo	IBM: McNealy	Apple: Jobs
Japan - sushi	Germany: bratwurst	France: tapas	USA: pizza

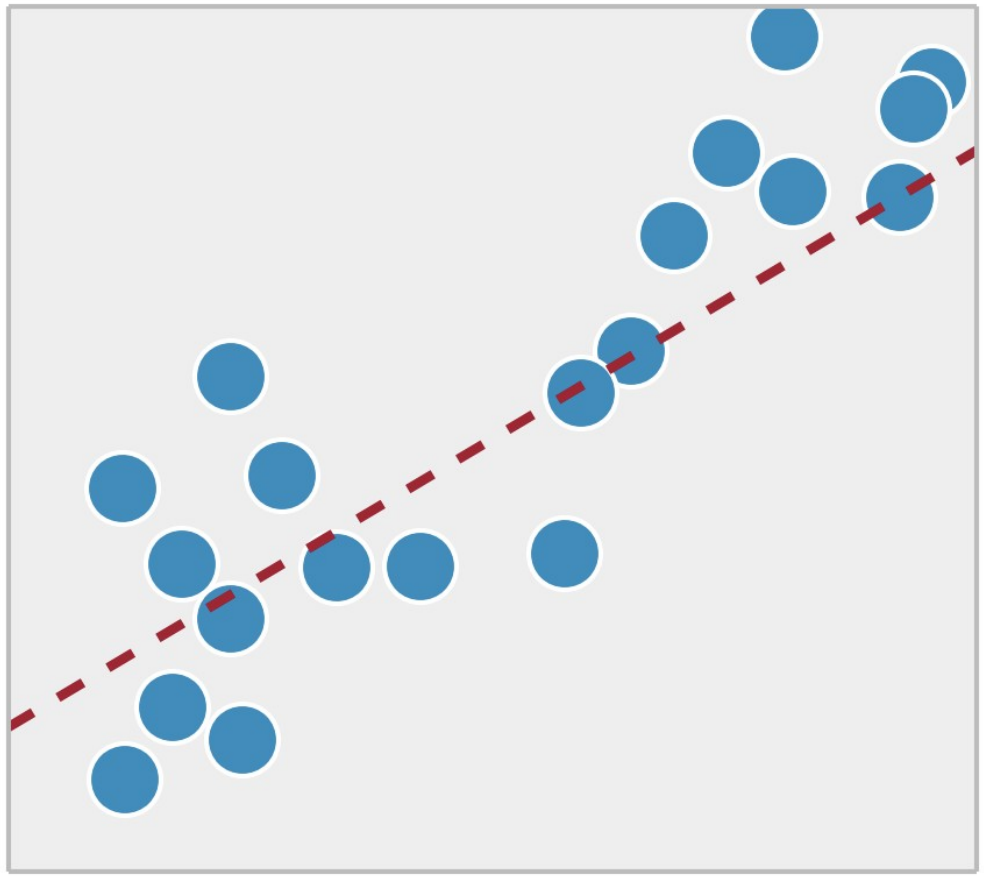
https://arxiv.org/pdf/1301.3781.pdf

Types of machine learning

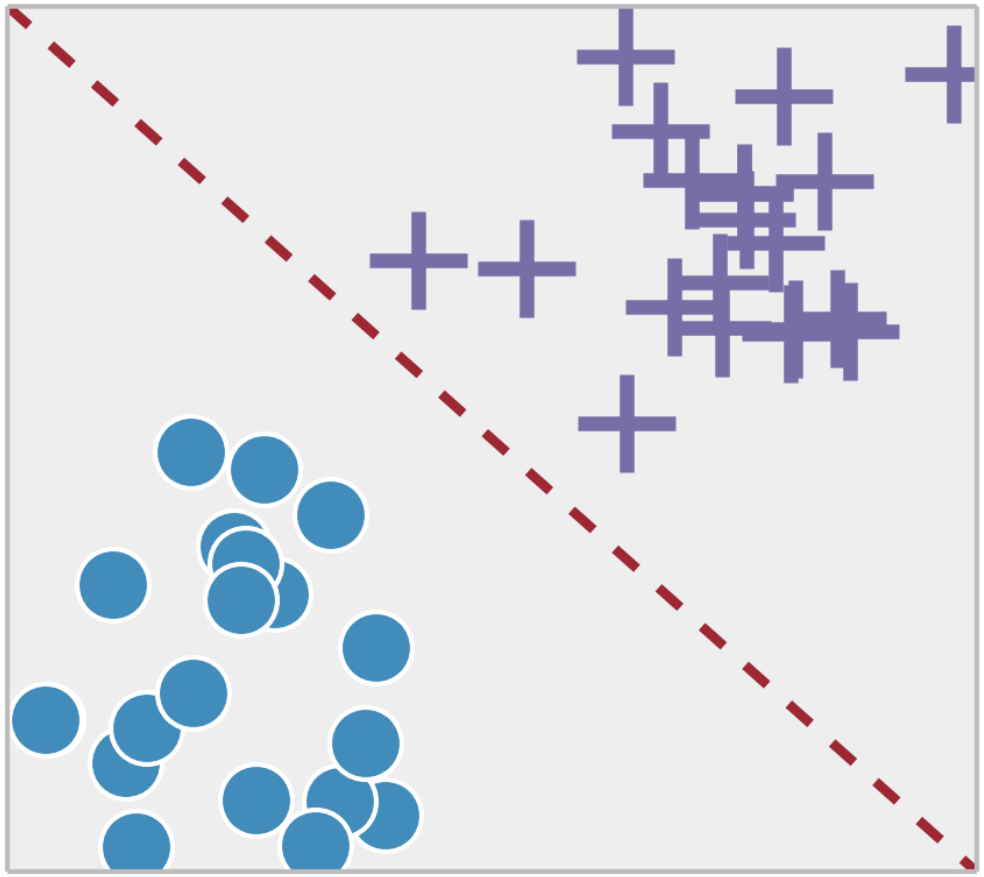


Supervised Learning

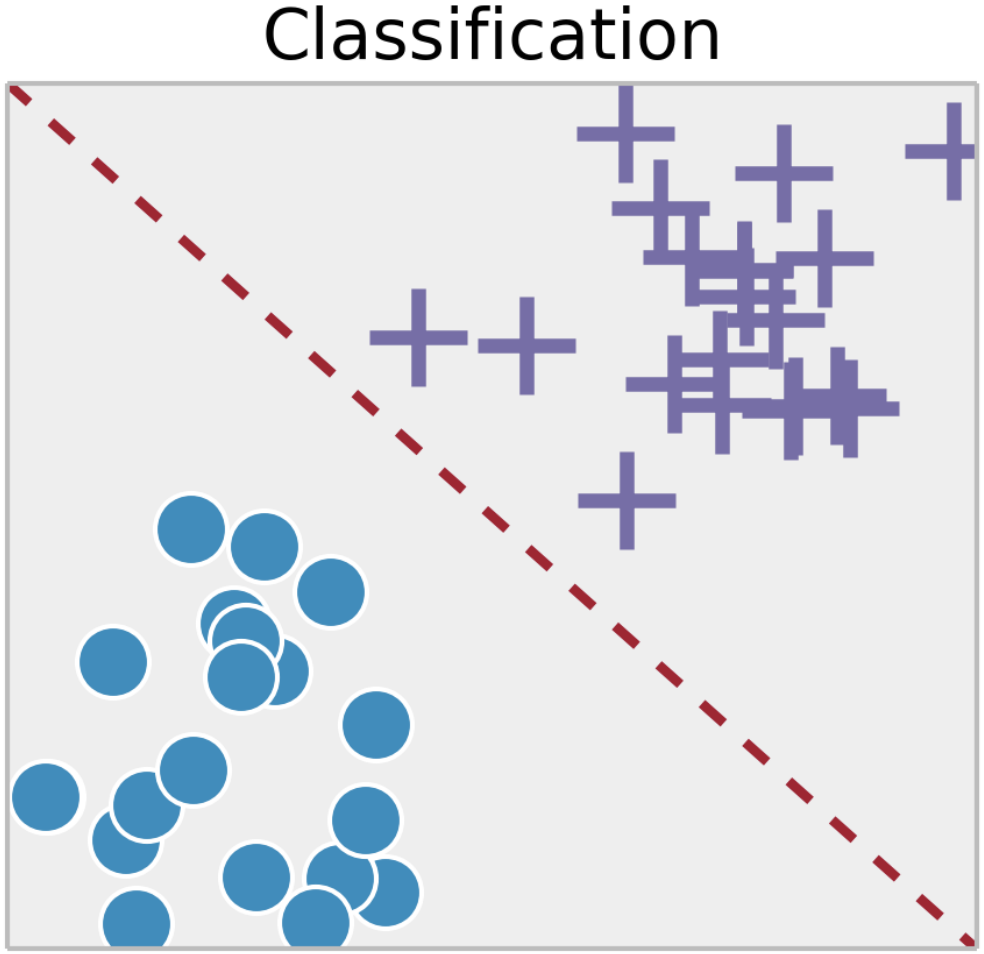
Regression

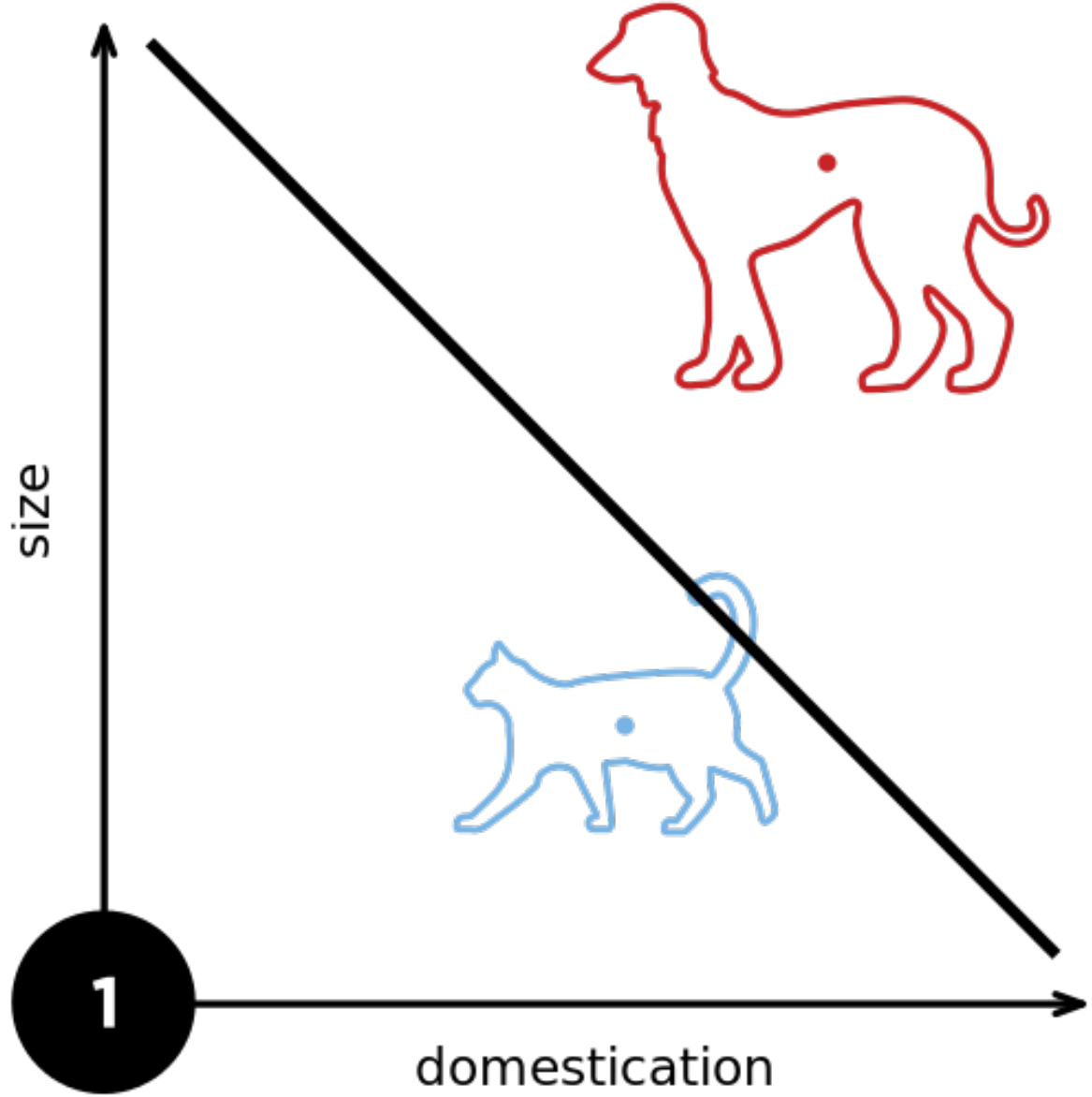


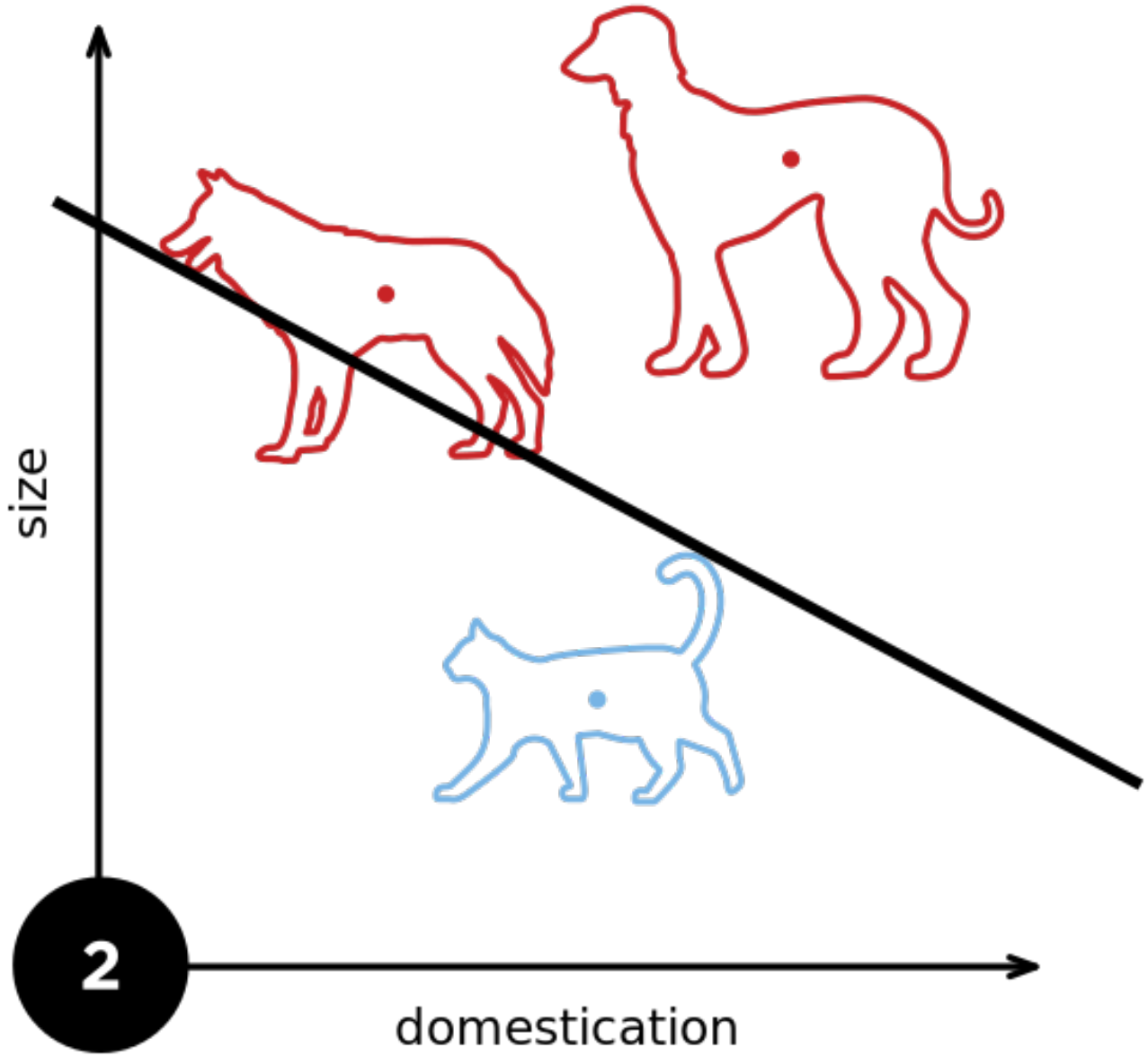
Classification

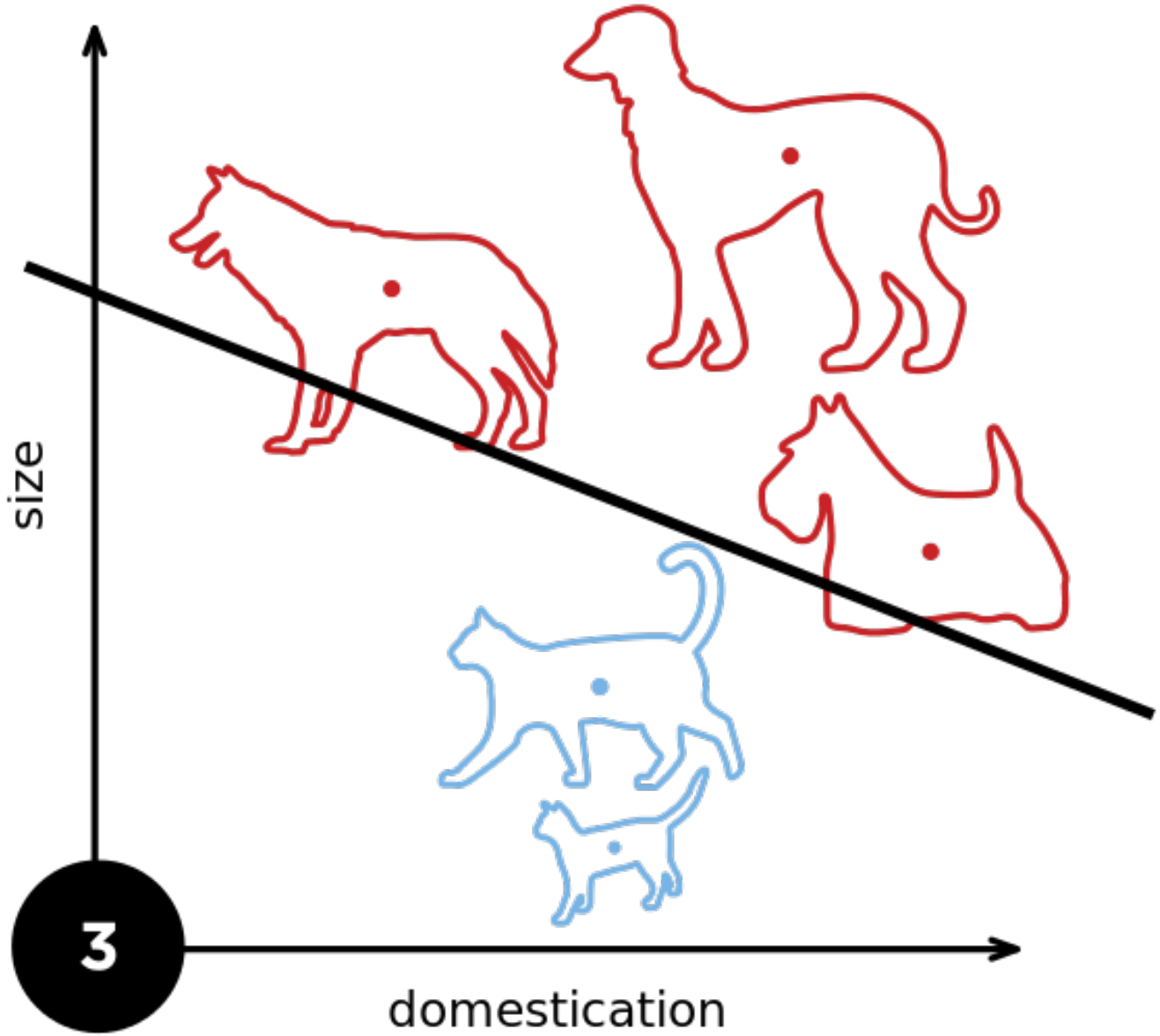


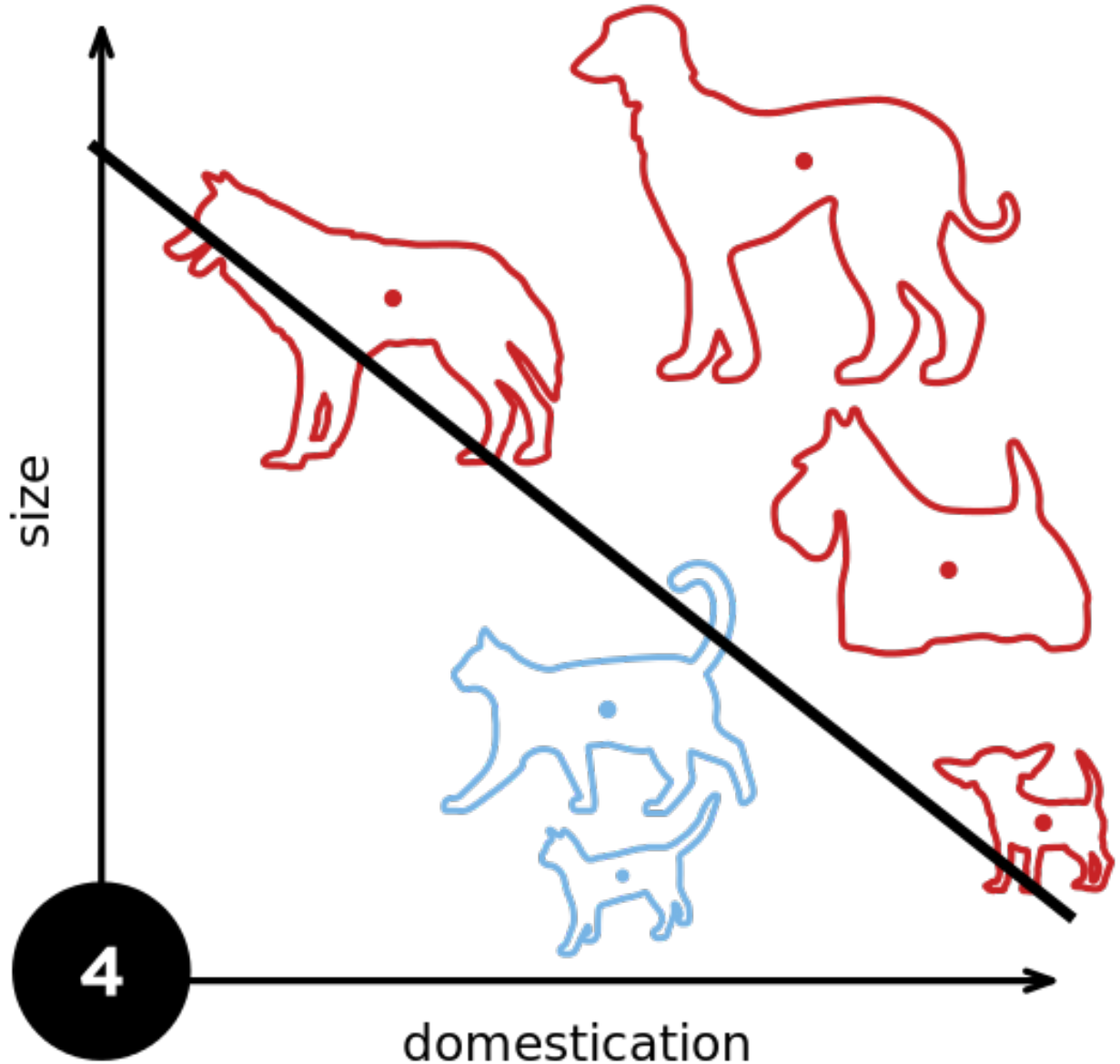
Supervised Learning

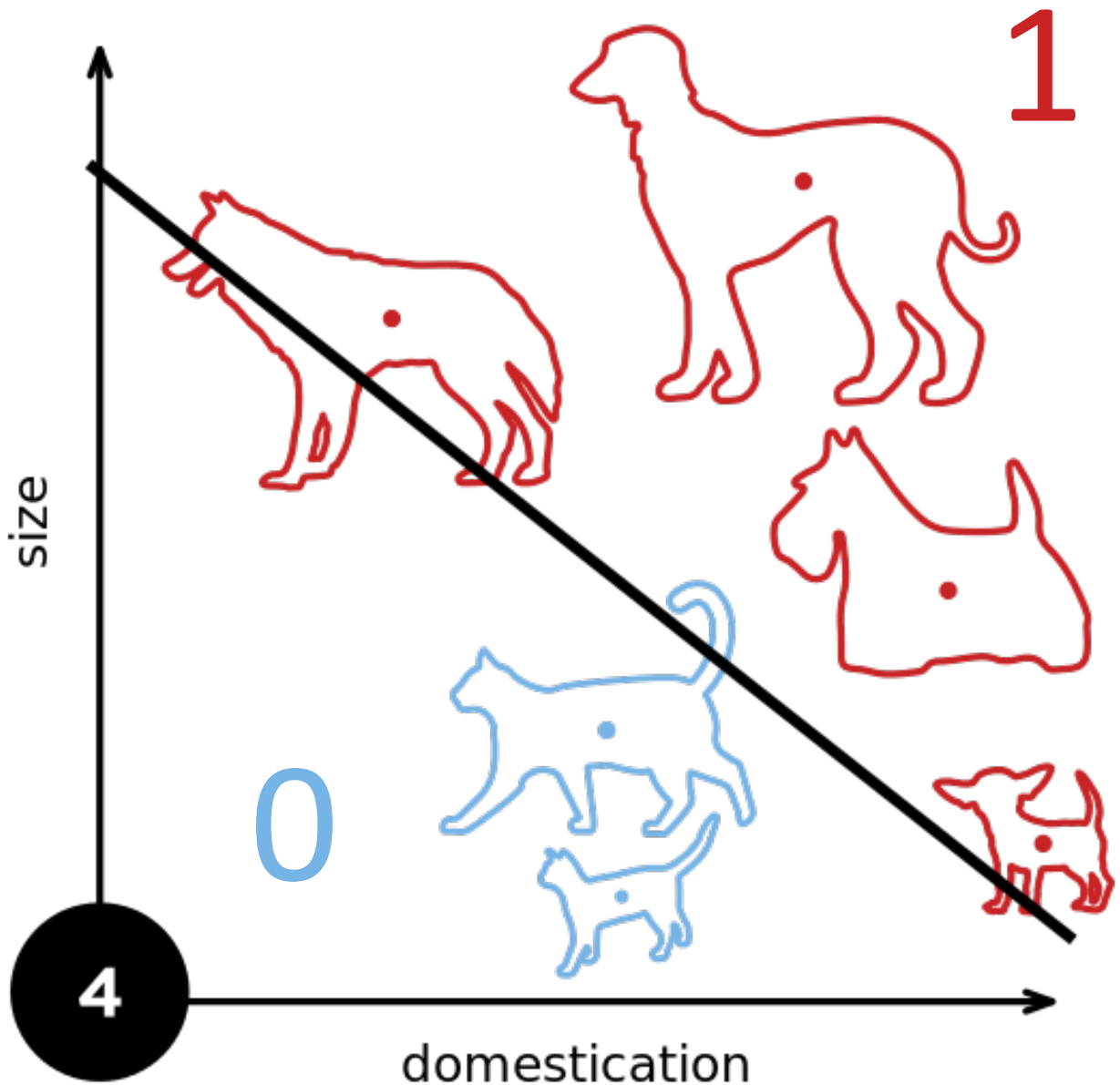



size
1
domestication

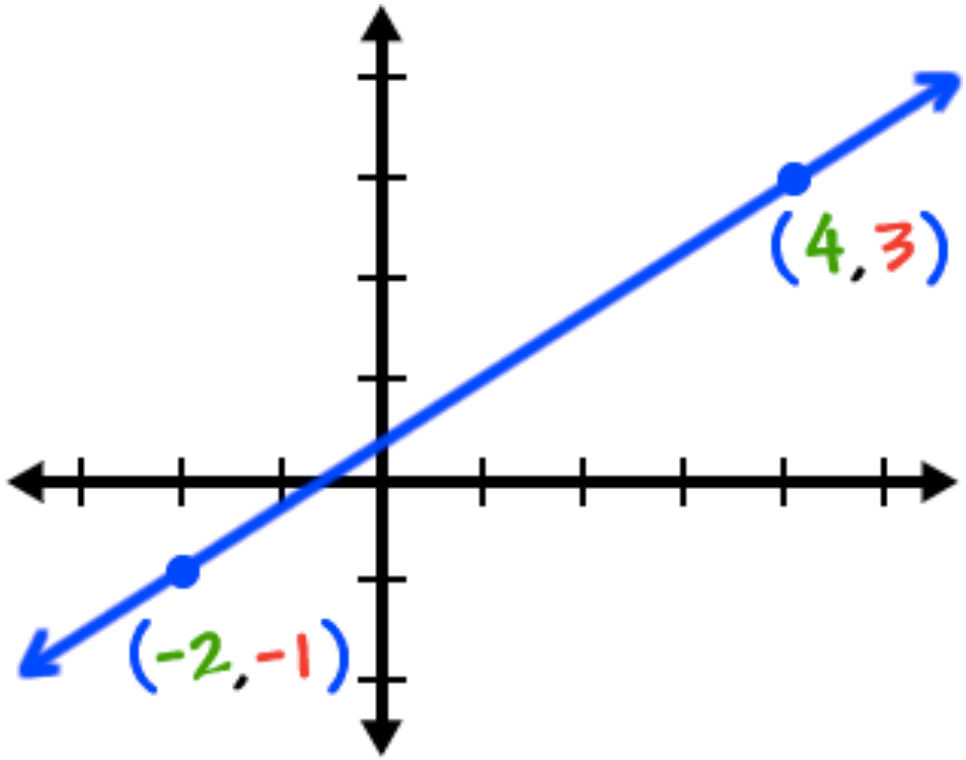

size
2
domestication


size
3
domestication

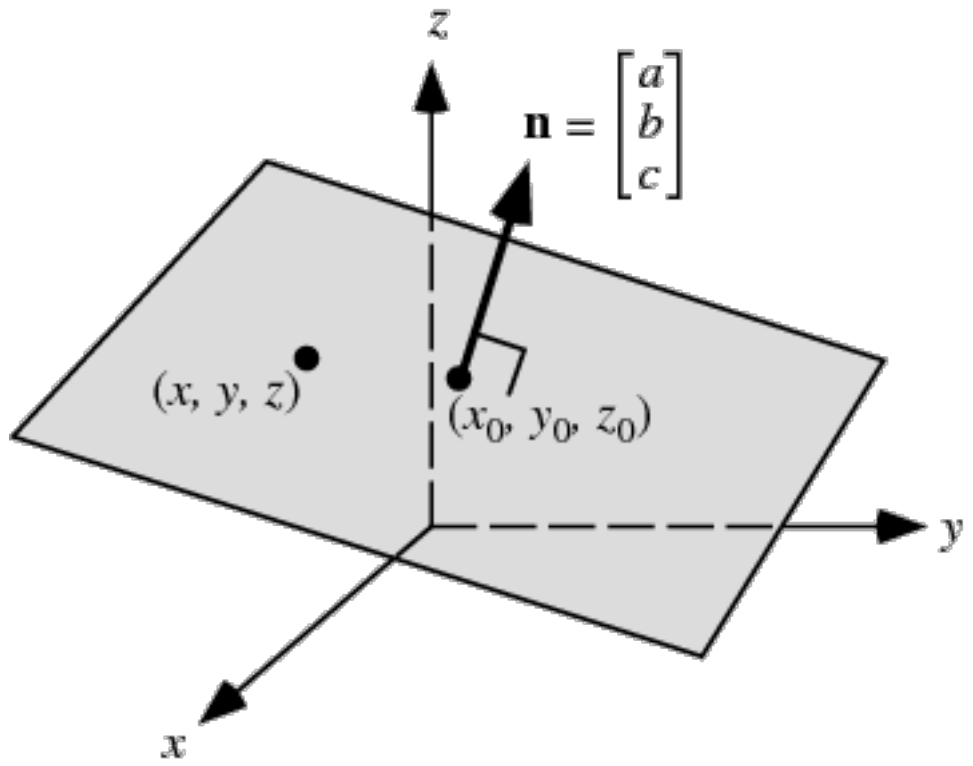

size
4
domestication


1
0
size
4
domestication

$$y = \underline{m}x + \underline{b}$$



$$z = \underline{m}x + \underline{n}y + \underline{b}$$



Cool learning algorithm

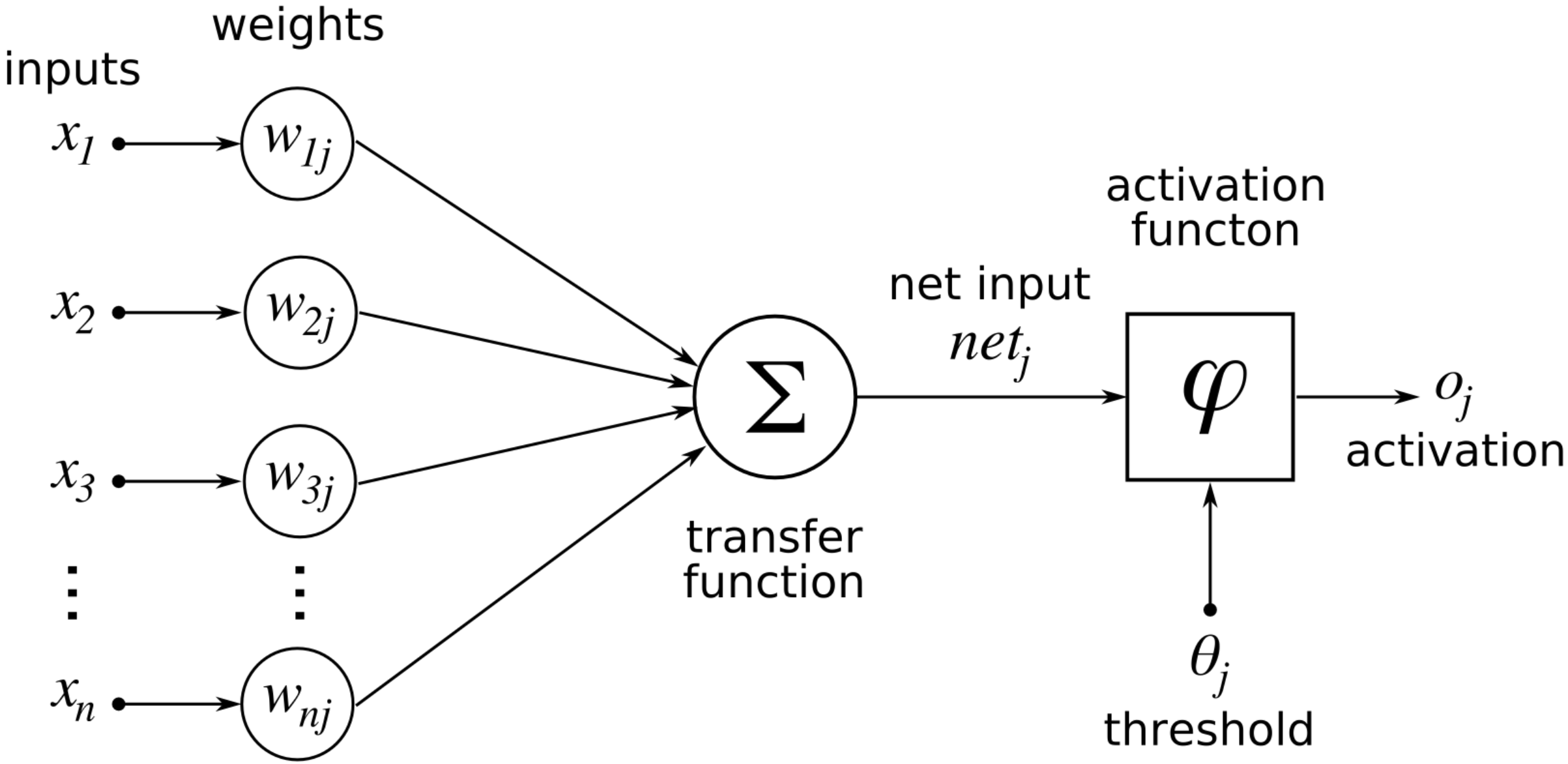
```
def classify(datapoint, weights):
```

Cool learning algorithm

```
def classify(datapoint, weights):
    prediction = sum(x * y for x, y in
      zip([1] + datapoint, weights))
```

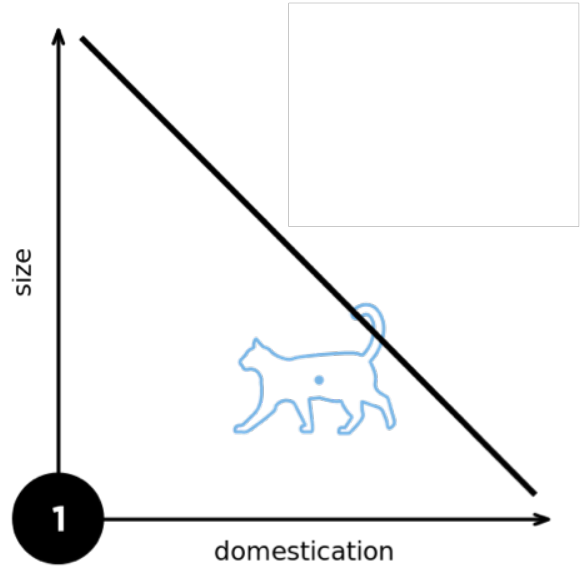
Cool learning algorithm

```
def classify(datapoint, weights):
    prediction = sum(x * y for x, y in
        zip([1] + datapoint, weights))
    if prediction < 0:
        return 0
    else:
        return 1
```


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

Cool learning algorithm

```
def classify(datapoint, weights):
    prediction = sum(x * y for x, y in
        zip([1] + datapoint, weights))
    if prediction < 0:
        return 0
    else:
        return 1
```

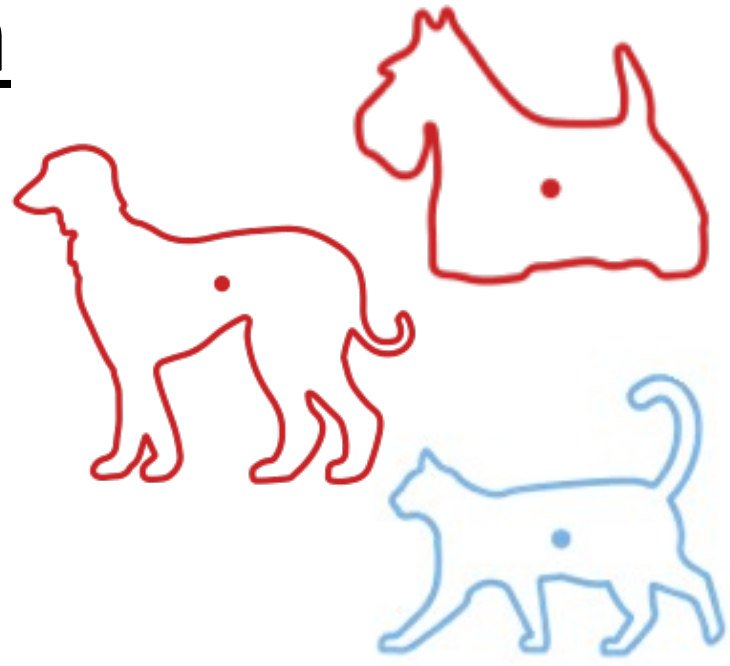


Cool learning algorithm

```
def train(data_set):
```

Cool learning algorithm

```
def train(data_set):
```



```
class Datum:
    def  init (self, features, label):
        self.features = [1] + features
        self.label = label
```

Cool learning algorithm

```
def train(data_set):
    weights = [0] * len(data_set[0].features)
```

[0, 0, 0]

Cool learning algorithm

```
def train(data_set):
    weights = [0] * len(data_set[0].features)
    total_error = threshold + 1
```

Cool learning algorithm

```
def train(data_set):
    weights = [0] * len(data_set[0].features)
    total_error = threshold + 1
    while total_error > threshold:
        total_error = 0
        for item in data_set:
            error = item.label - classify(item.features, weights)
            weights = [w + RATE * error * i
                       for w, i in zip(weights, item.features)]
            total_error += abs(error)
```

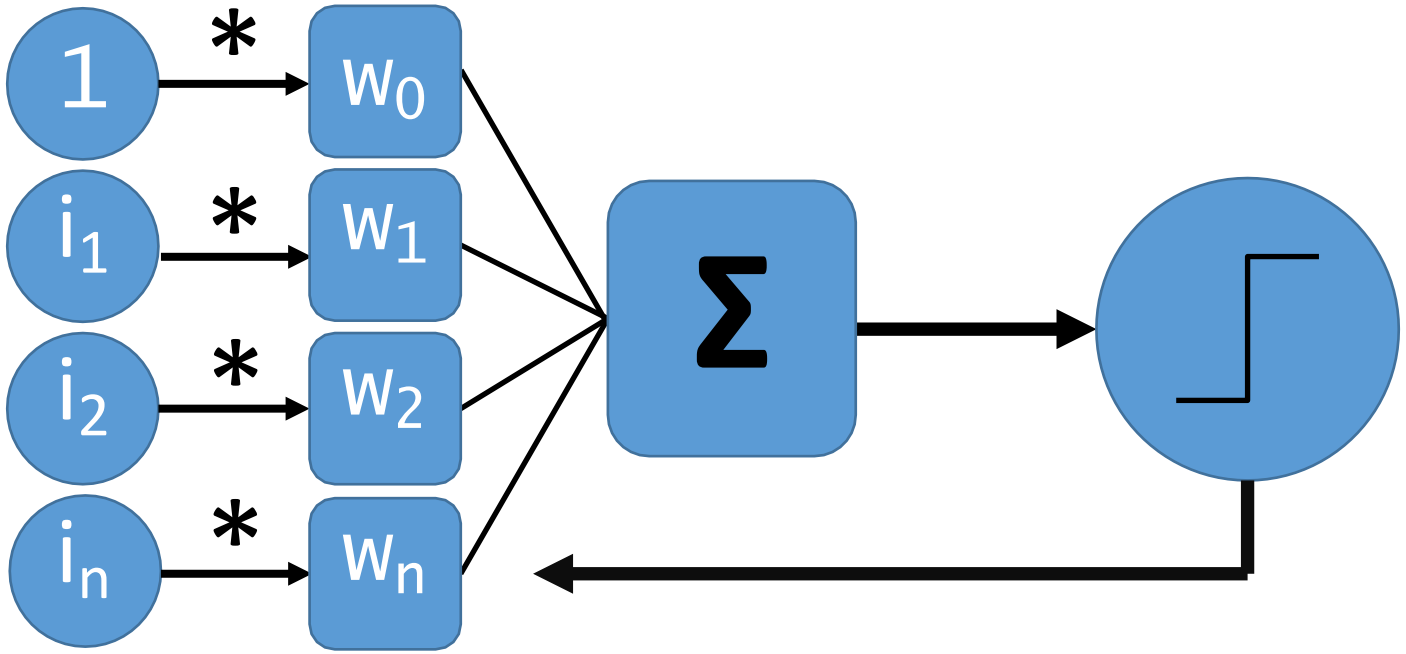
Cool learning algorithm

```
def train(data_set):
    weights = [0] * len(data_set[0].features)
    total_error = threshold + 1
    while total_error > threshold:
        total_error = 0
        for item in data_set:
            error = item.label - classify(item.features, weights)
            weights = [w + RATE * error * i
                       for w, i in zip(weights, item.features)]
            total_error += abs(error)
```

Cool learning algorithm

```
def train(data_set):
    weights = [0] * len(data_set[0].features)
    total_error = threshold + 1
    while total_error > threshold:
        total_error = 0
        for item in data_set:
            error = item.label - classify(item.features, weights)
            weights = [w + RATE * error * i
                       for w, i in zip(weights, item.features)]
            total_error += abs(error)
```

Cool learning algorithm



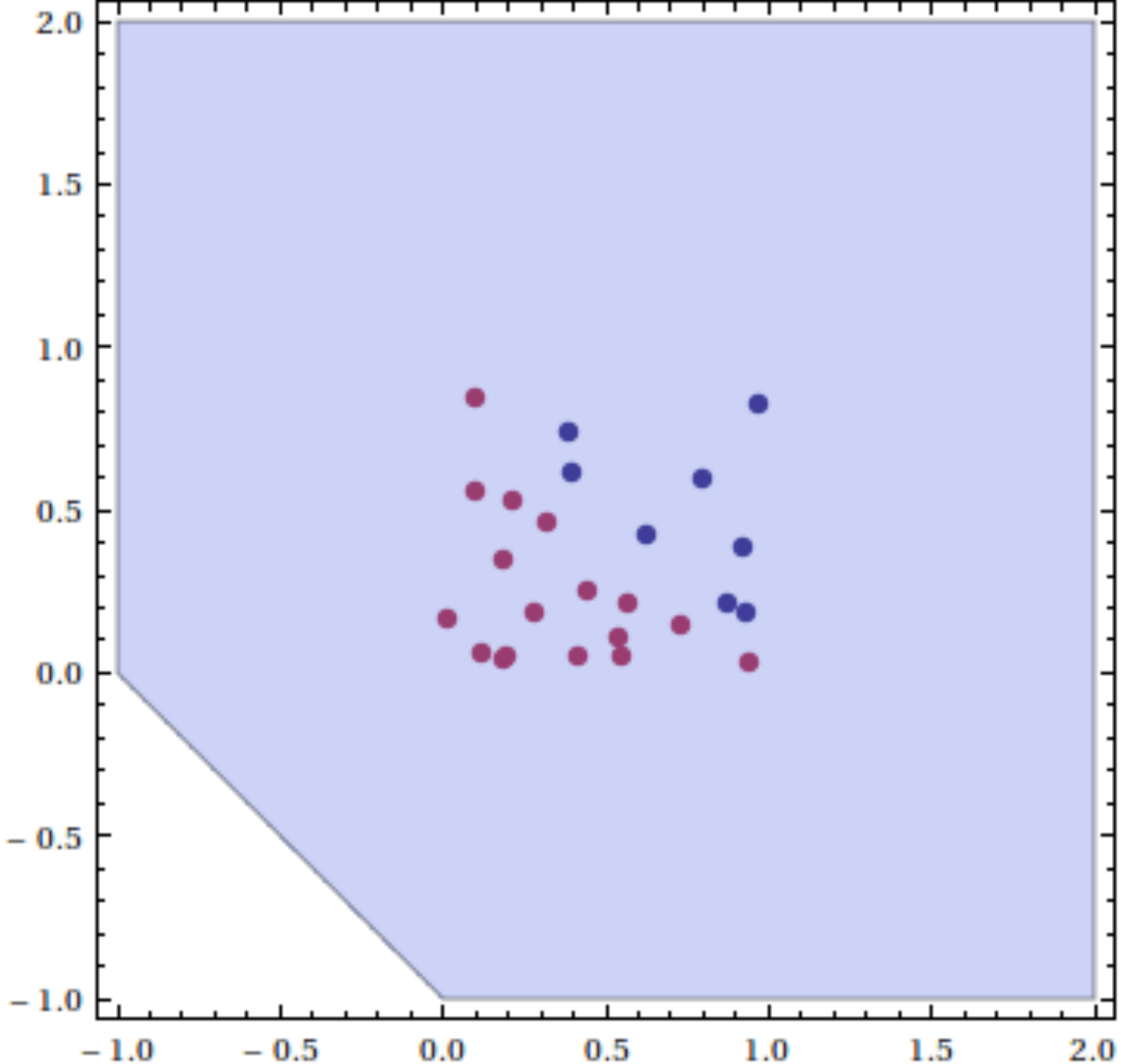

```
weights = [w + RATE * error * i
           for w, i in zip(weights, item.features)]
```

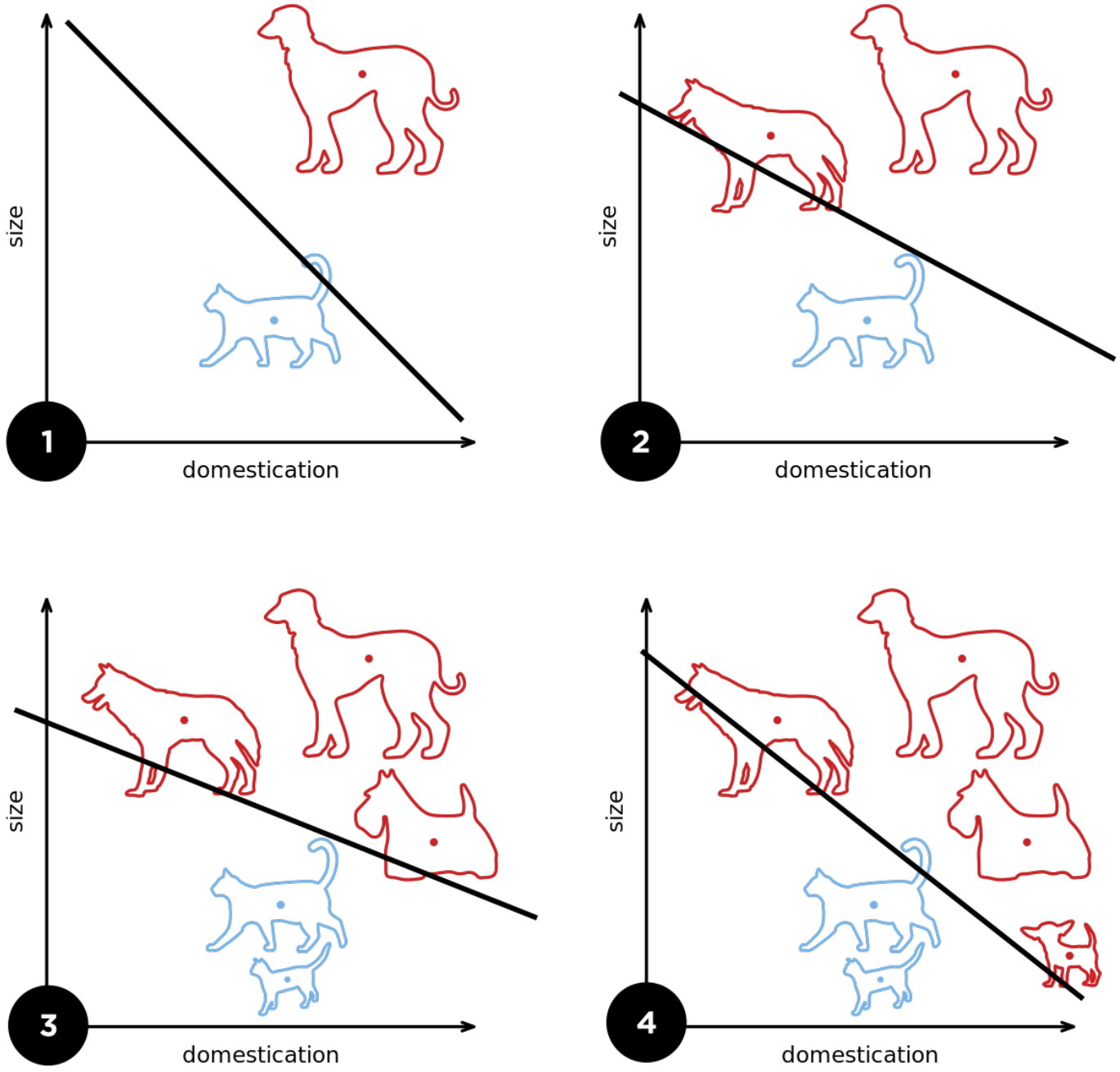
Cool learning algorithm

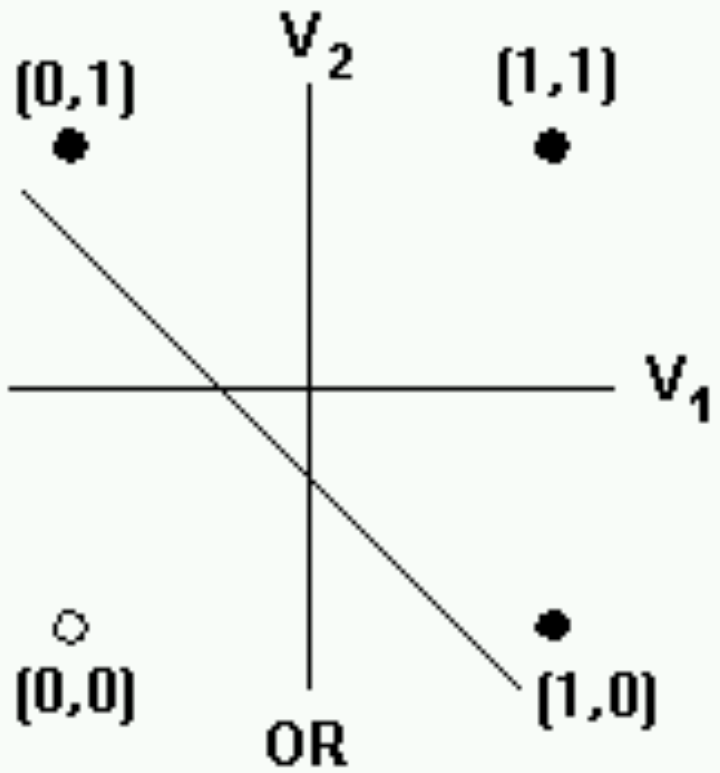
```
def train(data_set):
    weights = [0] * len(data_set[0].features)
    total_error = threshold + 1
    while total_error > threshold:
        total_error = 0
        for item in data_set:
            error = item.label - classify(item.features, weights)
            weights = [w + RATE * error * i
                       for w, i in zip(weights, item.features)]
            total_error += abs(error)
```

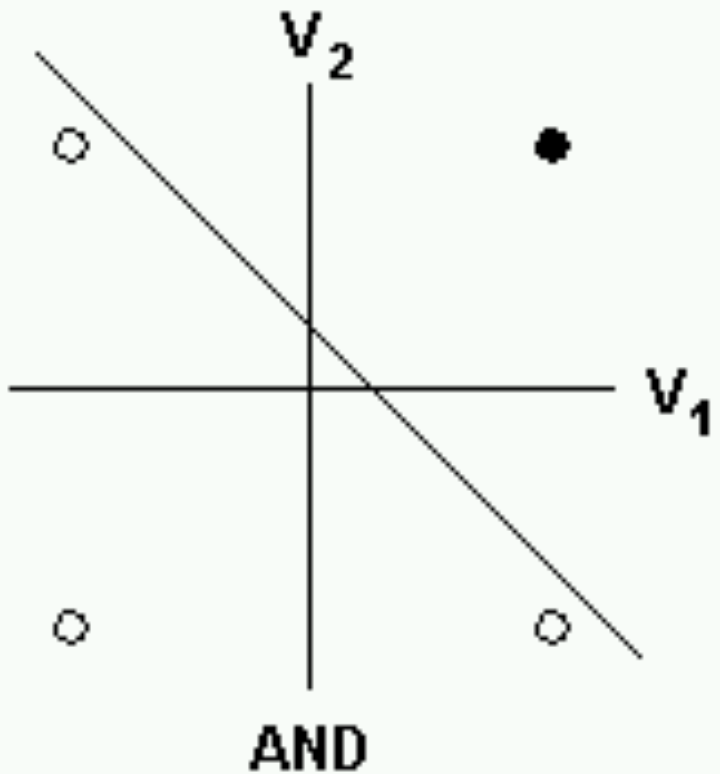
Cool learning algorithm

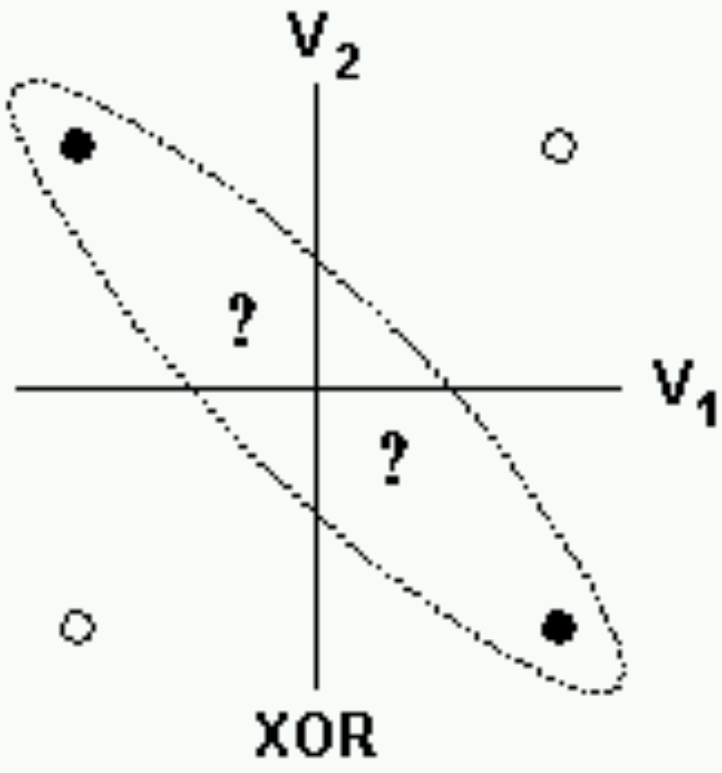
```
def train(data_set):
  weights = [0] * len(data_set[0].features)
  total_error = threshold + 1
  while total_error > threshold:
    total_error = 0
    for item in data_set:
      error = item.label - classify(item.features, weights)
      weights = [w + RATE * error * i
                 for w, i in zip(weights, item.features)]
      total_error += abs(error)
```


2.0
1.5
1.0
0.5
0.0
– 0.5
– 1.0
– 1.0
– 0.5
0.0
0.5
1.0
1.5
2.0

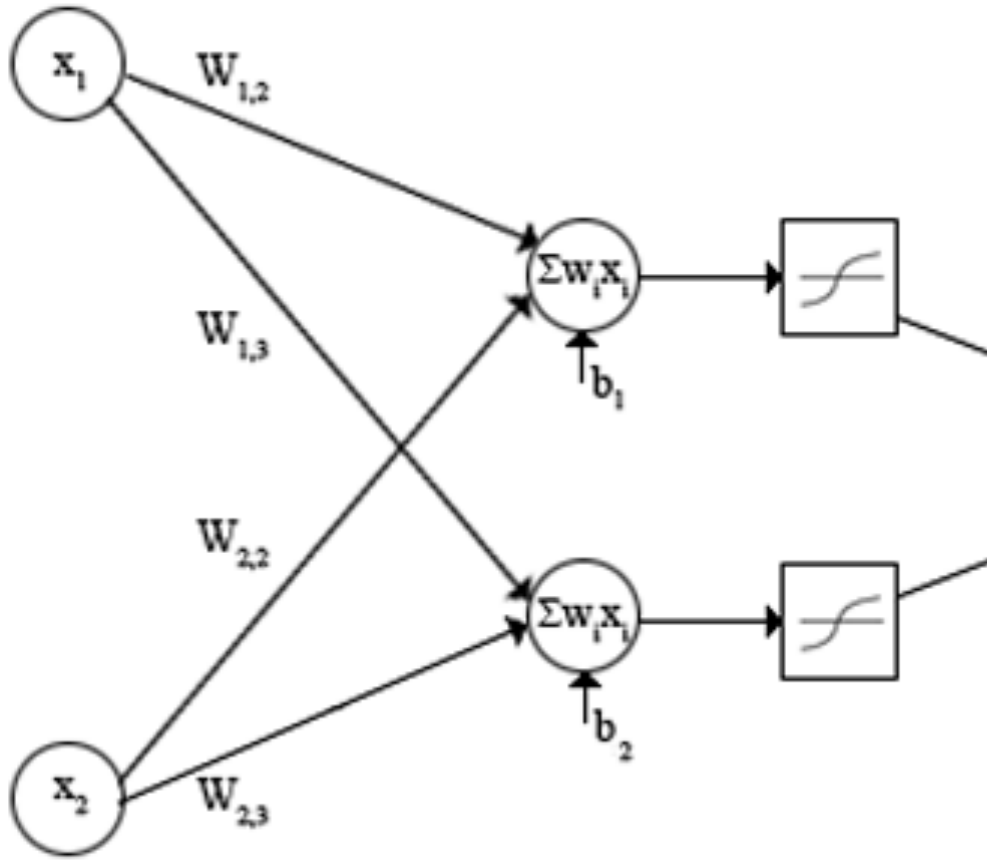

size
1
domestication
size
2
domestication
size
3
domestication
size
4
domestication

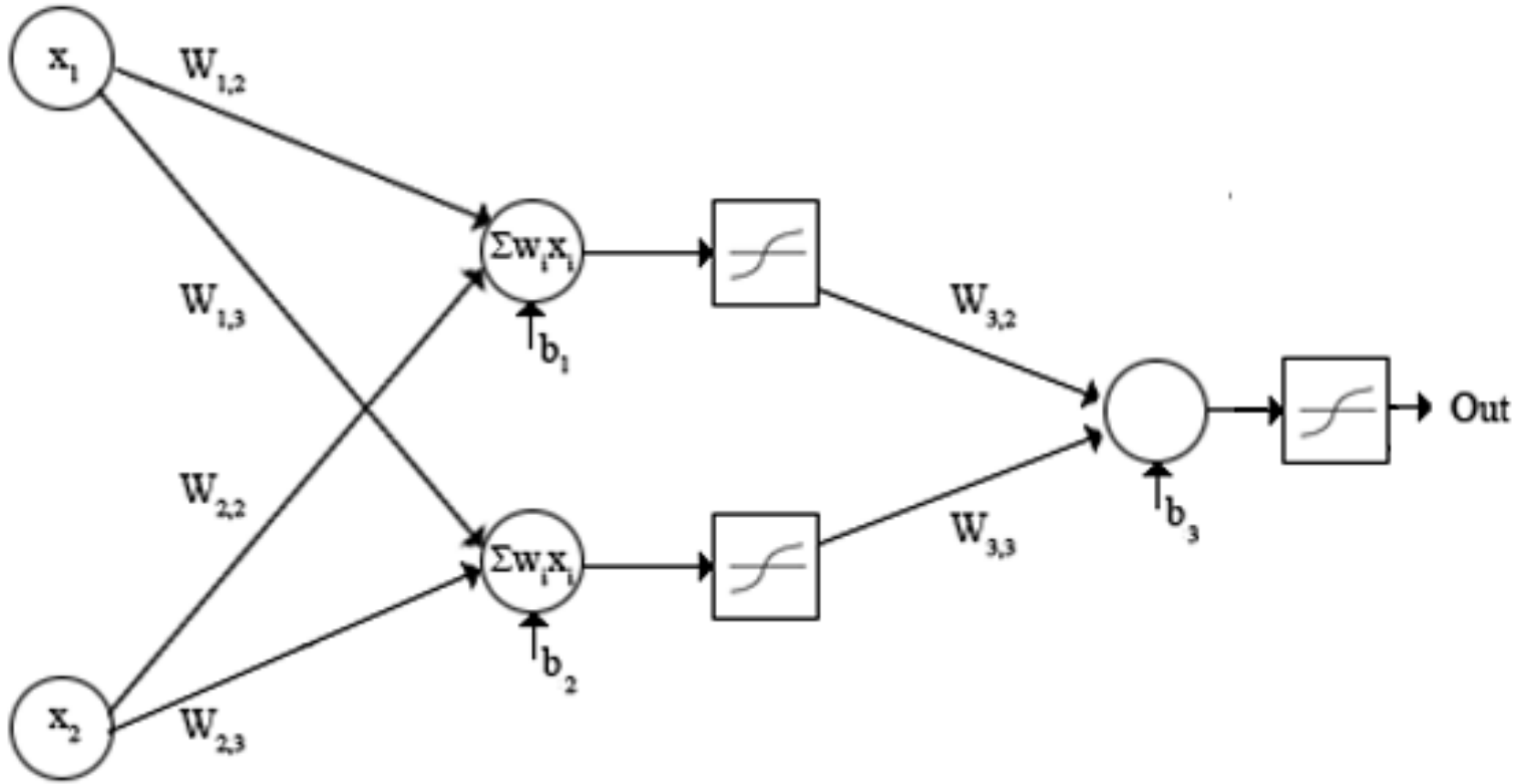

(0,1)
V2
(1,1)
V1
(0,0)
(1,0)
OR

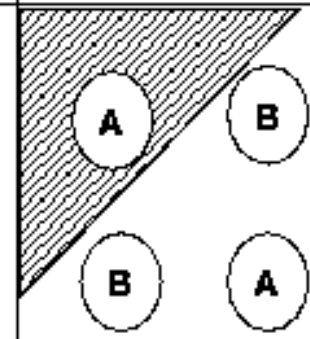
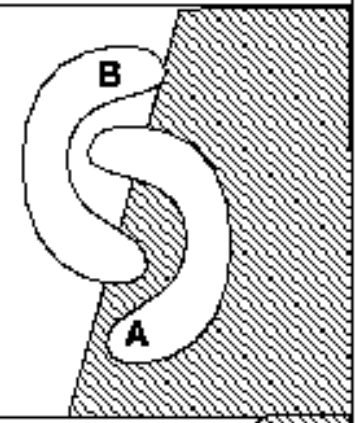
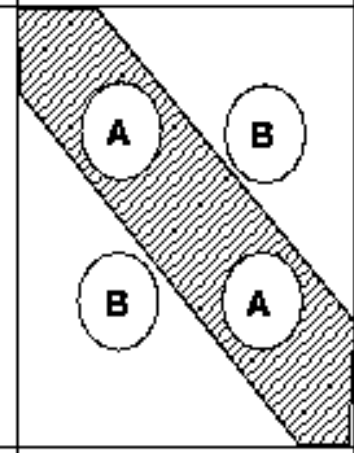
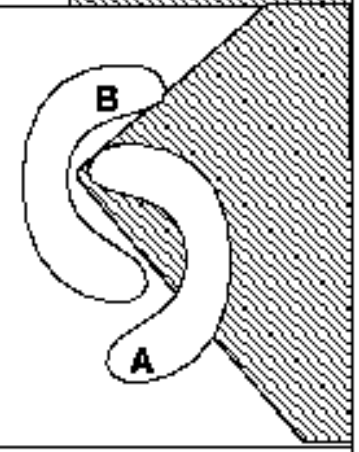
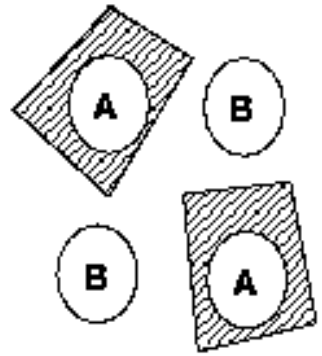
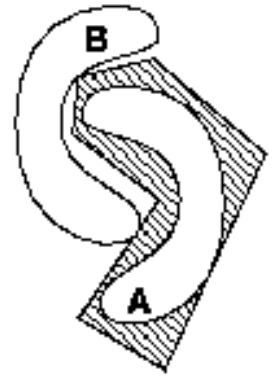

V2
V1
AND


V2
?
?
V1
XOR


x_1
$W_{1,2}$
$W_{1,3}$
$W_{2,2}$
x_2
$W_{2,3}$


x1
W1,2
W1,3
Σwixi
b1
W2,2
Σwixi
b2
x2
W2,3

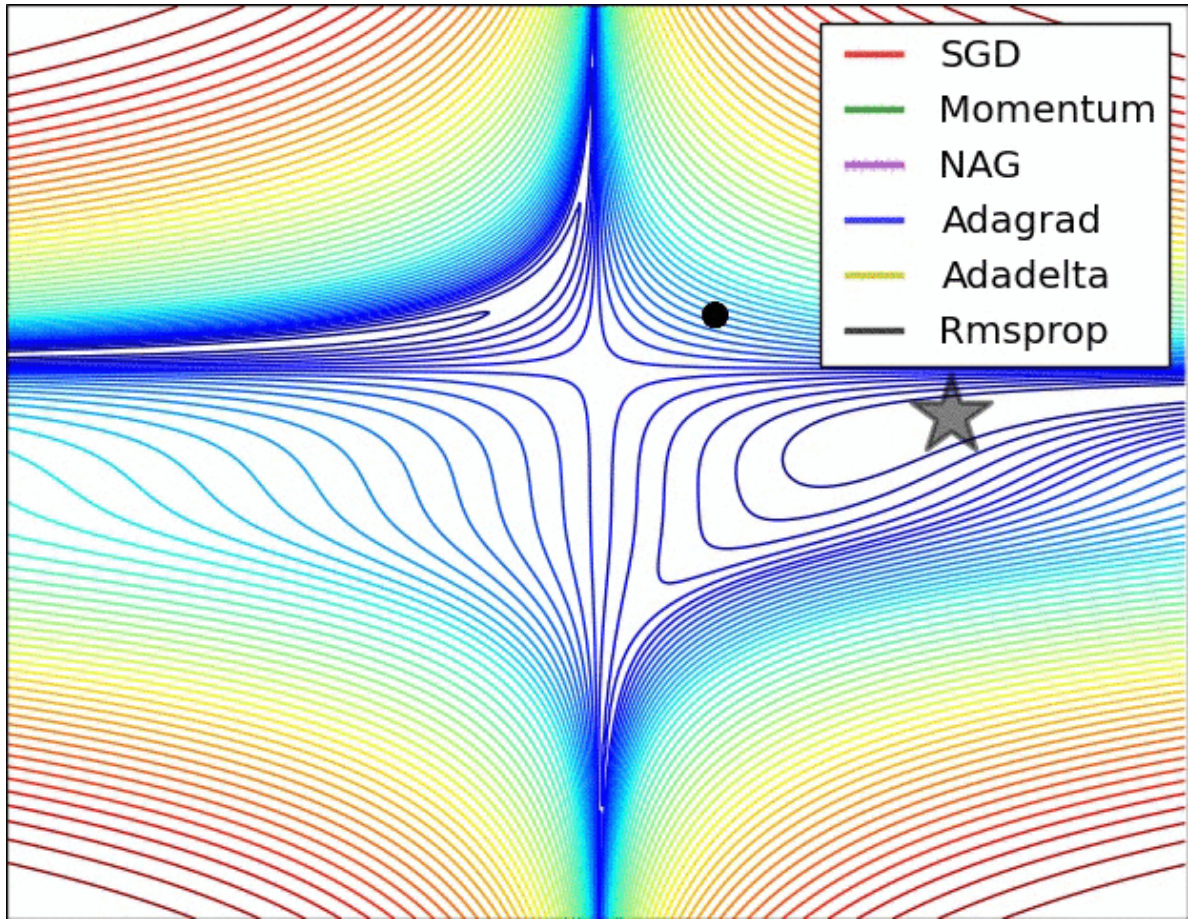

x1
x2
W1,2
W1,3
W2,2
W2,3
Σwixi
Σwixi
b1
b2
W3,2
W3,3
b3
Out

Structure	Regions	XOR	Meshed regions
single layer 	Half plane bounded by hyper-plane		
two layer 	Convex open or closed regions		
three layer 	Arbitrary (limited by # of nodes)		

Backpropogation

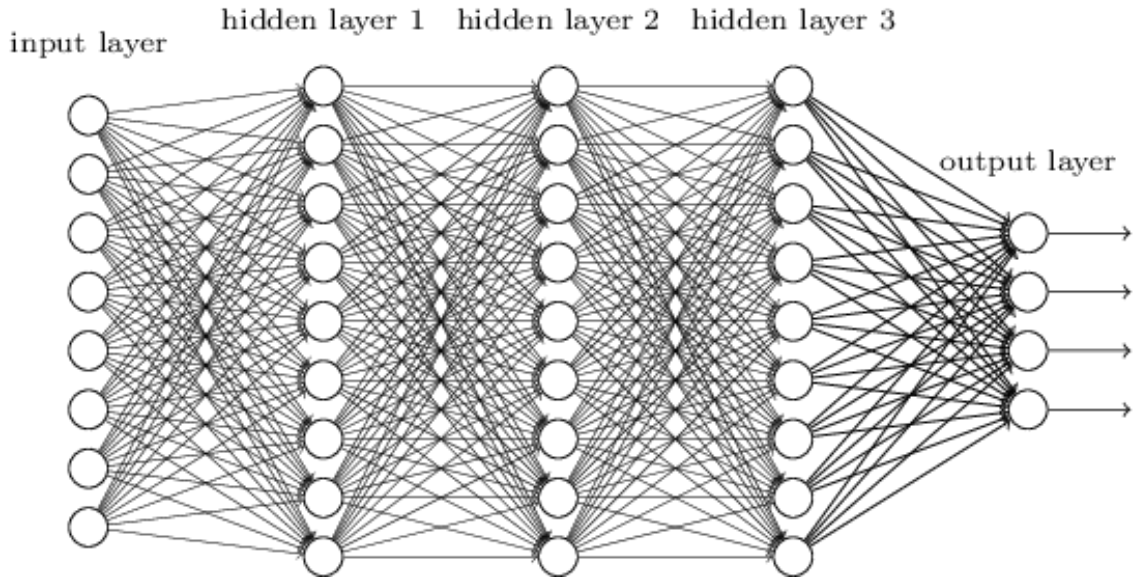
```
train(trainingSet) :
  initialize network weights randomly
  until average error stops decreasing (or you get tired):
    for each sample in trainingSet:
      prediction = network.output(sample)
      compute error (prediction – sample.output)
      compute error of (hidden -> output) layer weights
      compute error of (input -> hidden) layer weights
      update weights across the network
  save the weights
```

Gradient Descent



http://cs231n.github.io/

“Deep” neural networks



Artificial Neural Networks: How can I estimate the number of neurons and layers?

Yoshua Bengio, Head of Montreal Institute for Learning Algorithms, Professor @ U. Montreal

Very simple. Just keep adding layers until the test error does not improve anymore.

As for number of units, it is a hyper-parameter to be optimized, as usual. See my paper on guidelines for setting up hyper-parameters in deep networks, *Practical recommendations for gradient-based training of deep architectures*.

Written May 7, 2013 • View Upvotes

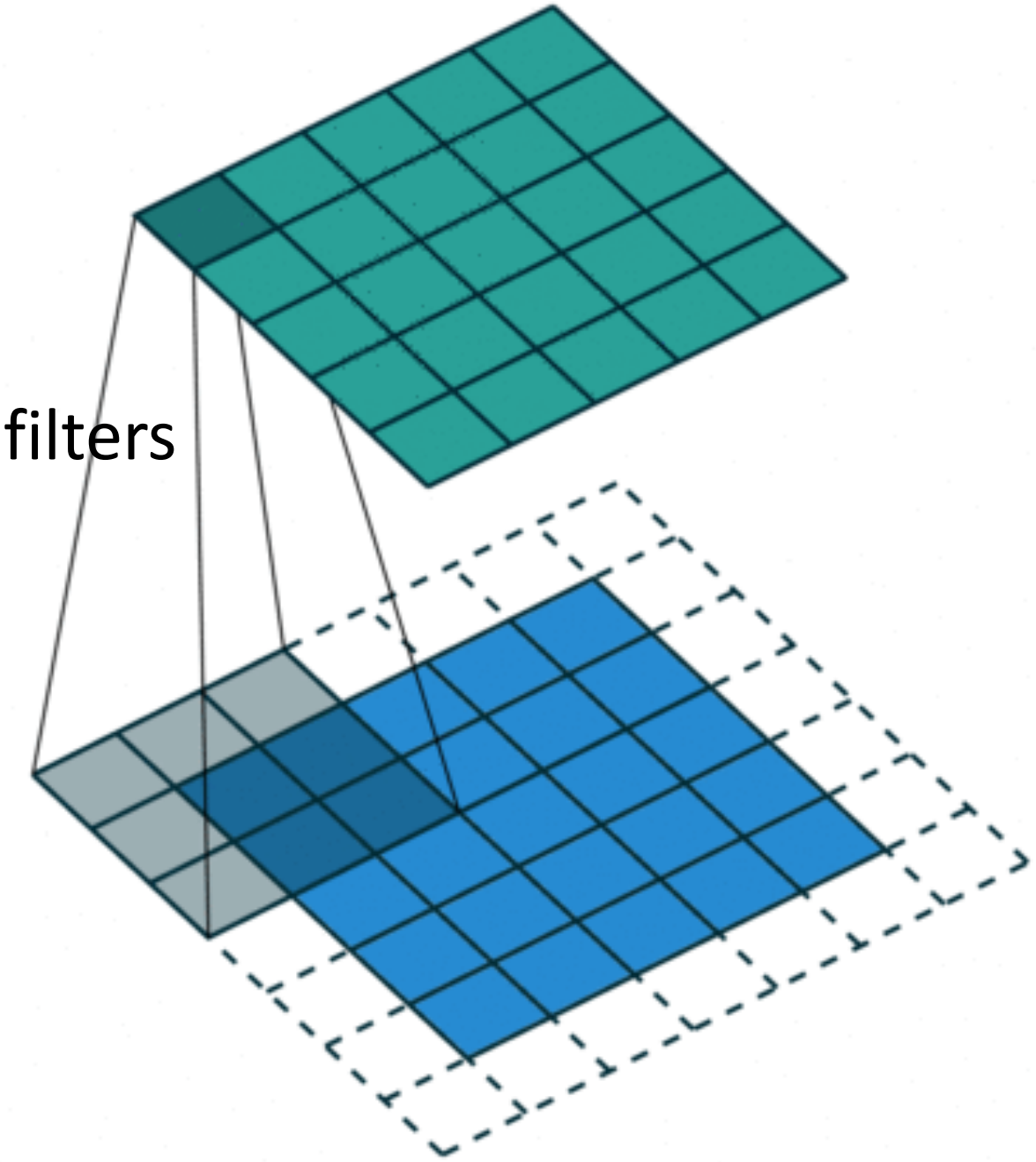
ImageNet LSVR Competition

What is a kernel?

- A kernel is just a matrix
- Used for edge detection, blurs, filters

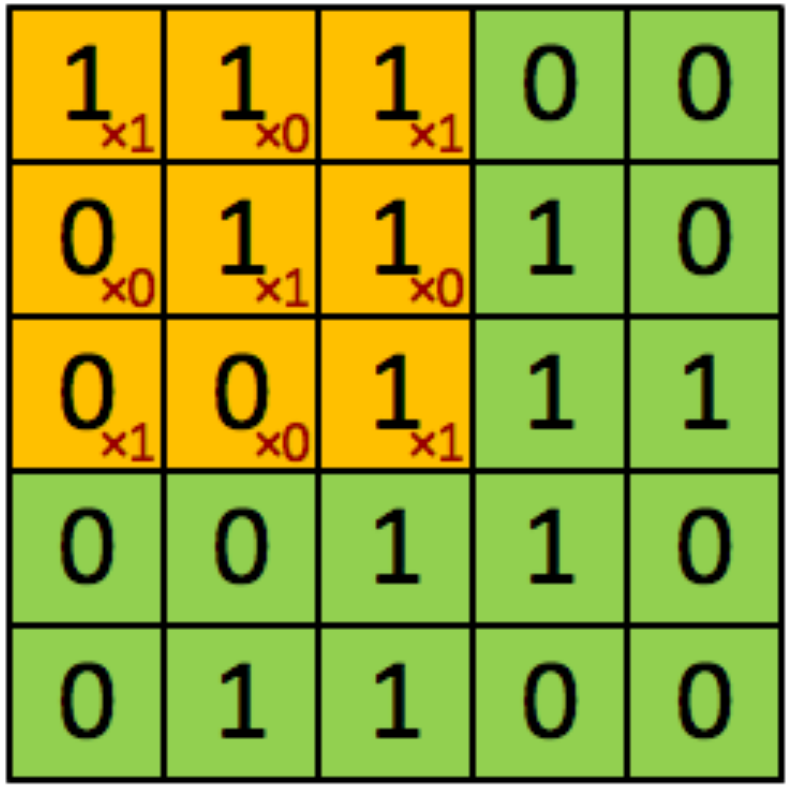
$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix} \quad \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

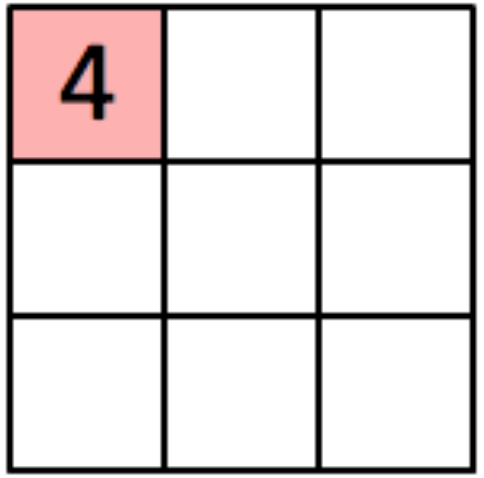
$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix} \quad \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \quad \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

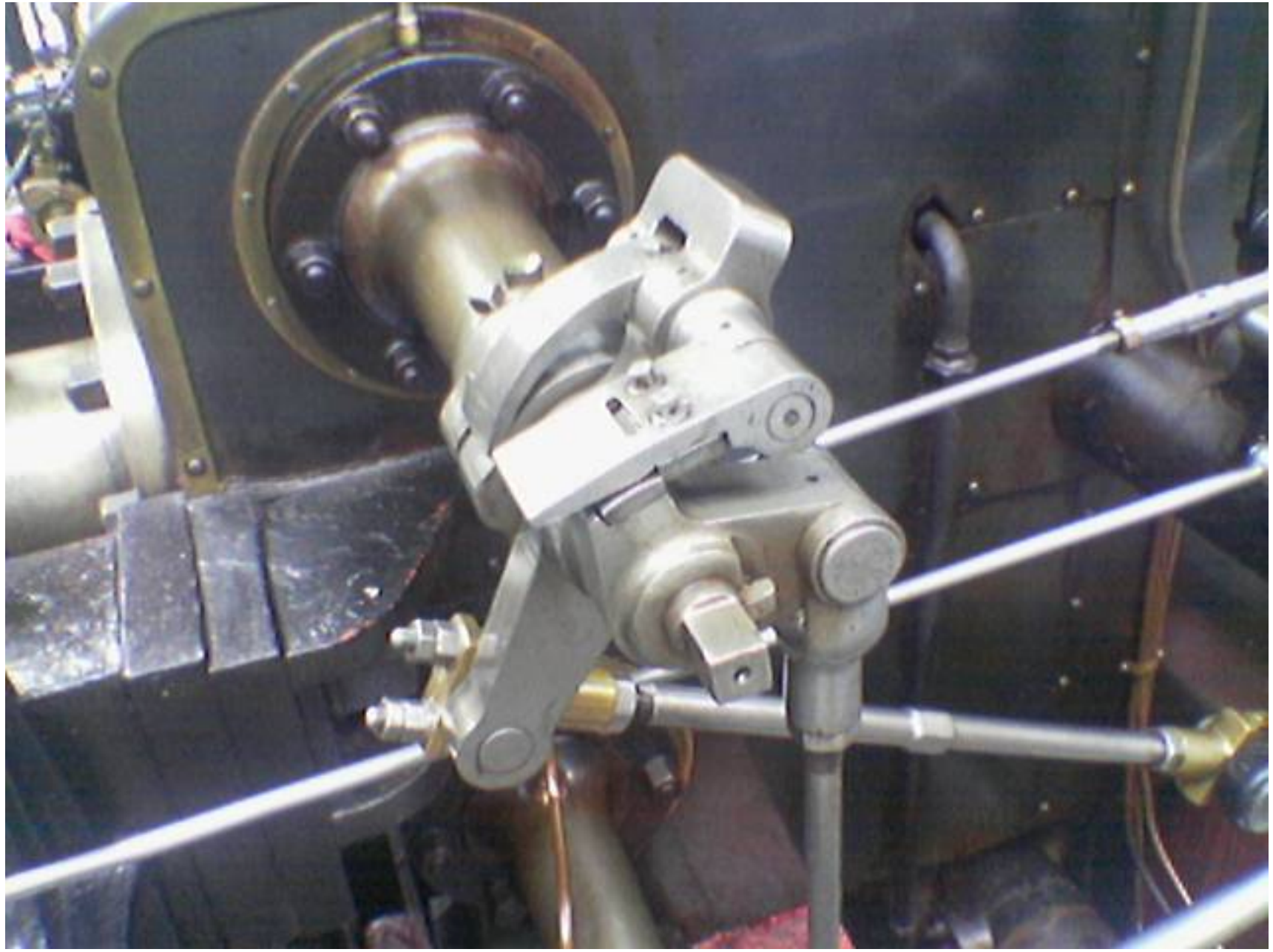


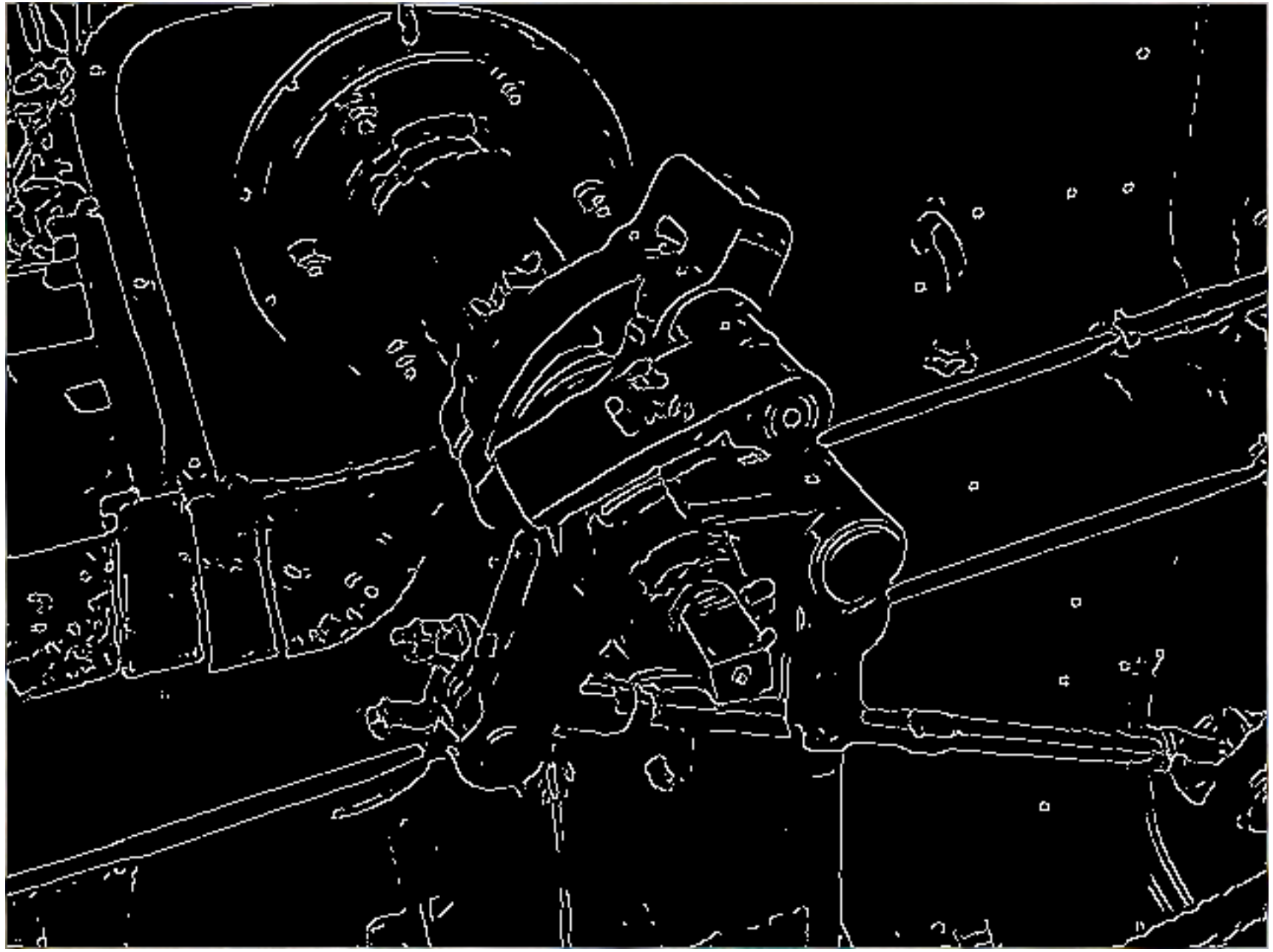
Image

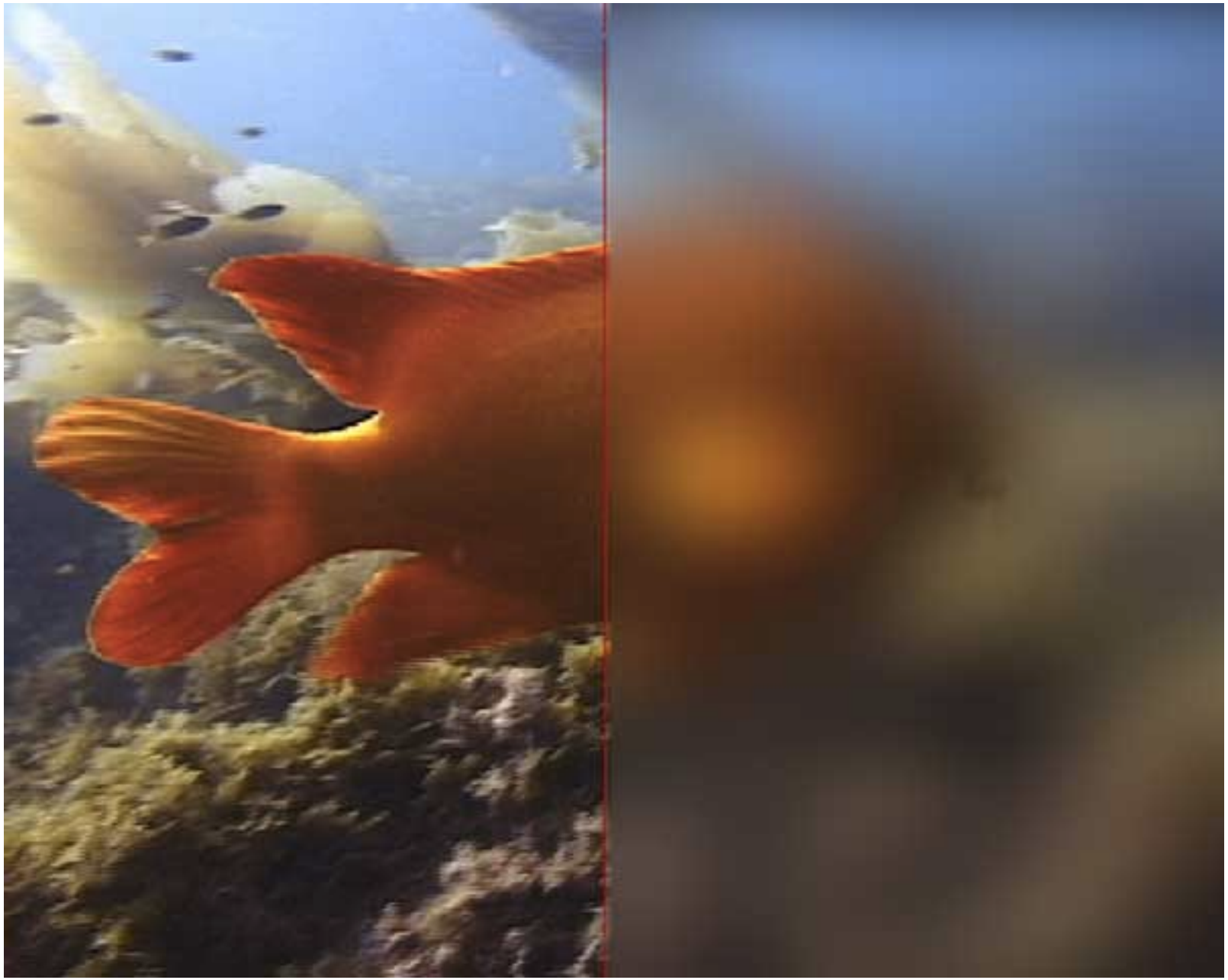
Convolved Feature

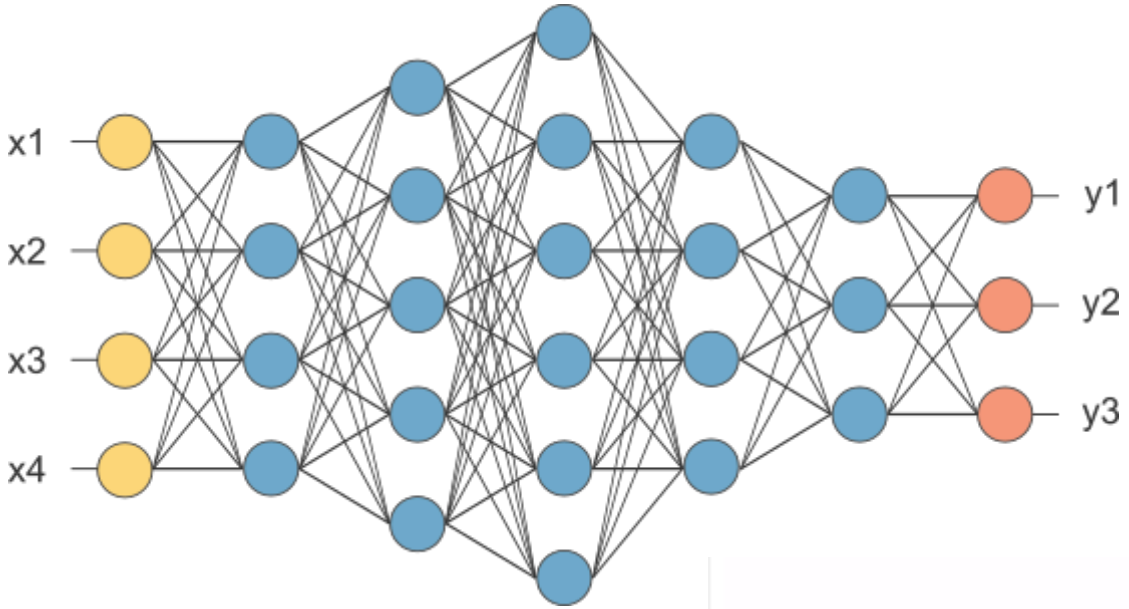


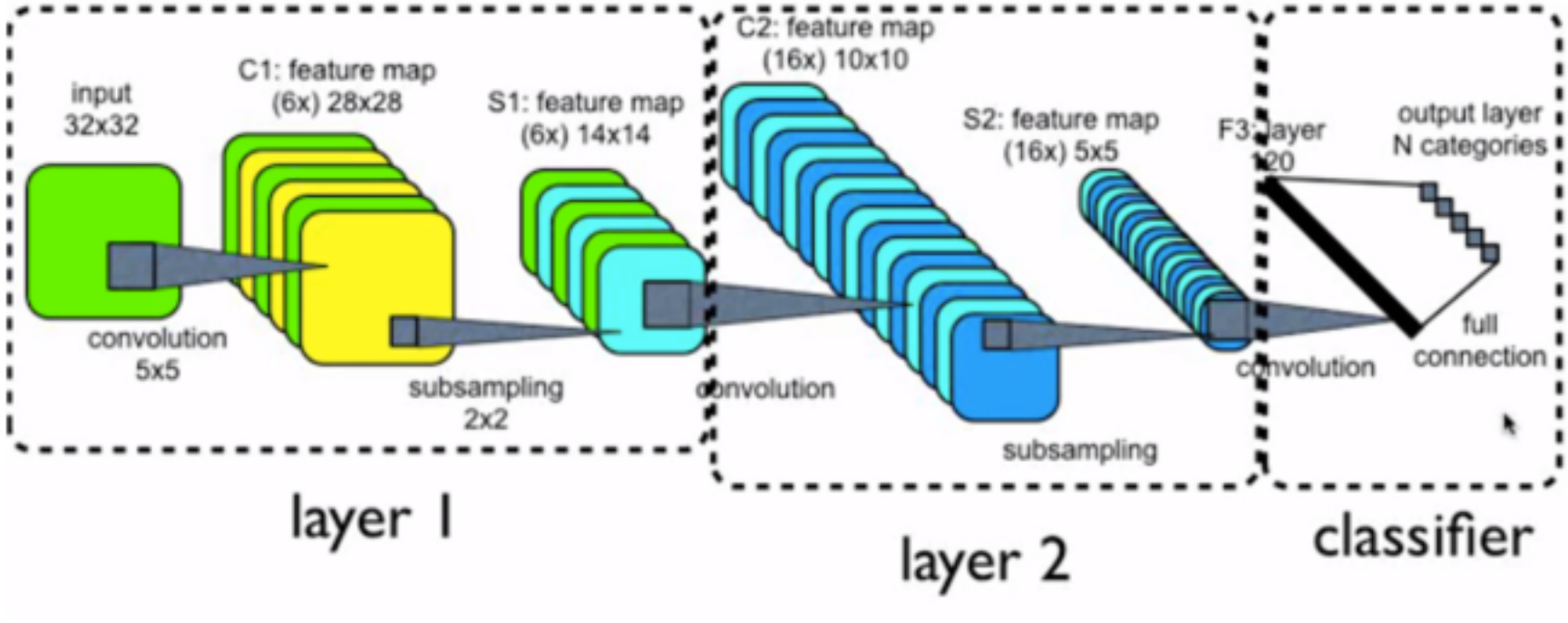




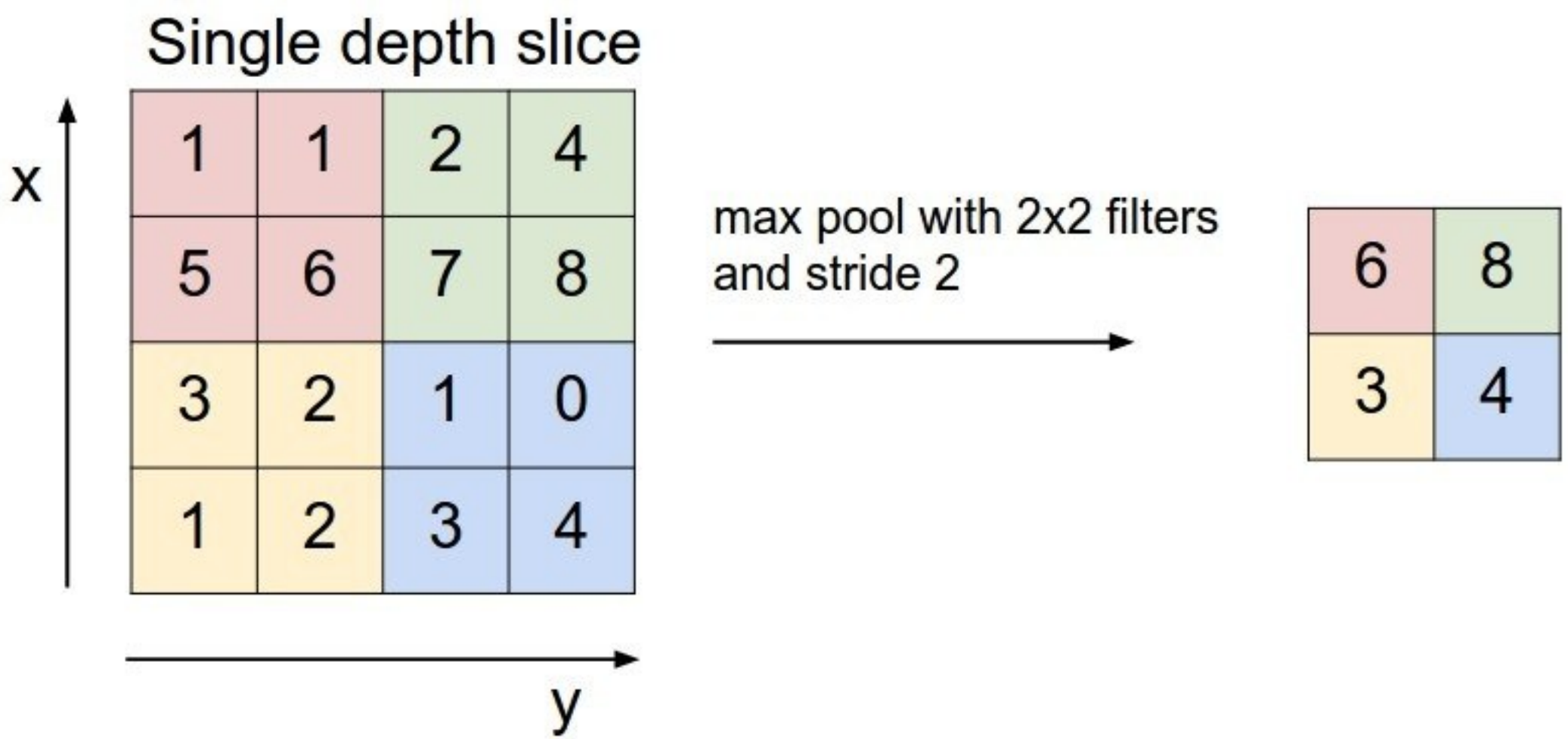




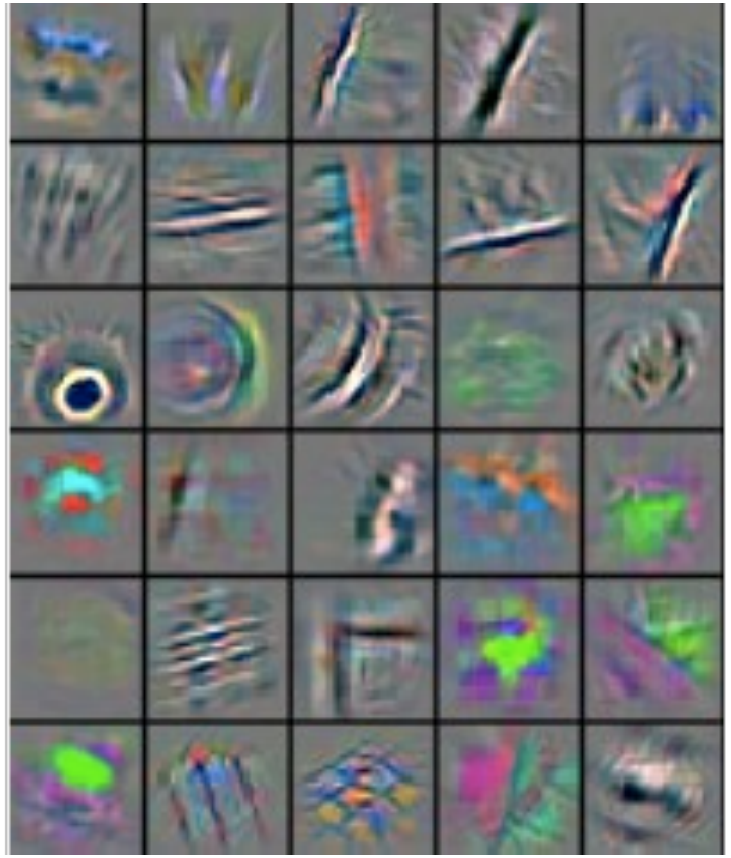

x1
x2
x3
x4
y1
y2
y3

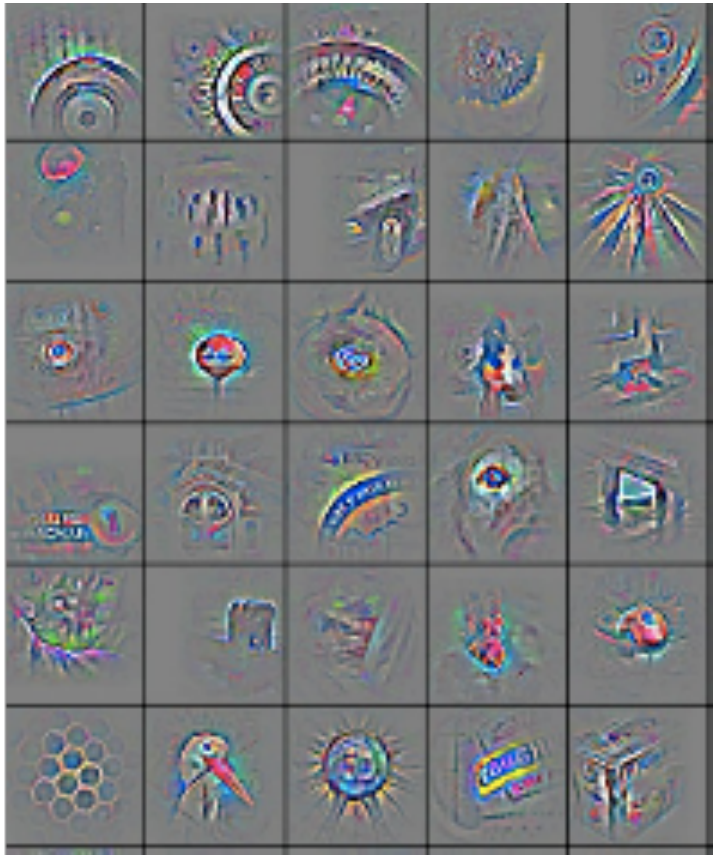

input
32x32
C1: feature map
(6x) 28x28
S1: feature map
(6x) 14x14
C2: feature map
(16x) 10x10
S2: feature map
(16x) 5x5
F3: layer
120
output layer
N categories
convolution
5x5
subsampling
2x2
convolution
subsampling
convolution
full
connection
layer 1
layer 2
classifier

Pooling (Downsampling)

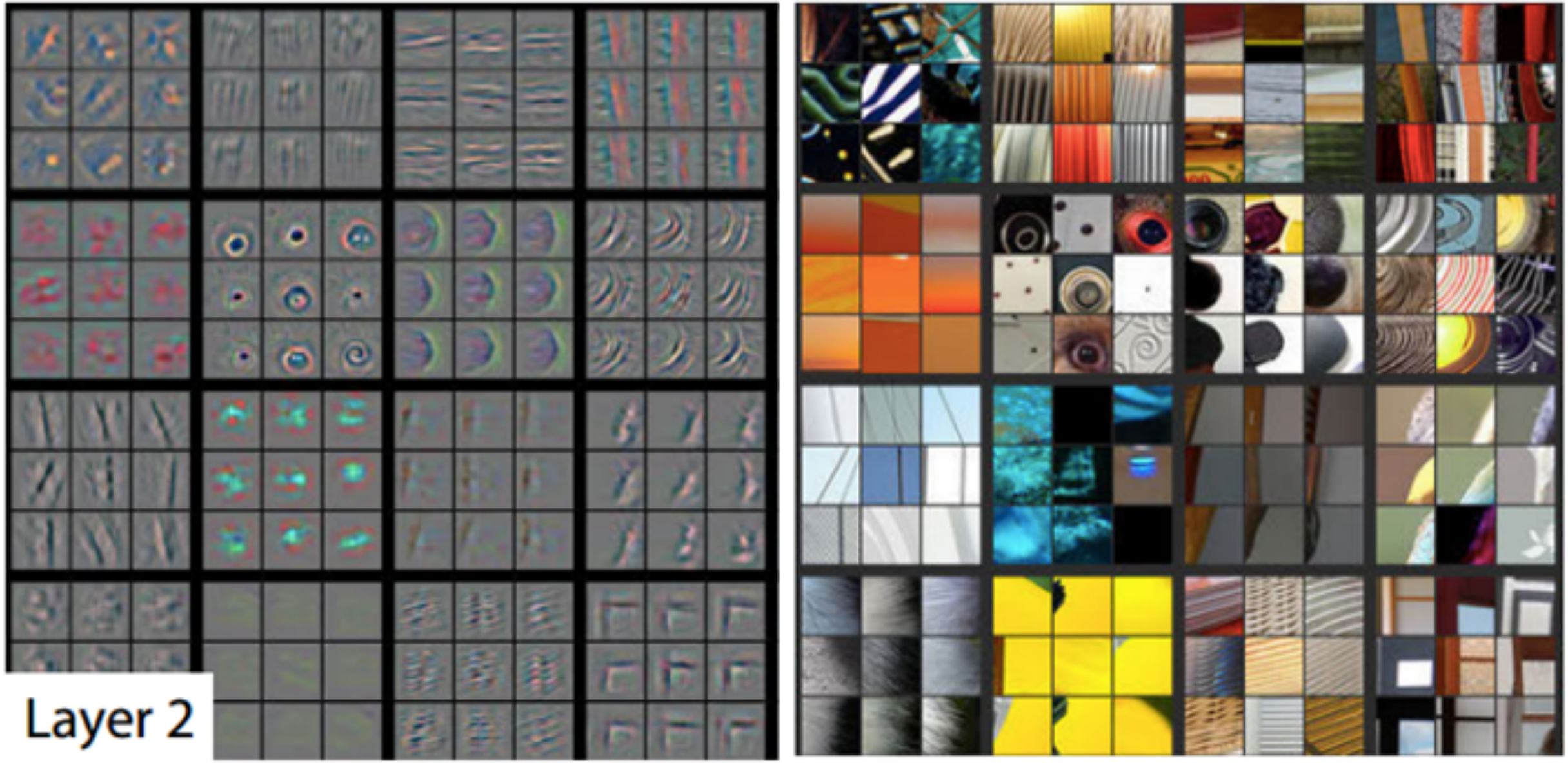


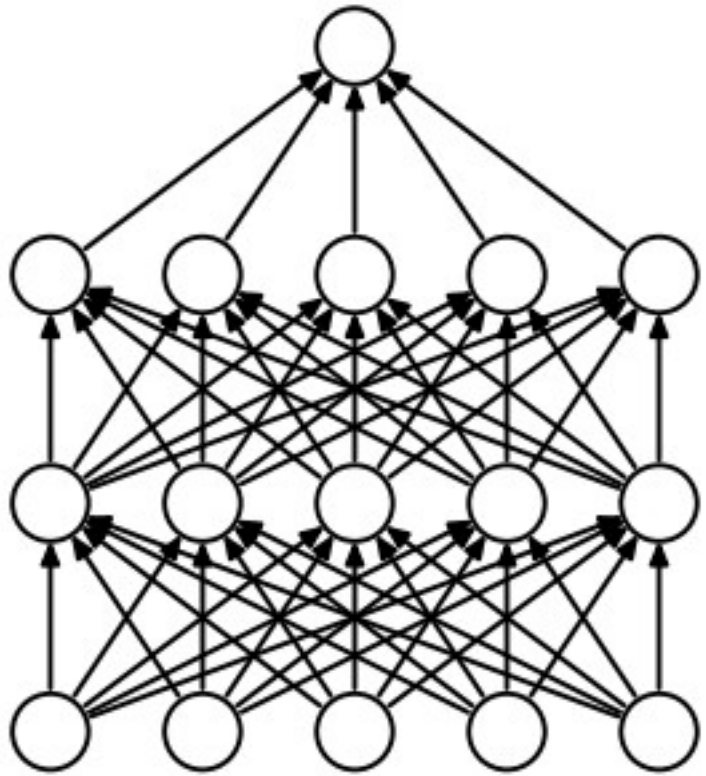
Low level features



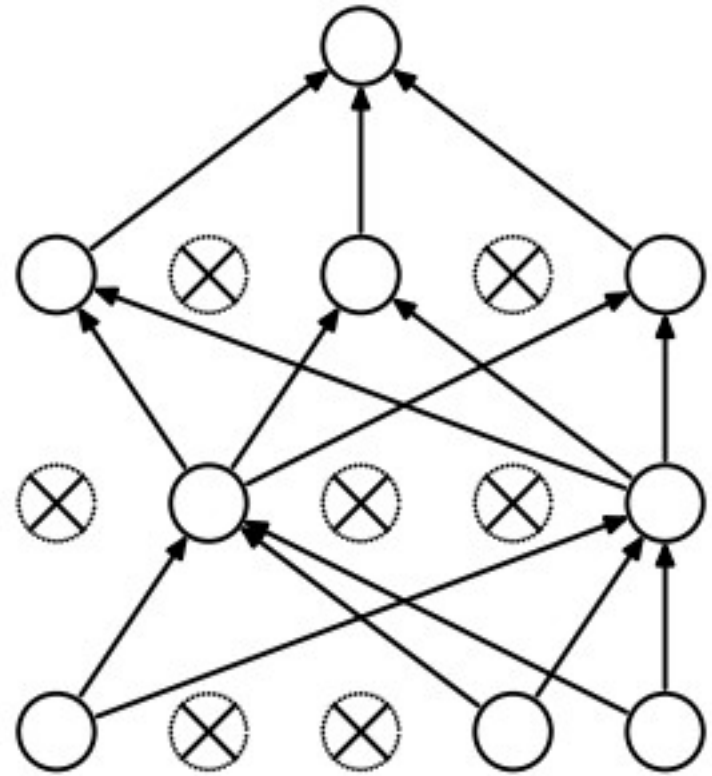


Layer 1


Layer 2

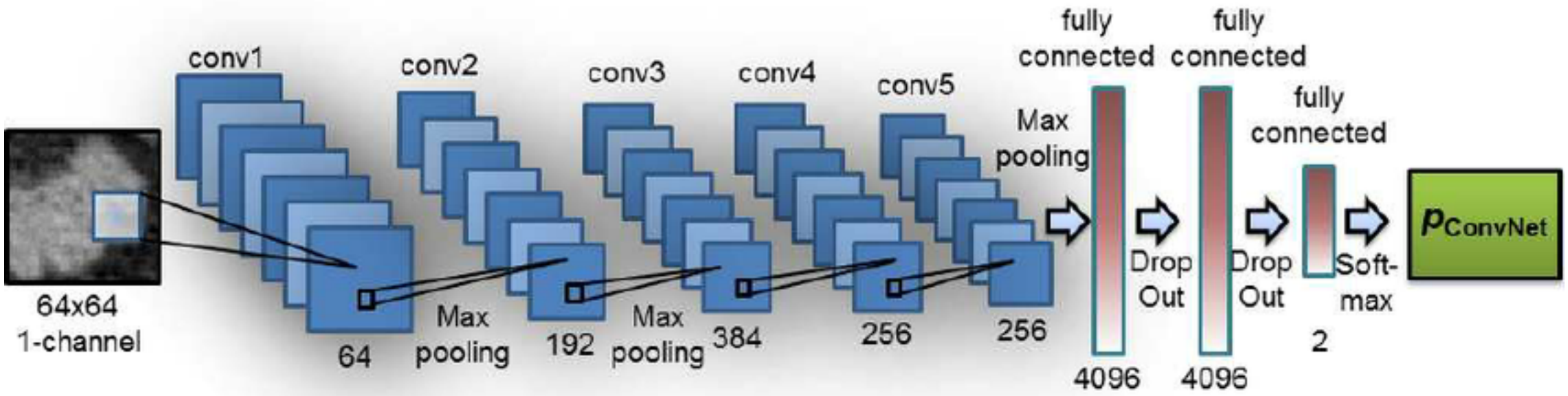


(a) Standard Neural Net

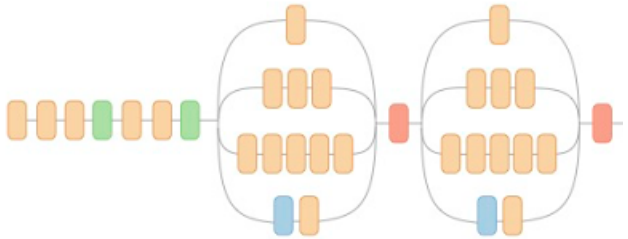


(b) After applying dropout.

Convolutional neural network

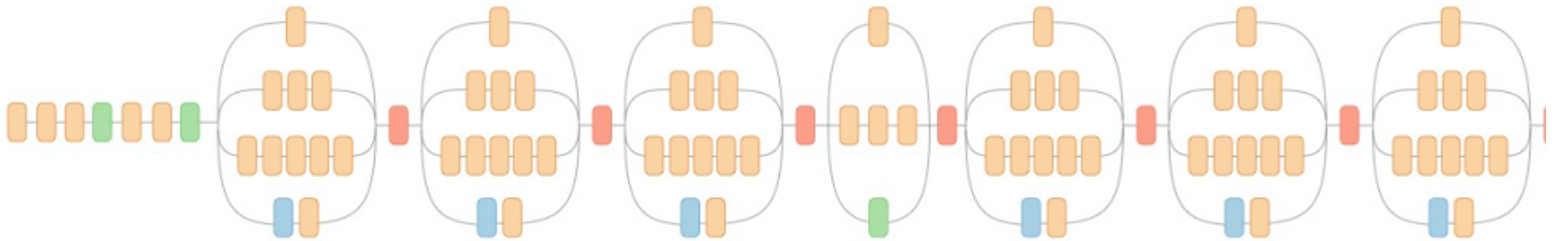


Google Inception Model



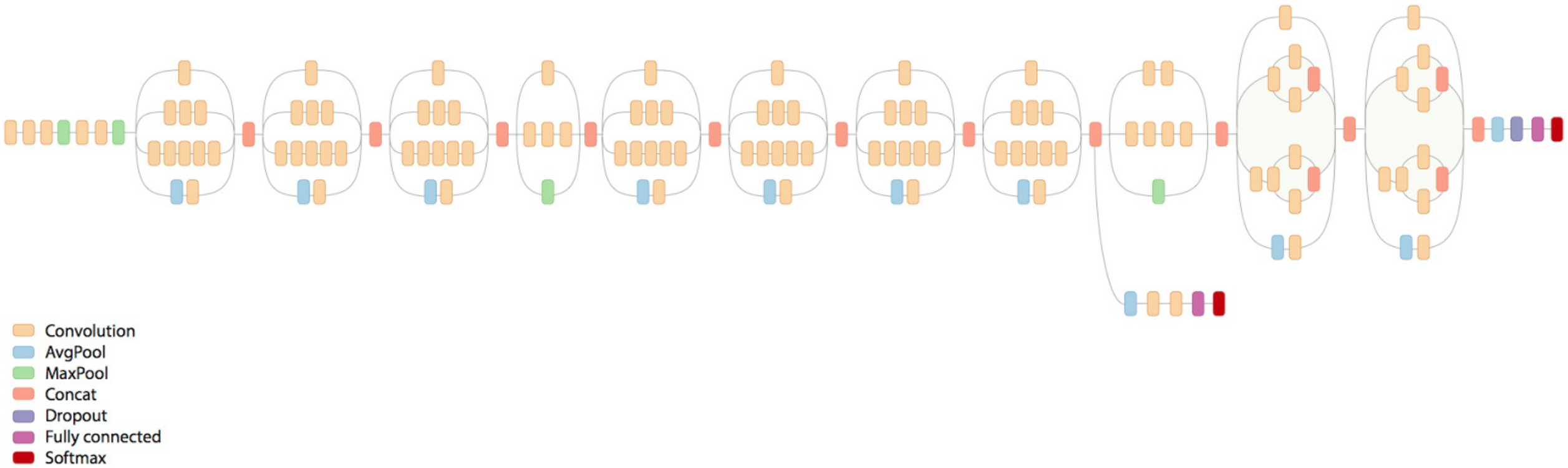
Convolution
AvgPool
MaxPool
Concat
Dropout
Fully connected
Softmax

Google Inception Model



Convolution
AvgPool
MaxPool
Concat
Dropout
Fully connected
Softmax

Google Inception Model



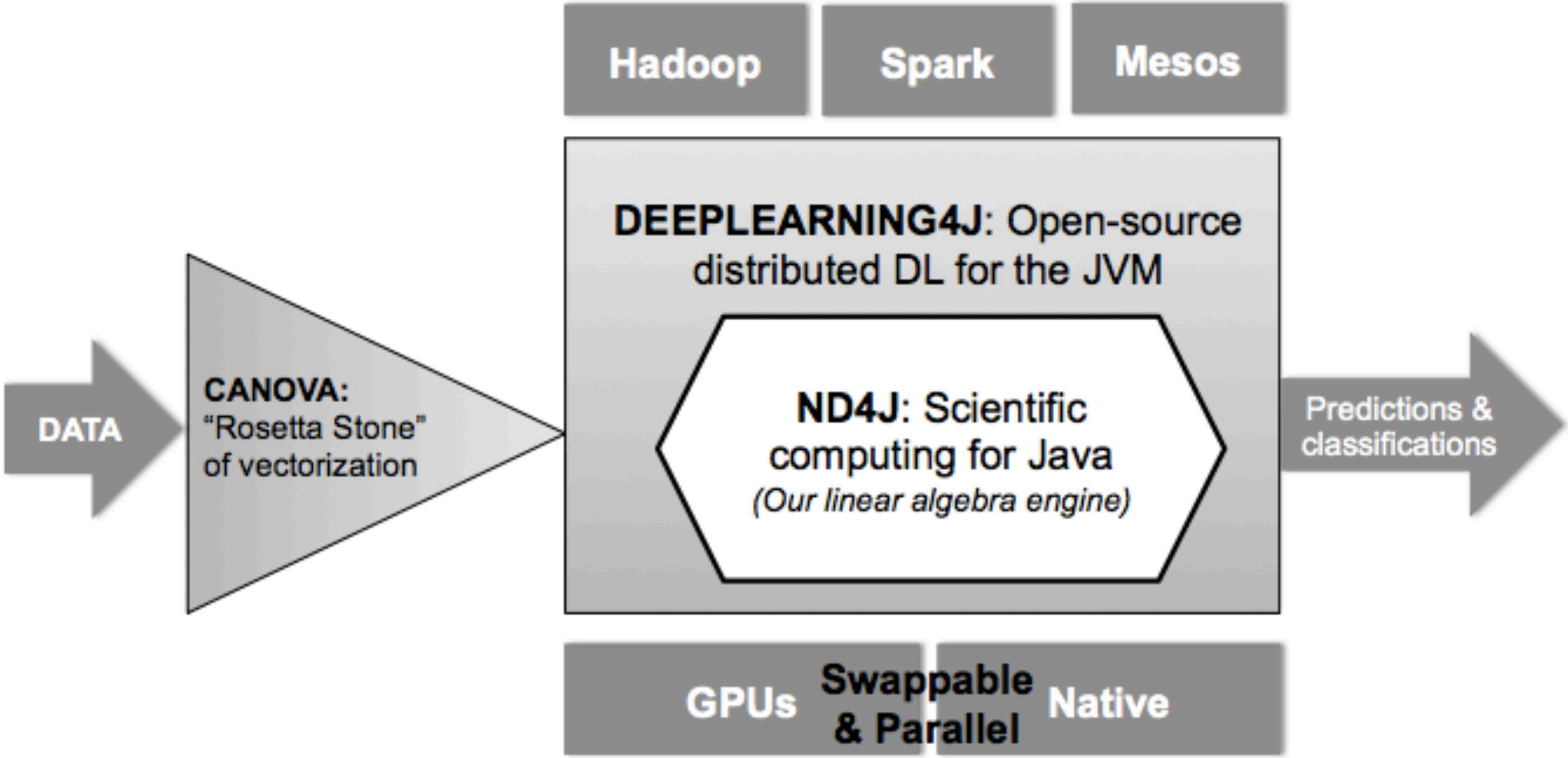
Deep Learning for Java

Open-source, distributed, deep-learning library for the JVM

QUICKSTART

What is Deeplearning4j?

Deeplearning4j is the first commercial-grade, open-source, distributed deep-learning library written for Java and Scala. Integrated with Hadoop and Spark, DL4J is designed to be used in business environments on distributed GPUs and CPUs. Skymind is its commercial support arm.


Hadoop
Spark
Mesos
DEEPLEARNING4J: Open-source distributed DL for the JVM
ND4J: Scientific computing for Java
(Our linear algebra engine)
DATA
CANOVA:
"Rosetta Stone"
of vectorization
Predictions & classifications
GPUs
Swappable & Parallel
Native

Network Configuration

```
MultiLayerConfiguration mlc =
  new NeuralNetConfiguration.Builder()
    .seed(12345)
    .optimizationAlgo(STOCHASTIC_GRADIENT_DESCENT)
    .iterations(1)
    .learningRate(0.006)
    .updater(NESTEROVS)
    .momentum(0.9)
    .regularization(true)
    .l2(1e-4)
    .list()
    …
```

Network Configuration

```
…
.layer(0, new DenseLayer.Builder()
  .nIn(28 * 28) // Number of input datapoints.
.nOut(1000) // Number of output datapoints.
  .activation(Activation.RELU).weightInit(XAVIER)
   .build())
.layer(1, new OutputLayer.Builder(NEGATIVELOGLIKELIHOOD)
   .nIn(1000).nOut(10)
   .activation(SOFTMAX).weightInit(XAVIER).build())
.pretrain(false)
.backprop(true)
.build();
```

Model Initialization

```
MultiLayerNetwork mlpNet =
   new MultiLayerNetwork(conf);

mlpNet.init();
```

Training the model

```
DataSetIterator dataSetIterator = ...

for(int i=0; i < numEpochs; i++) {
    model.fit(dataSetIterator);
}
```

Evaluation

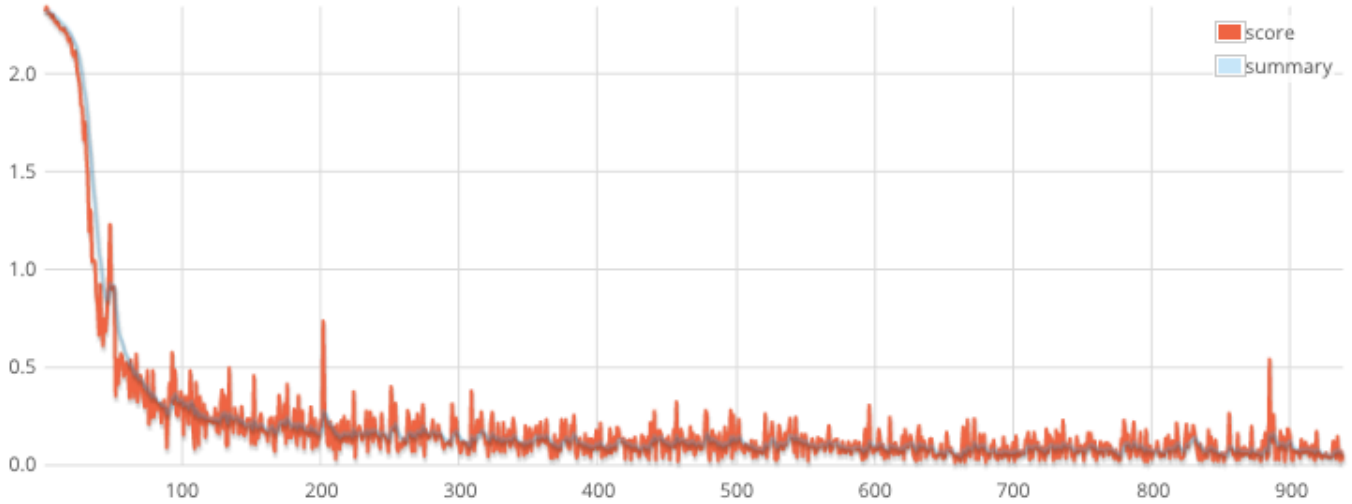
```
evaluator = new Evaluation(outputNum);
while(testSetIterator.hasNext()){
  DataSet next = dataSetIterator.next();
  INDArray guesses =
    model.output(next.getFeatureMatrix(),false);
  INDArray realOutcomes = next.getLabels();
  evalaluator.eval(, output);
}

log.info(eval.stats());
```

DL4J Training UI

- Overview
- Model
- System
- Language

Model Score vs. Iteration

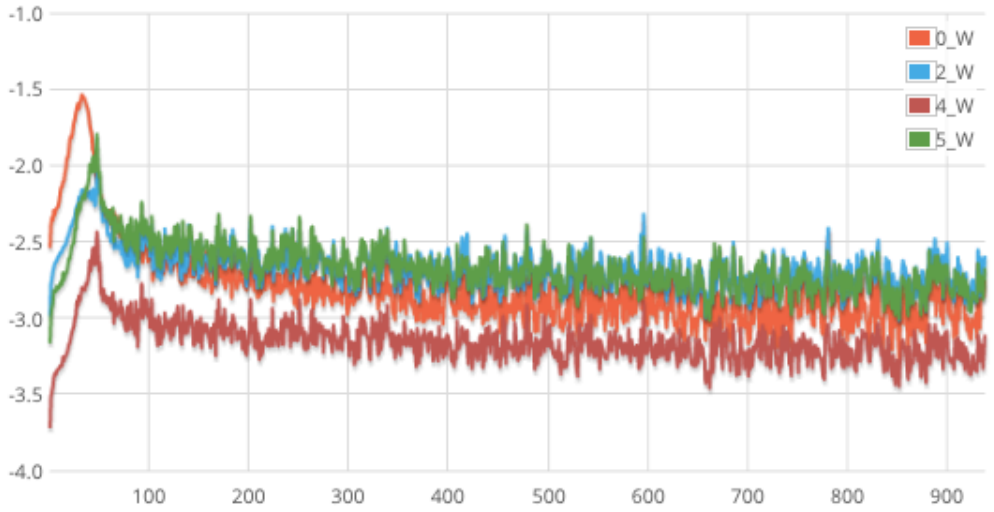


Score : 2.55863, **Iteration :** 537

Model and Training Information

Model Type	MultiLayerNetwork
Layers	6
Total Parameters	431080
Start Time	
Total Runtime	
Last Update	2017-02-06 11:34:56
Total Parameter Updates	938
Updates/sec	2.86
Examples/sec	91.43

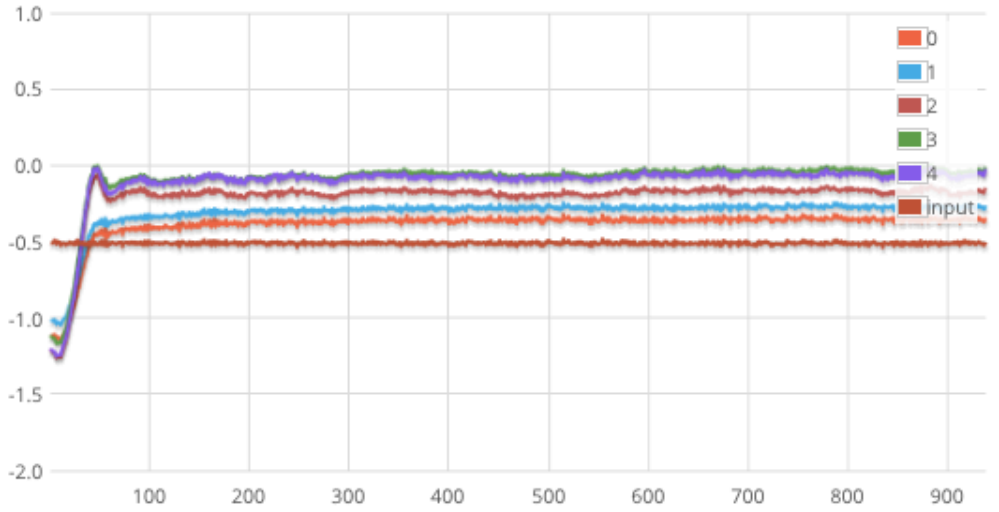
Update:Parameter Ratios (Mean Magnitudes): $\log_{10}$



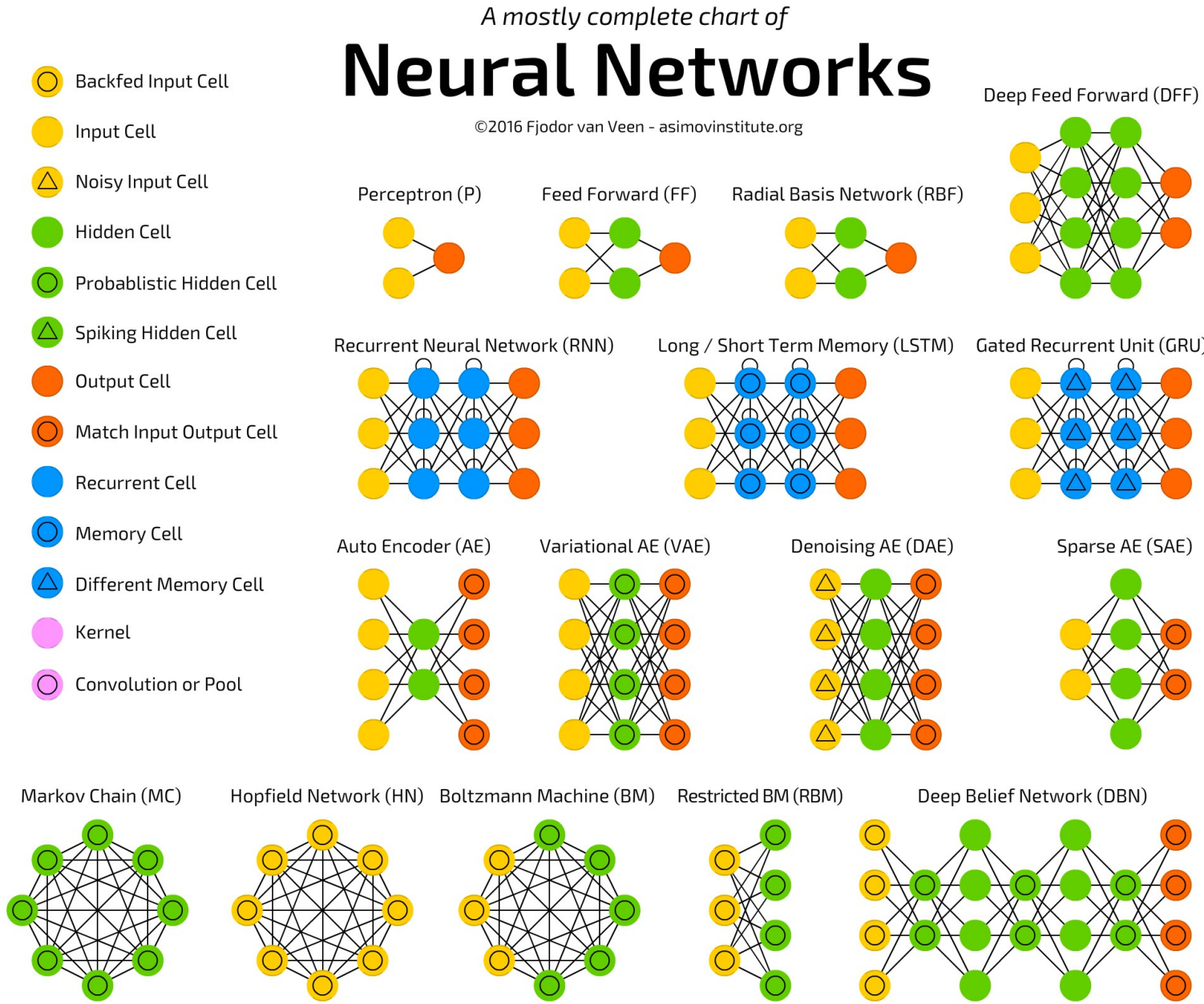
Ratio : 0.89543, **$\log_{10}$ Ratio :** -0.04797 , **Iteration:** 401

Standard Deviations: $\log_{10}$

Updates | Gradients | Activations



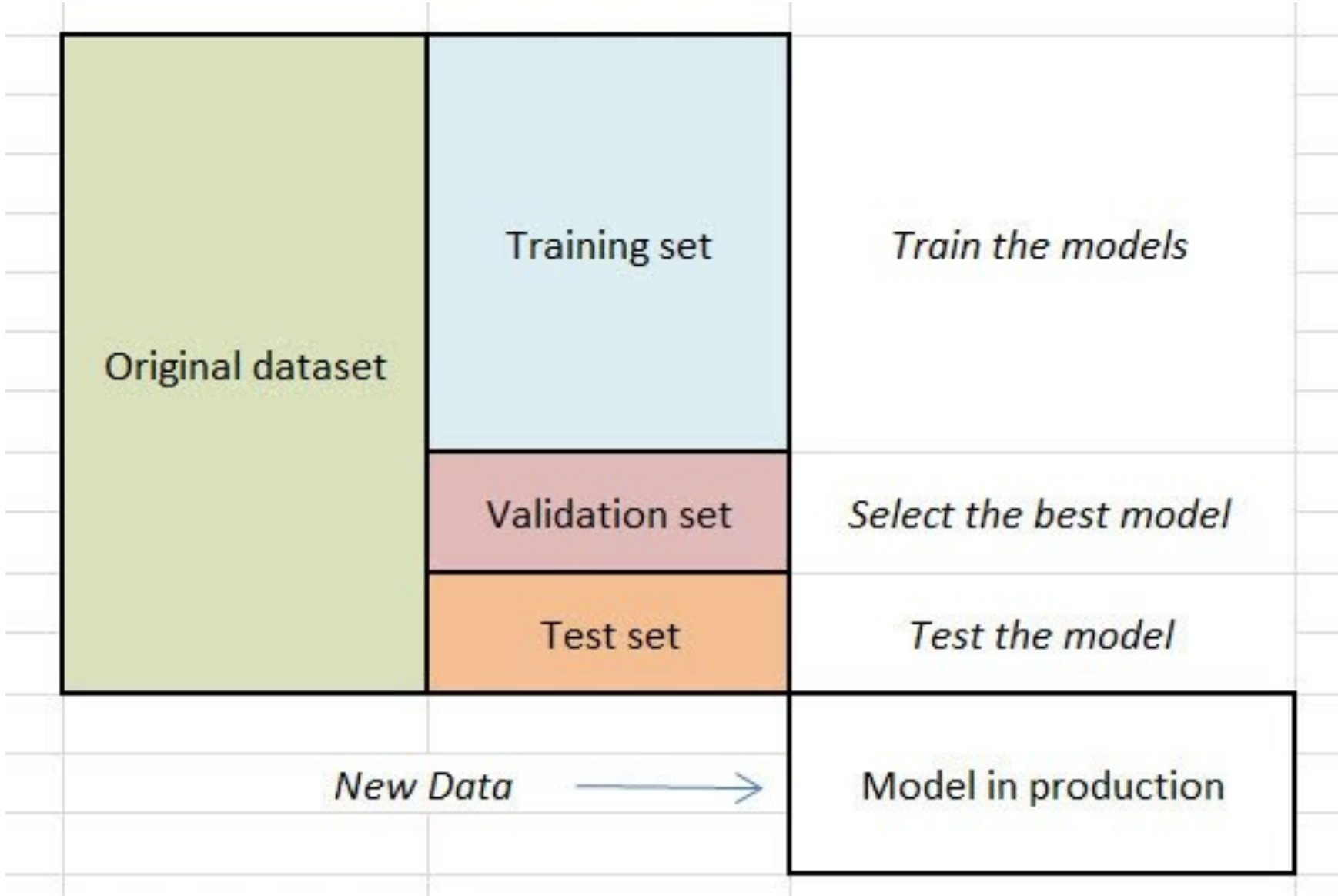
St. Dev : 0, **$\log_{10}$ St. Dev :** 0 , **Iteration:** 0



"A Neural Network Zoo," Fjdor Van Neen http://www.asimovinstitute.org/neural-network-zoo/

Data Science/Engineering

- Data selection
- Data processing
  - Formatting & Cleaning
  - Sampling
- Data transformation
  - Feature scaling & Normalization
  - Decomposition & Aggregation
  - Dimensionality reduction


Original dataset
Training set
Validation set
Test set
Train the models
Select the best model
Test the model
New Data
Model in production

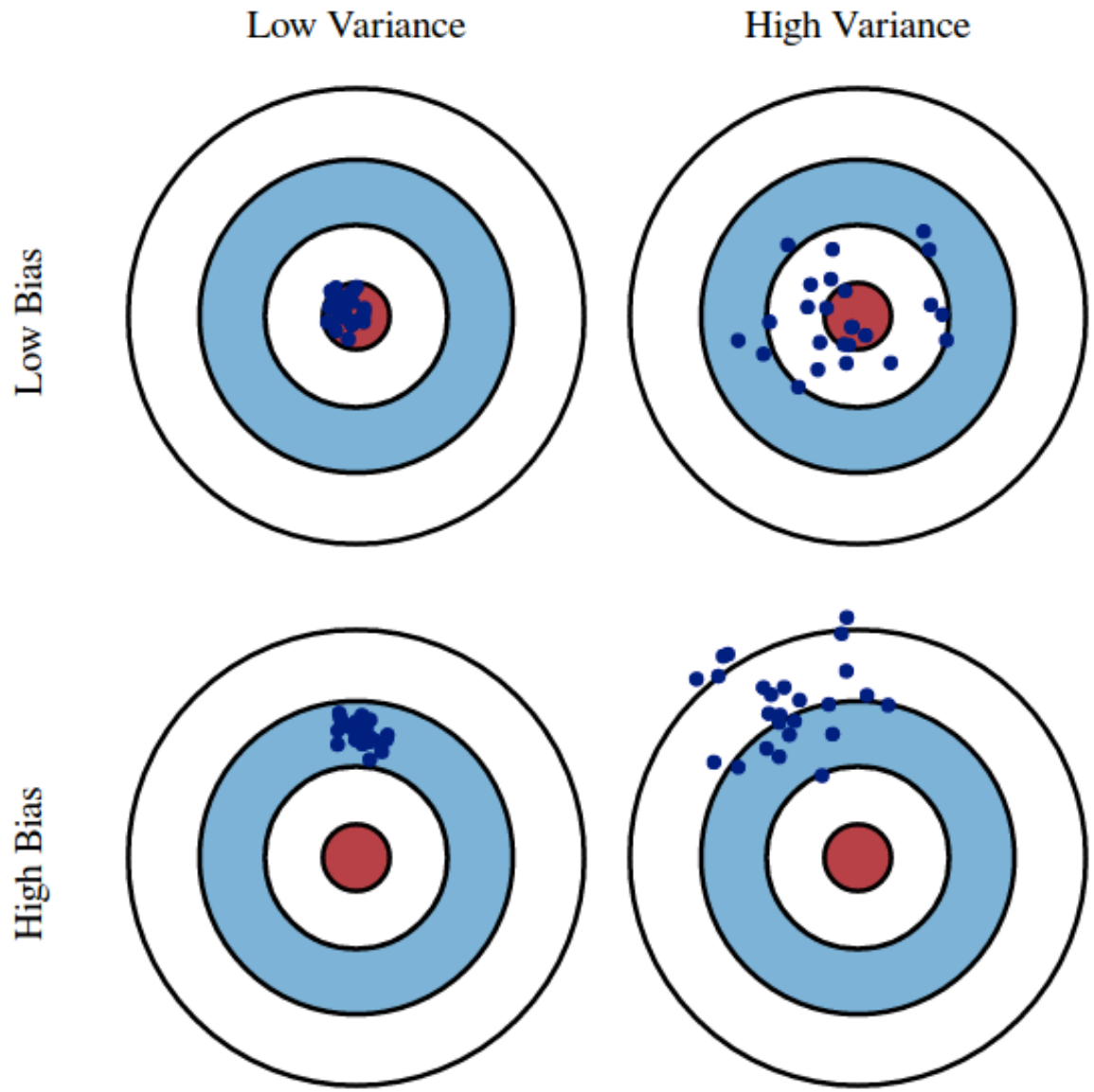
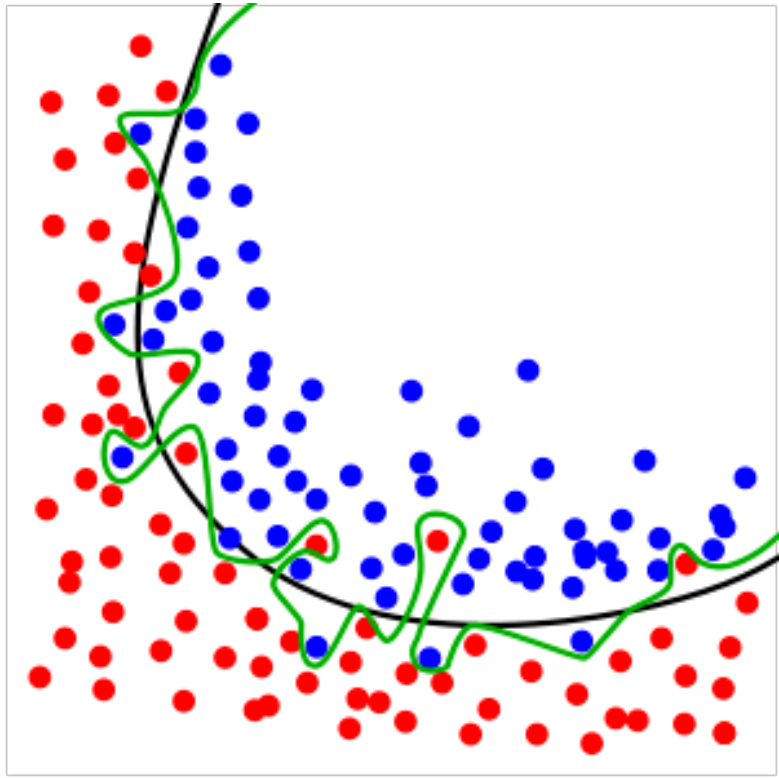


Fig. 1 Graphical illustration of bias and variance.

Common Mistakes

- Training set – 70%/30% split
- Test set – Do not show this to your model!
- Sensitivity vs. specificity
- Overfitting



Training your own model

- Requirements
  - Clean, labeled data set
  - Clear decision problem
  - Patience and/or GPUs
- Before you start

Preparing data for ML

- Generating Labels
- Dimensionality reduction
- Determining salient features
- Visualizing the shape of your data
- Correcting statistical bias
- Getting data in the right format

Further resources

- CS231 Course Notes
- Deeplearning4j Examples
- Visualizing MNIST
- Neural Networks and Deep Learning
- Andrew Ng’s Machine Learning class
- Awesome Public Datasets

Thank You!

SKYMIND

coursera

Mary, Mark, Margaret, Hanneli