Adobe Workshop on Machine Learning

How many of you

- Are you eager to work in a real project with Machine Learning (ML)?
- Have you used machine learning on your hobby projects?
- Are involved in projects that use ML technology implementation or related activities?
- Feel you understand the underlying ML paradigm?

About me

- A Machine Learning enthusiast and expert;
- 10 years of experience in software development;
- Associate Professor at UPB, ETTI;
- 8 year of experience in research:
  - PhD and postdoc research in the field of computer vision and multimedia retrieval;
  - Over 40 publications in international journals, conferences and workshops;
  - Participation at various international competitions;
  - Part of the organizing team for various conferences and competitions;
  - Technical reviewer for various journals, conferences and workshops.

More info on http://ionut.mironica.ro

PART 1

Agenda

- What is and what types of ML exist?
- How does is work?
- How can be used?
- How we can combine different ML algorithms?
- Ethics in ML ?
- Applications

Agenda

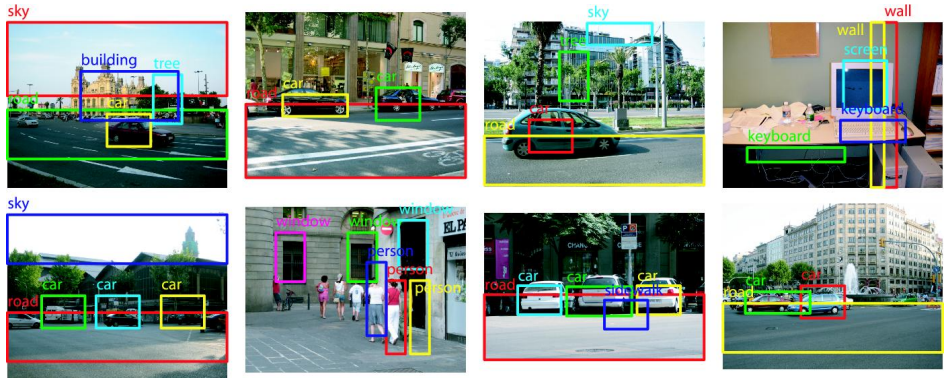
- Types of Machine Learning
- Feature extraction
- Classical Machine Learning algorithms
- Deep learning algorithms
- Tools for Machine Learning

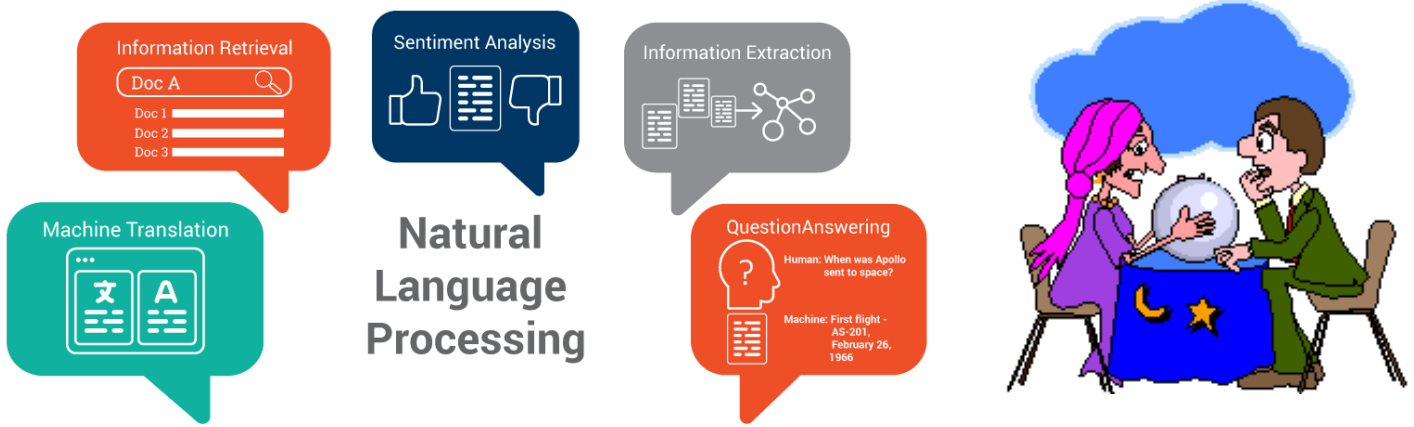
Artificial Intelligence

Artificial Intelligence (AI) is the science of making the things smart and it can be defined as:

"Human Intelligence exhibited by machines"

A broad term of getting computers to perform human tasks. The scope of AI is disputed and constantly changing over the time.





[Images from: http://bryanrussell.org and http://m.el-dosuky.com]

Machine Learning

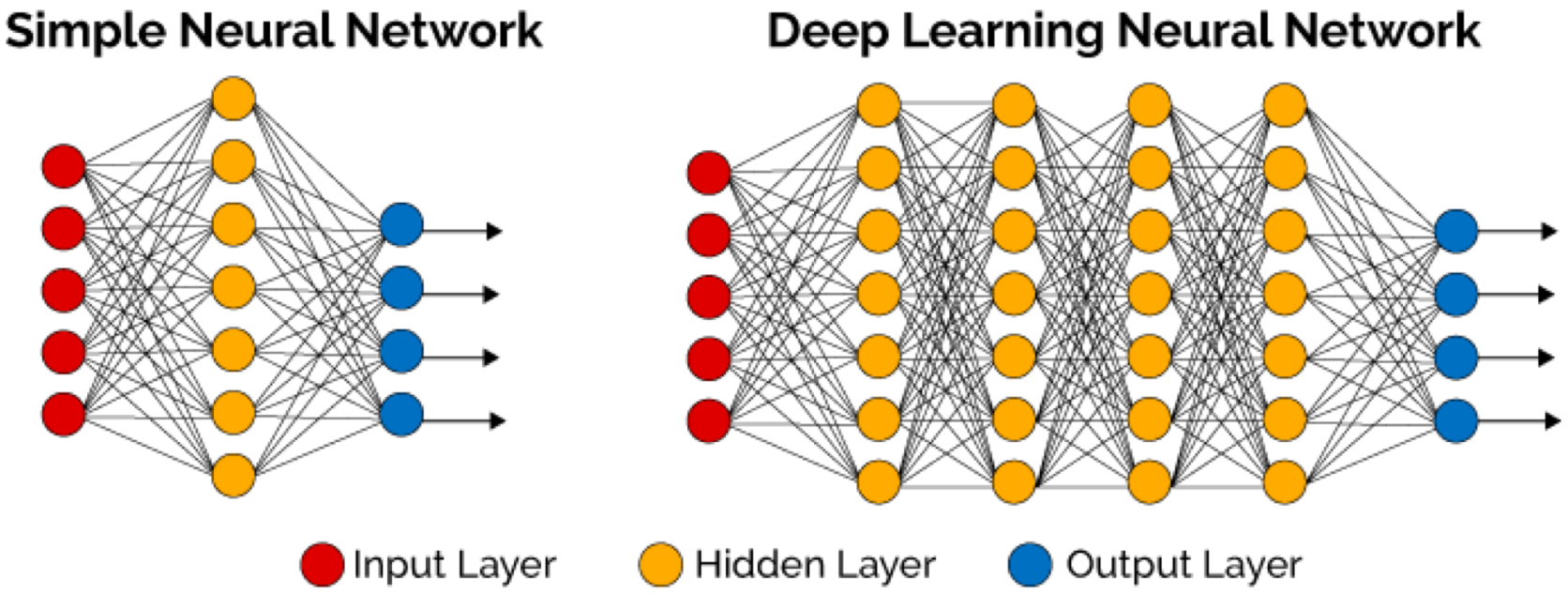
Machine Learning (ML) can be defined as:

"An approach to achieve artificial intelligence through systems that can learn from experience to find patterns in that data"

ML involves teaching computer by examples and learning patterns and not programming specific rules. And these patterns can be found in **data**.

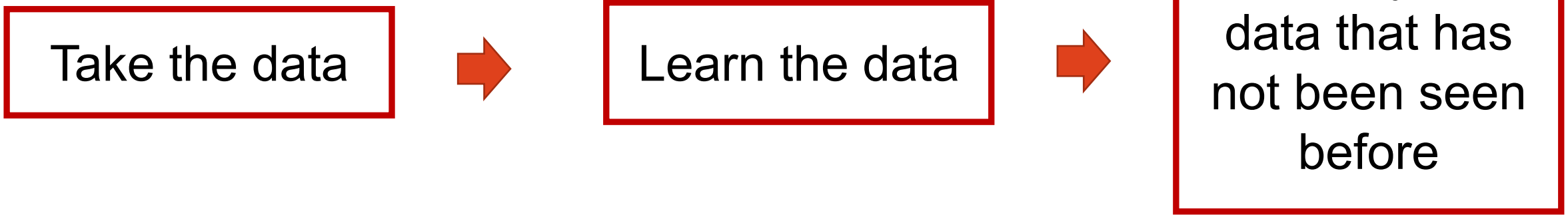
Deep Learning

The subset of machine learning composed of algorithms that permit software to train itself to perform tasks, like speech and image recognition, by exposing multilayered neural networks to vast amounts of data.



[Image from: http://www.global-engage.com/life-science/deep-learning-in-digital-pathology]

Machine Learning



Classical programming vs Machine learning

USE CASE: SPAM FILTER

If email contains "Viagra" → Spam

If email contains "Weight loss" → Spam

If email contains "…" → Spam

If email contains "…" → Spam

Take the spam / non spam examples

↓

Learn the data / generate model

↓

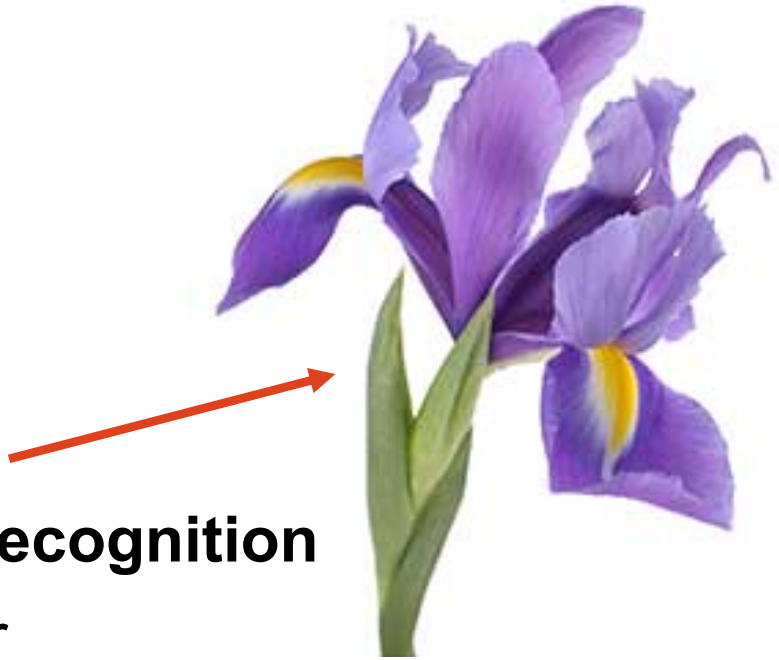
Classify the emails

Feature extraction

- Features are used to train a ML system.
- Represents properties of the things that you are trying to learn about

Weather forecasting

- humidity
- pressure
- temperature



Plant recognition

- color
- sepal / petal width / height

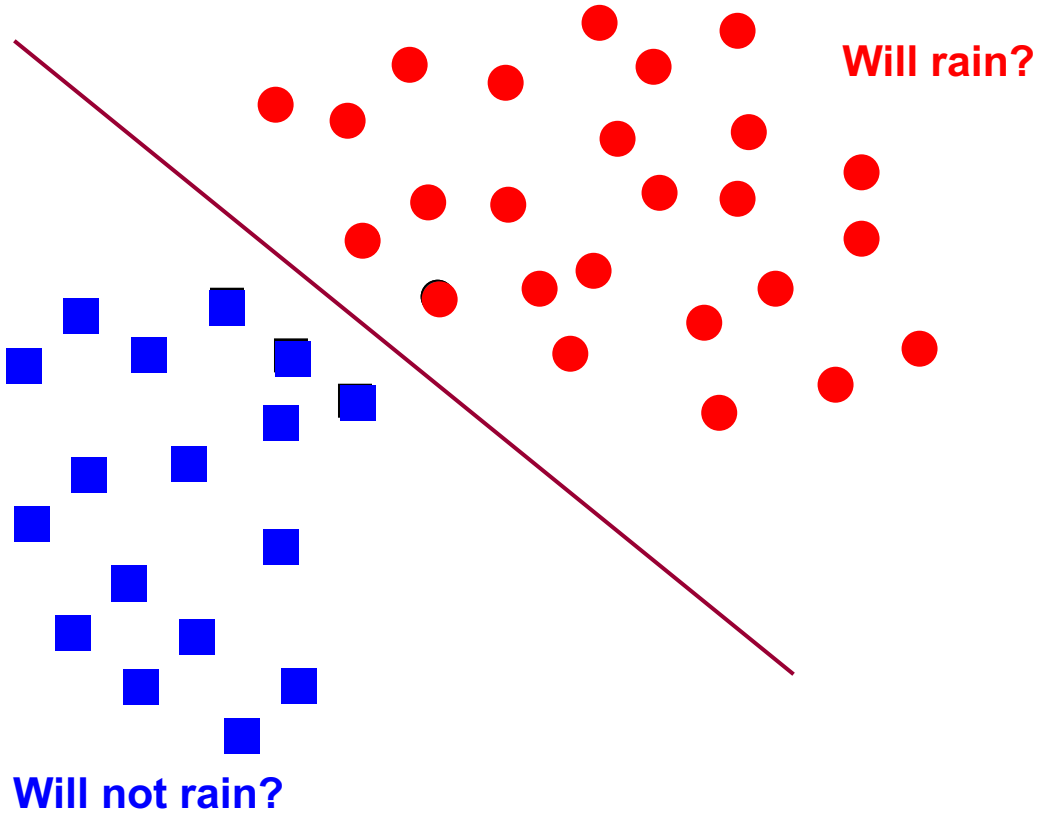
[Image from: www.collinsdictionary.com]

Feature extraction

WEATHER FORECASTING

- humidity
- pressure

Choosing the useful features can have a high impact on the system's classification accuracy.



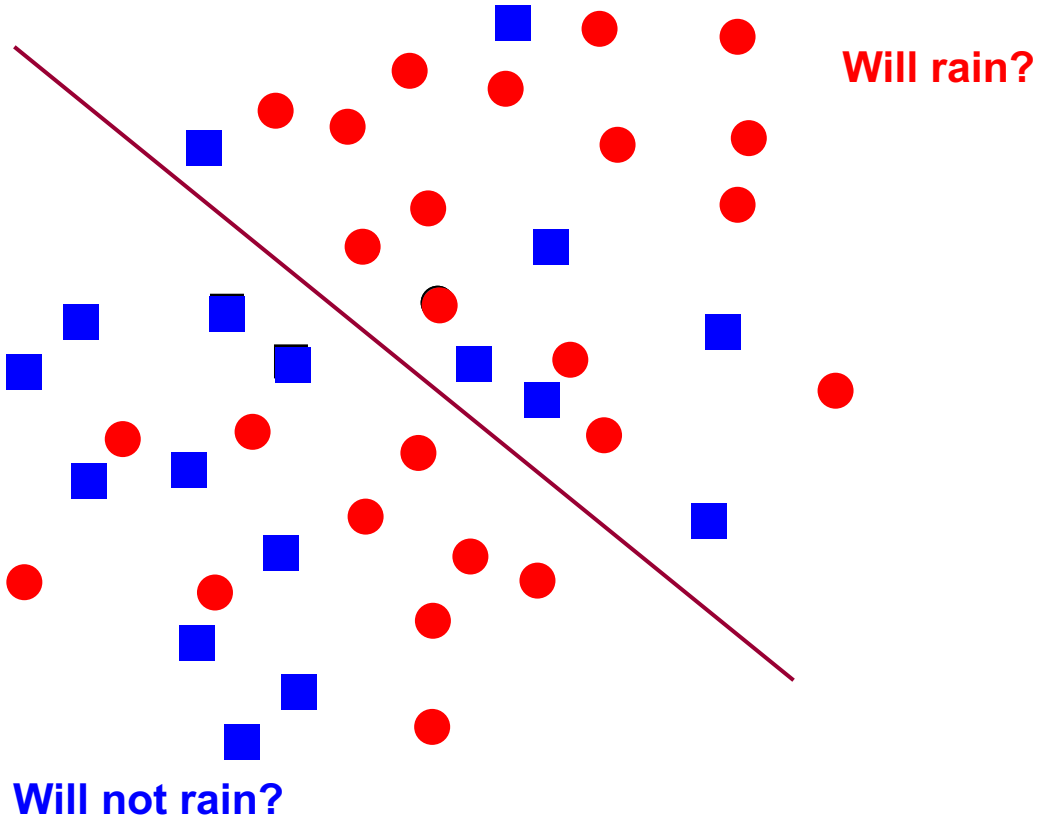
Feature extraction

WEATHER FORECASTING

- humidity
- pressure

Choosing the useful features is not a trivial task

There are systems that may have millions of features, so the visualization of the feature space is almost impossible



Feature extraction

Adding more dimensions may help on improving the performance, allowing the algorithm to better separate the classes

More dimensions can generate

"CURSE OF DIMENSIONALITY":

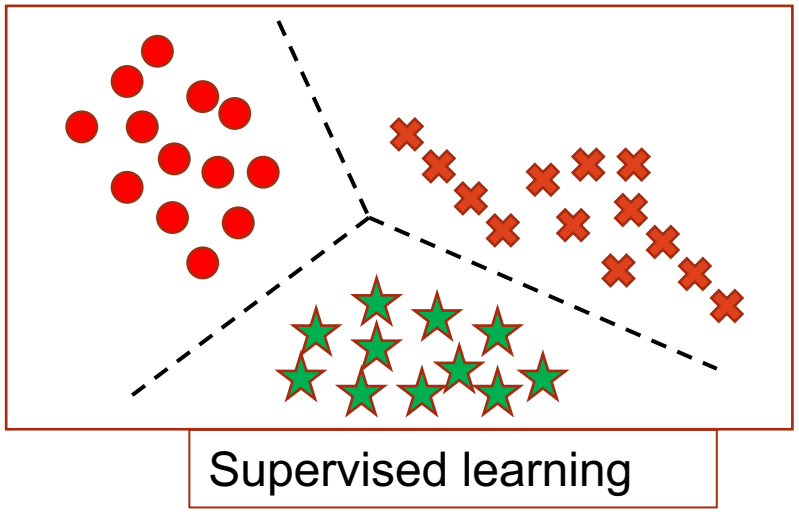
A phenomena that occurs when the dimensionality of the data increases, the sparsity of the data increases.

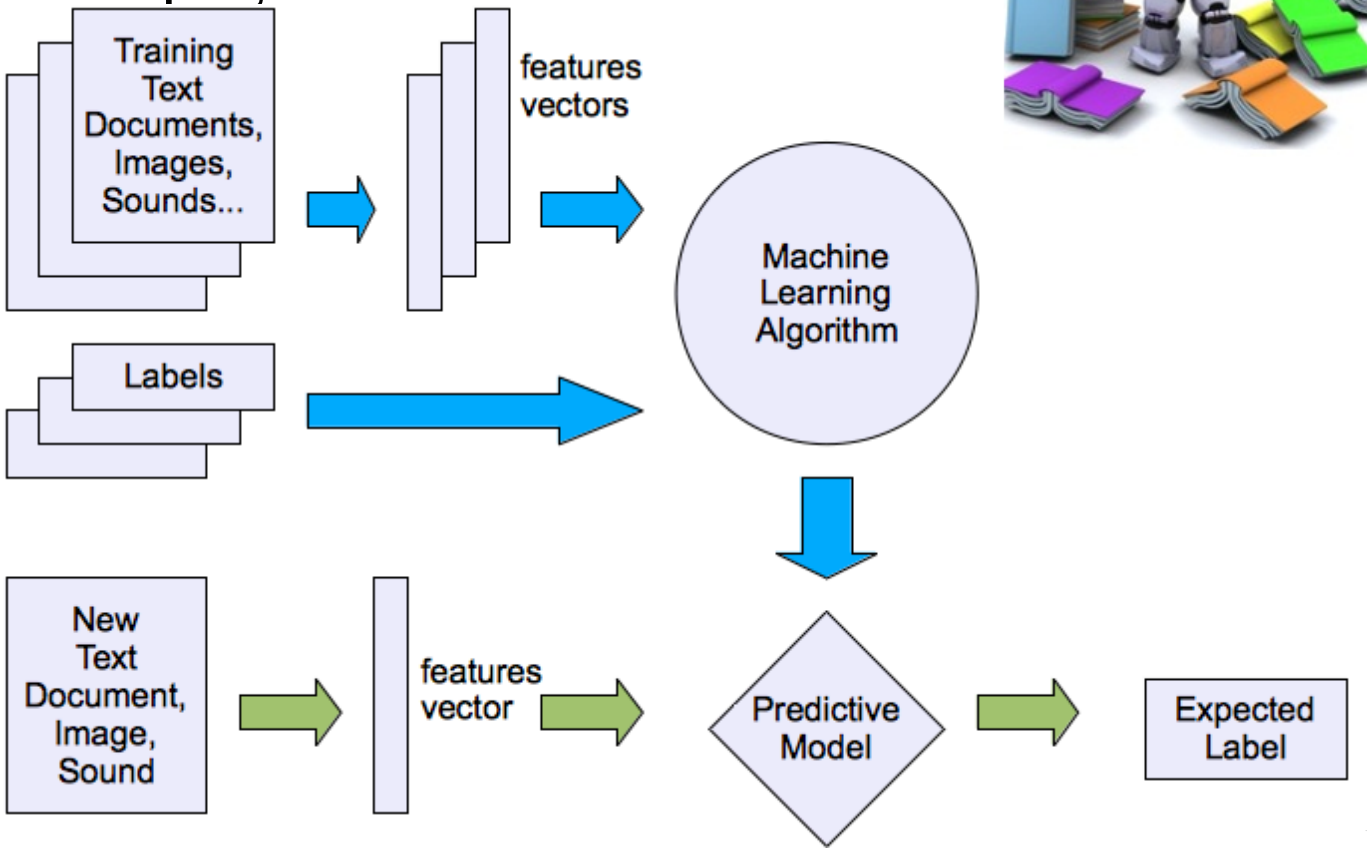
[Image from: https://www.futuristspeaker.com]

Machine learning classification

SUPERVISED LEARNING

- Training data contains the desired output;

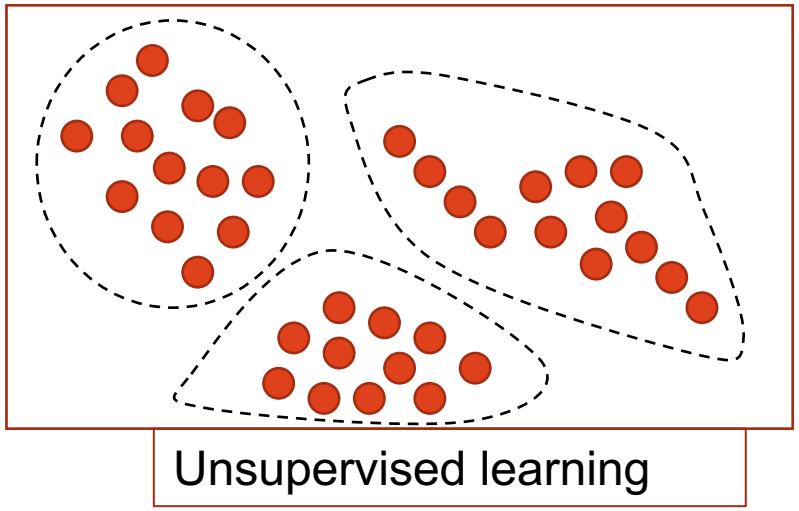


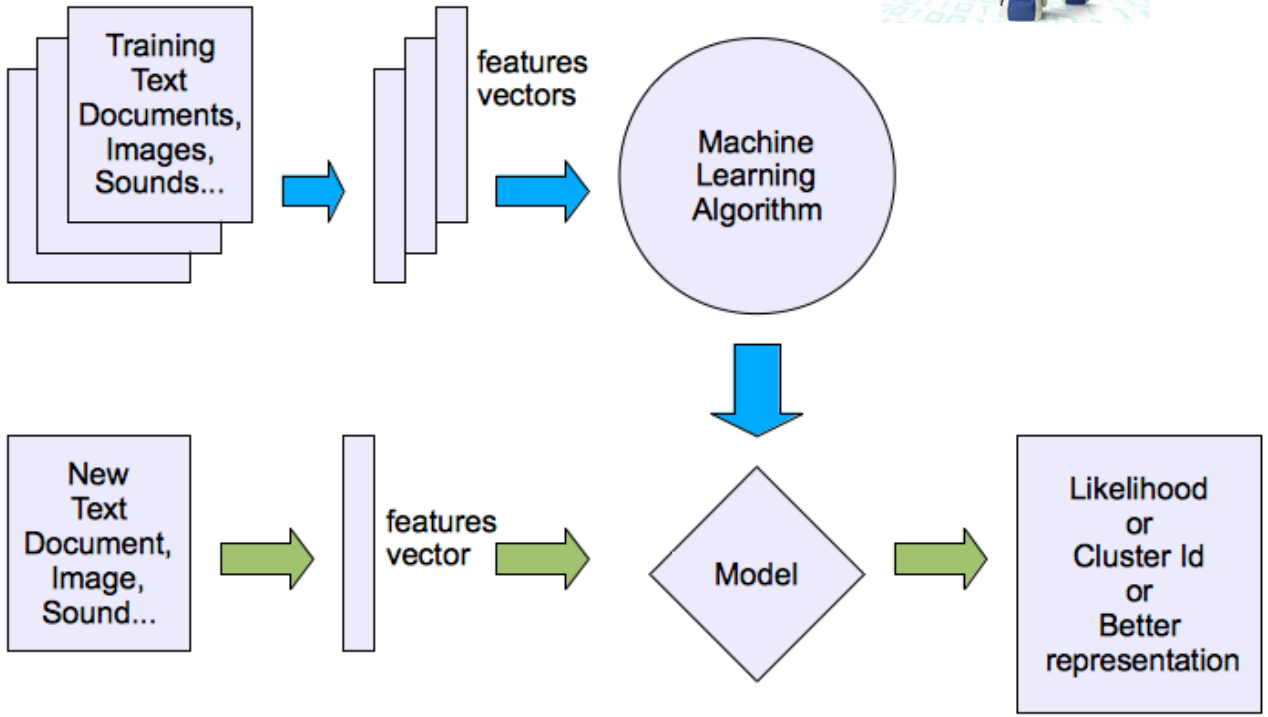


Machine learning classification

UNSUPERVISED LEARNING

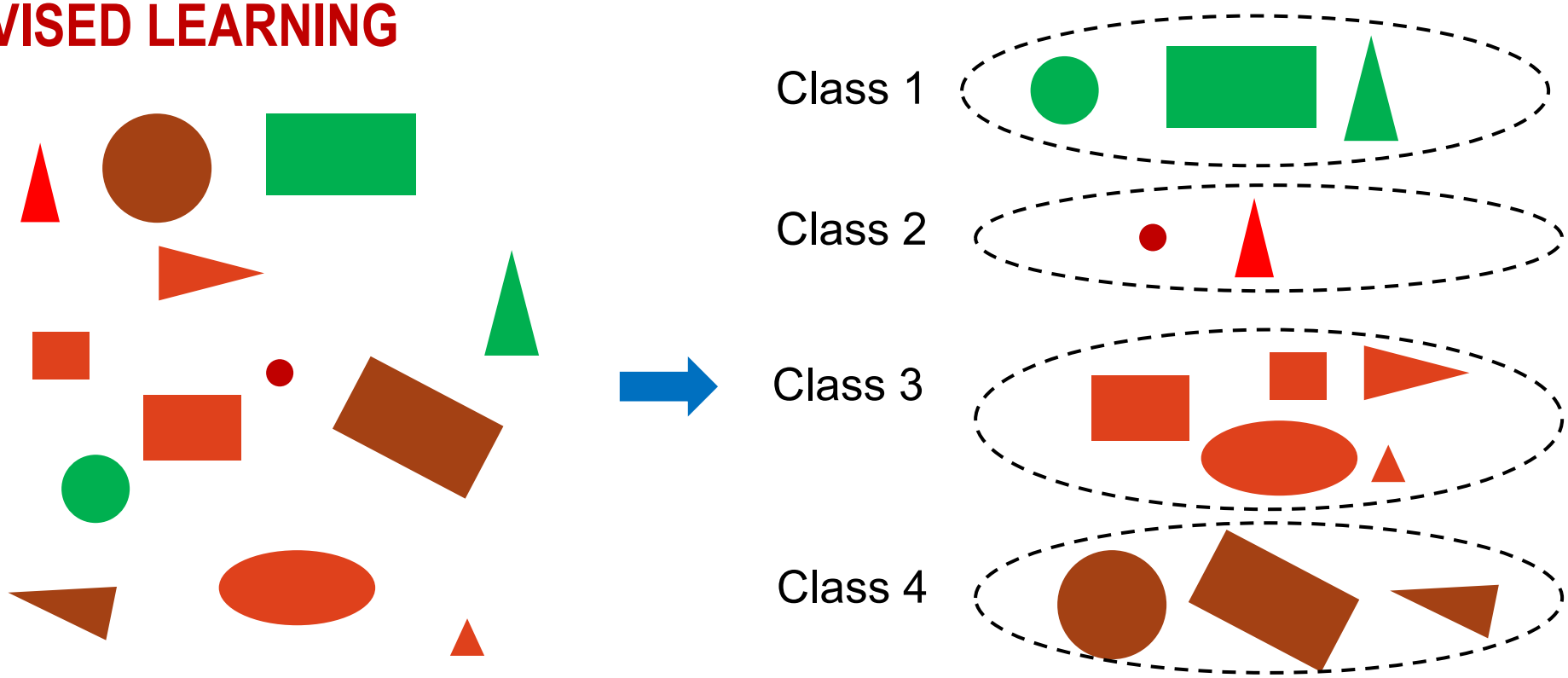
- Training data doesn't contain output;
- The basic idea is to find templates and patterns in data to be automatically highlighted.





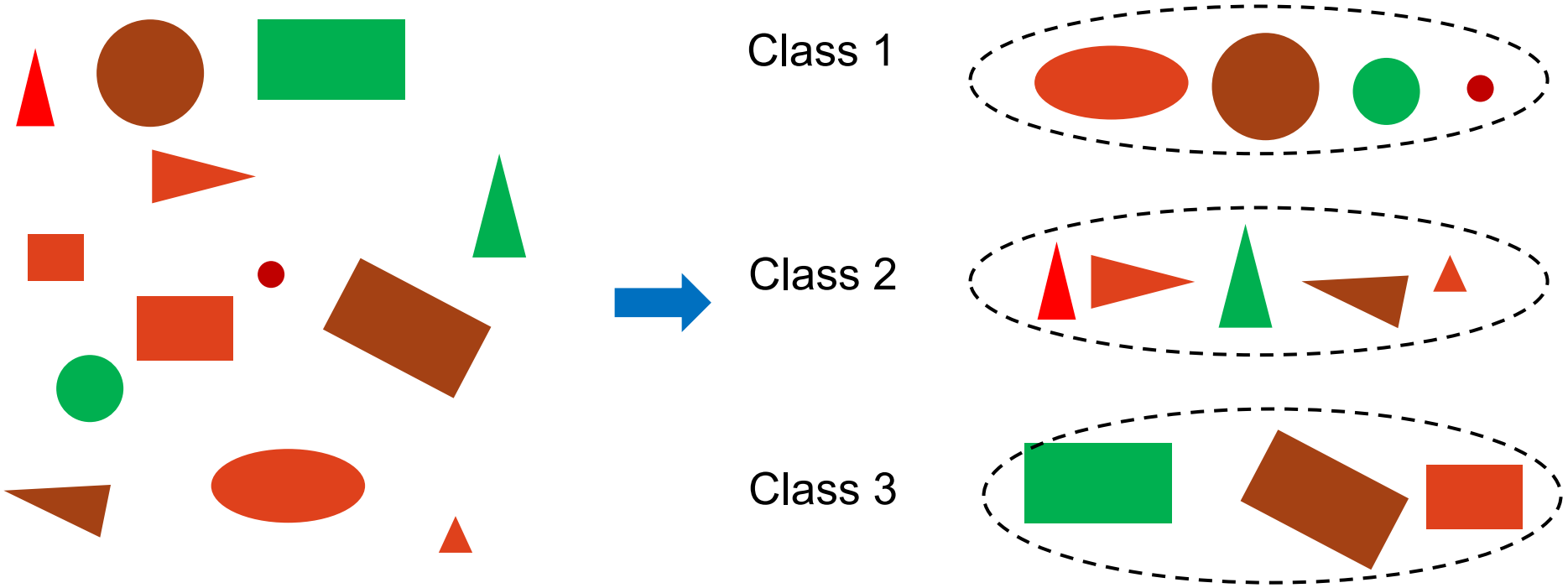
Machine learning classification

UNSUPERVISED LEARNING



Machine learning classification

UNSUPERVISED LEARNING



Machine learning classification

UNSUPERVISED LEARNING

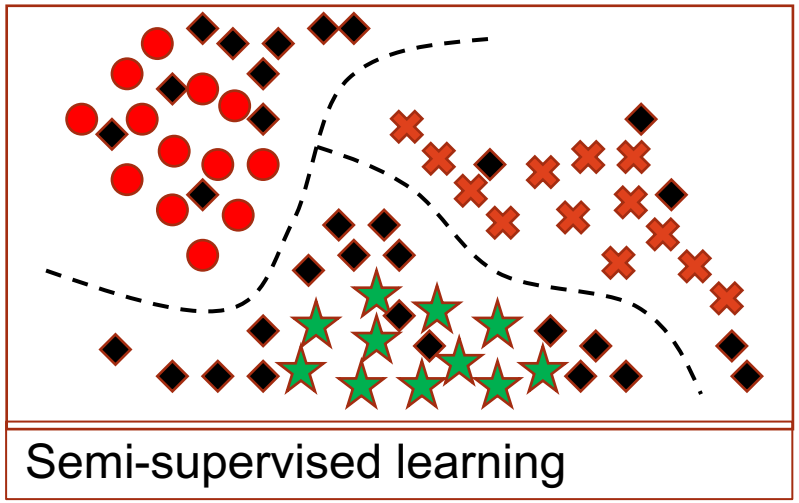
- Similarity – a subjective concept;
- Hard to define at human level.

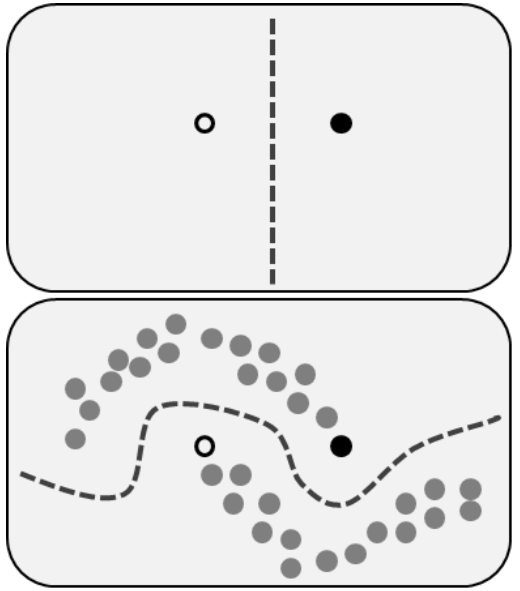
[Images from: www.cesar.com and www.purina.com]

Machine learning classification

SEMI-SUPERVISED LEARNING

- Semi-supervised learning is a class of supervised learning tasks and techniques that also make use of unlabeled data for training – typically a small amount of labeled data with a large amount of unlabeled data.



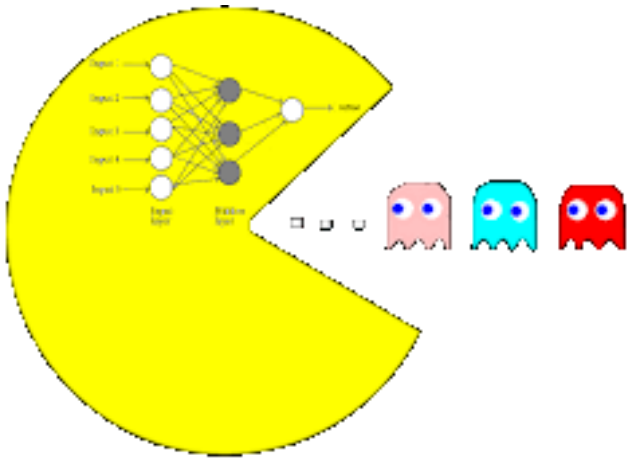


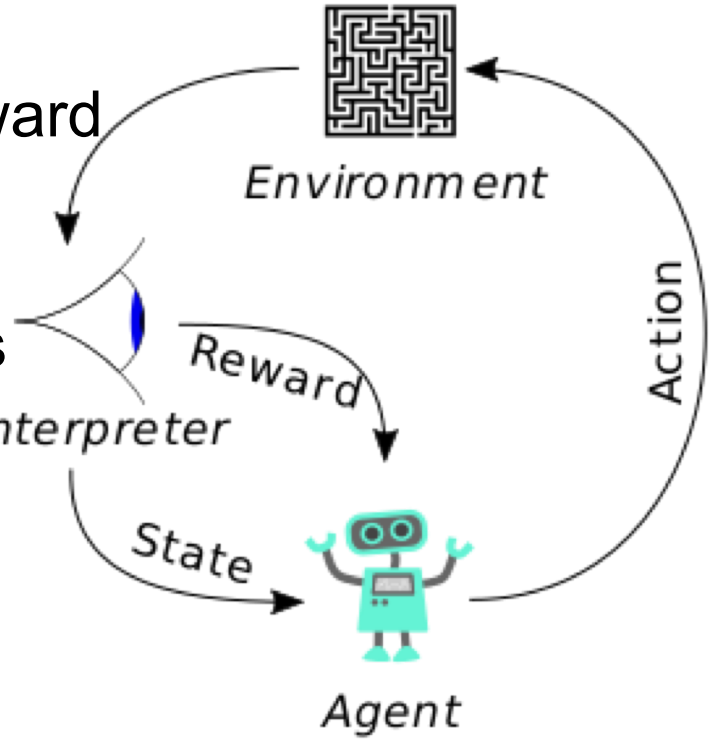
[Image from: http://wiki.com]

Machine learning classification

REINFORCEMENT LEARNING

- Instead of optimizing a cost function is maximizing a reward function
- Learning by trial and error throw reward and punishment;
- The program learns by playing the game millions on times

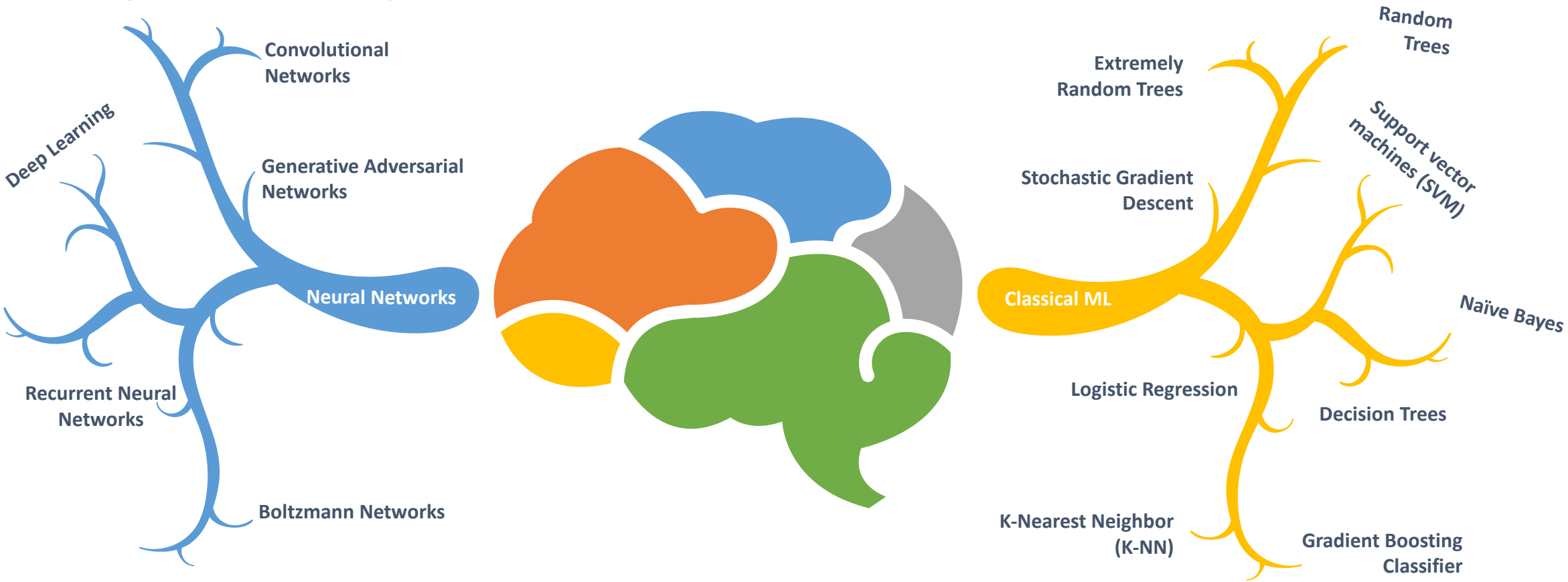




[Image from: http://wiki.com]

Machine learning algorithms

SUPERVISED LEARNING

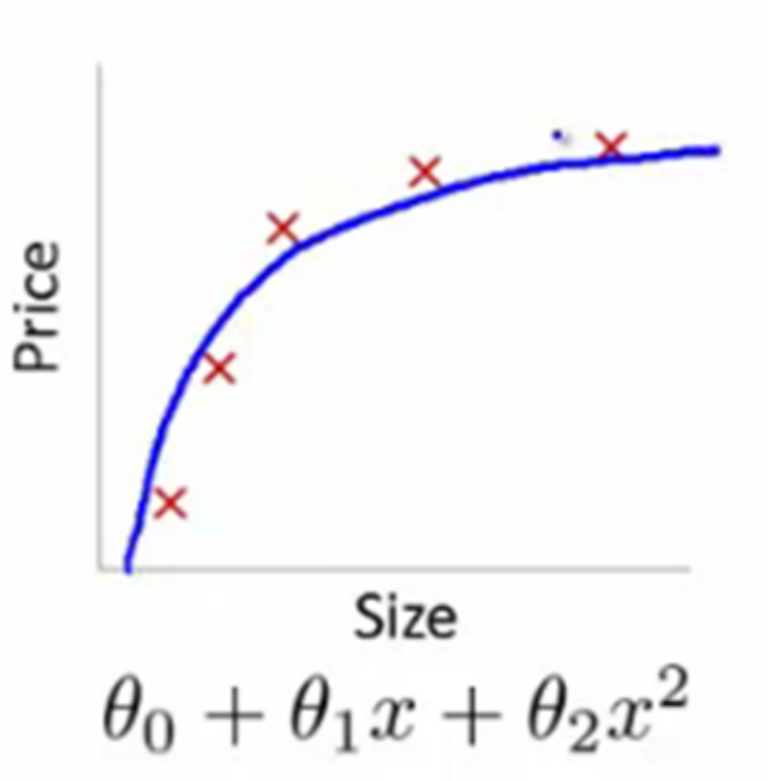


What is the best algorithm?

No free lunch theorem

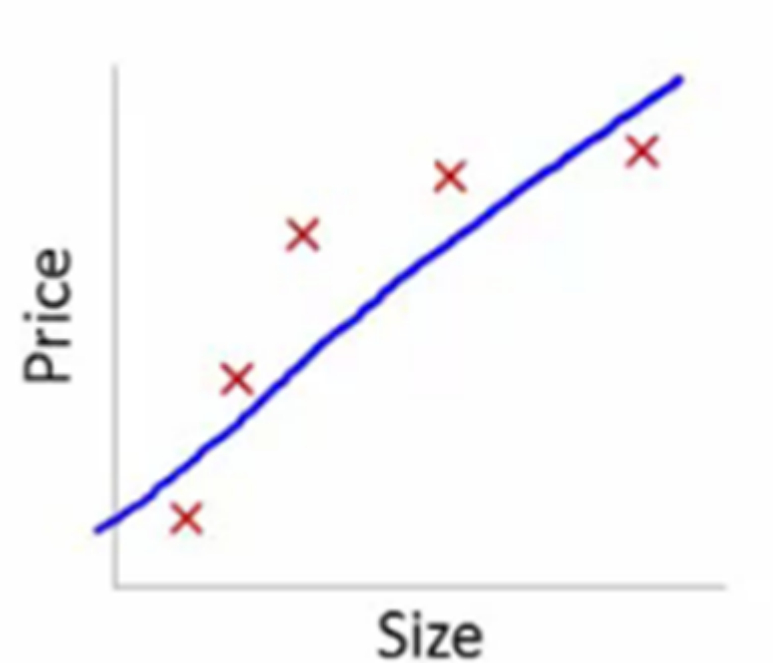
[Image from: https://cynicalbabblings.wordpress.com]

Generalization



Generalization

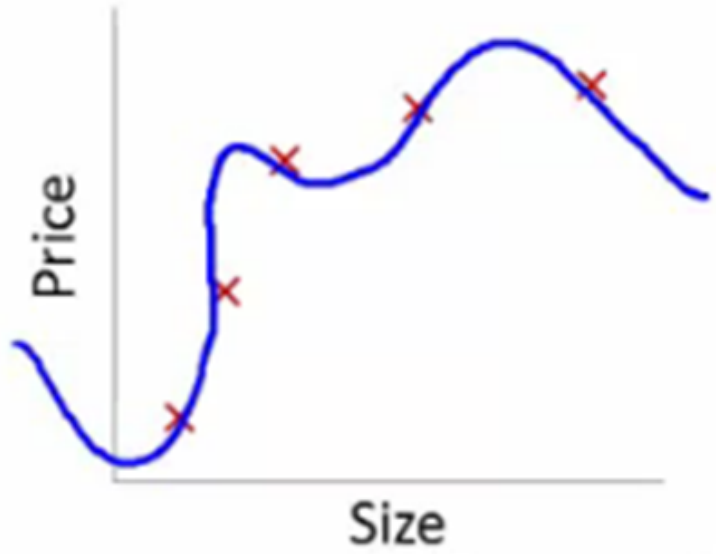
UNDERFITTING A model is underfitted when it is too simple with regards to the data that is trying to model.



This is because the model is unable to capture the relationship between the input examples (called X) and the target values (called Y).

Generalization

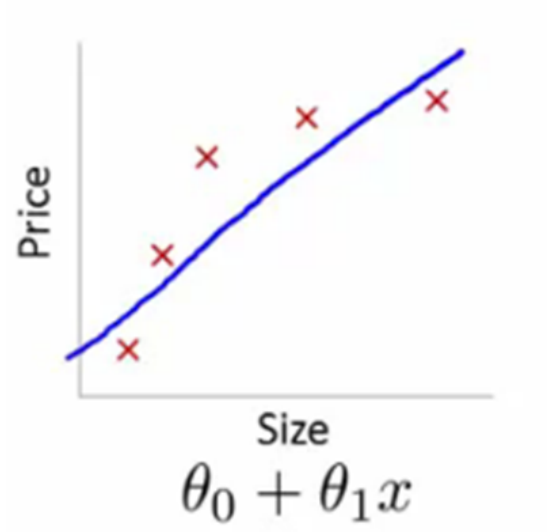
OVERFITTING The model is overfitted when on the training data the model performs well but it does not have reasonable results on the evaluation data.



This is because the model is memorizing the data used during the training sample and it is unable to generalize to unseen examples.

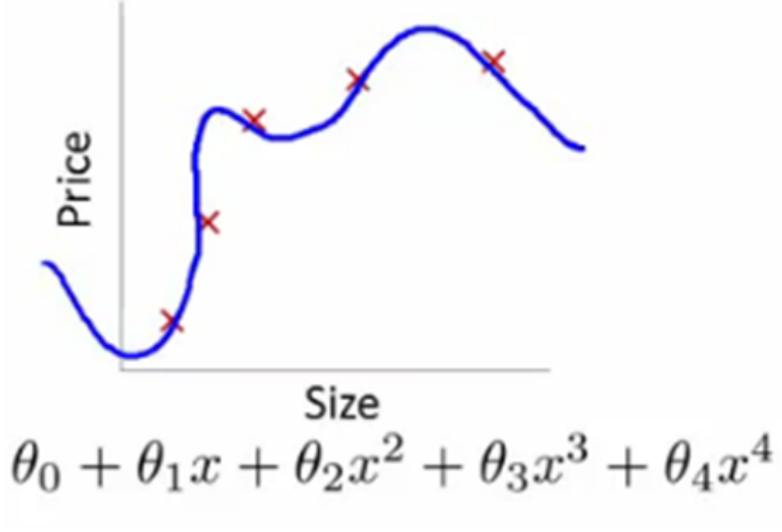
Generalization

UNDERFITTING



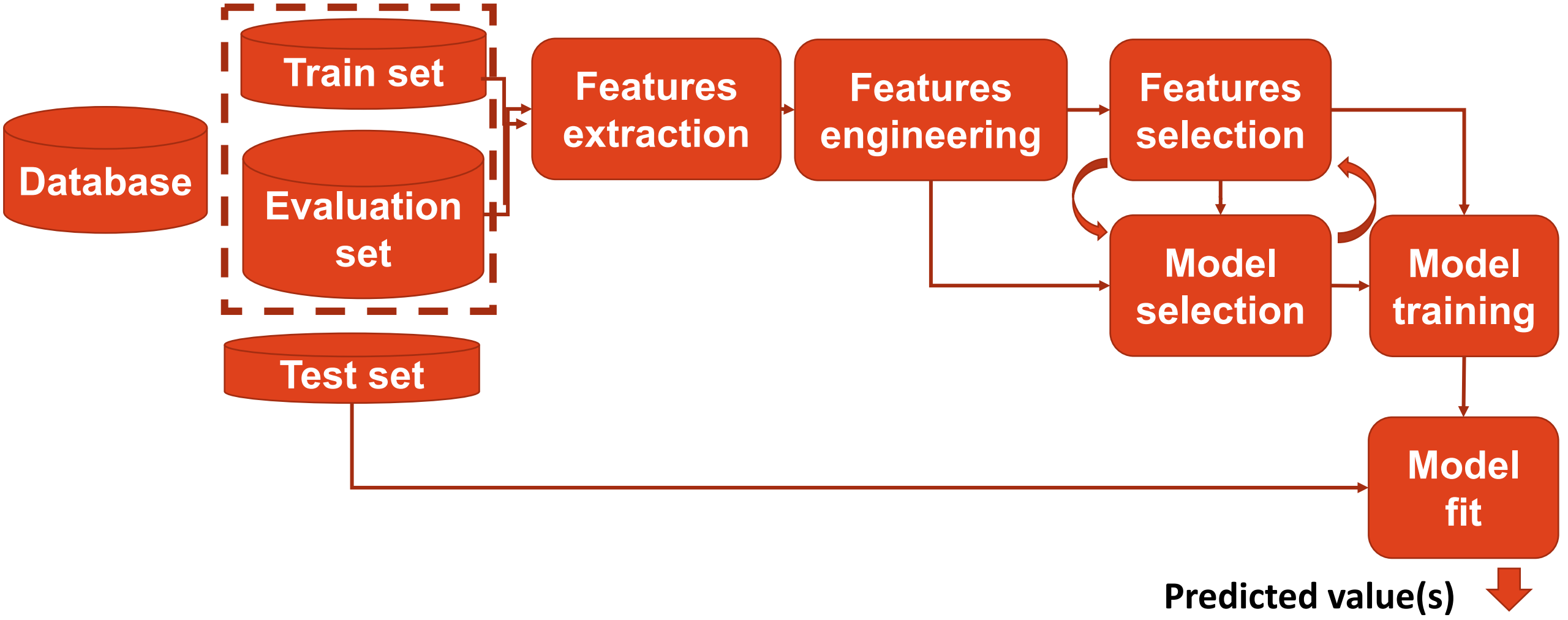
- Increase the model complexity
- Increase the complexity / performance of your features
- Remove unnecessary features

OVERFITTING



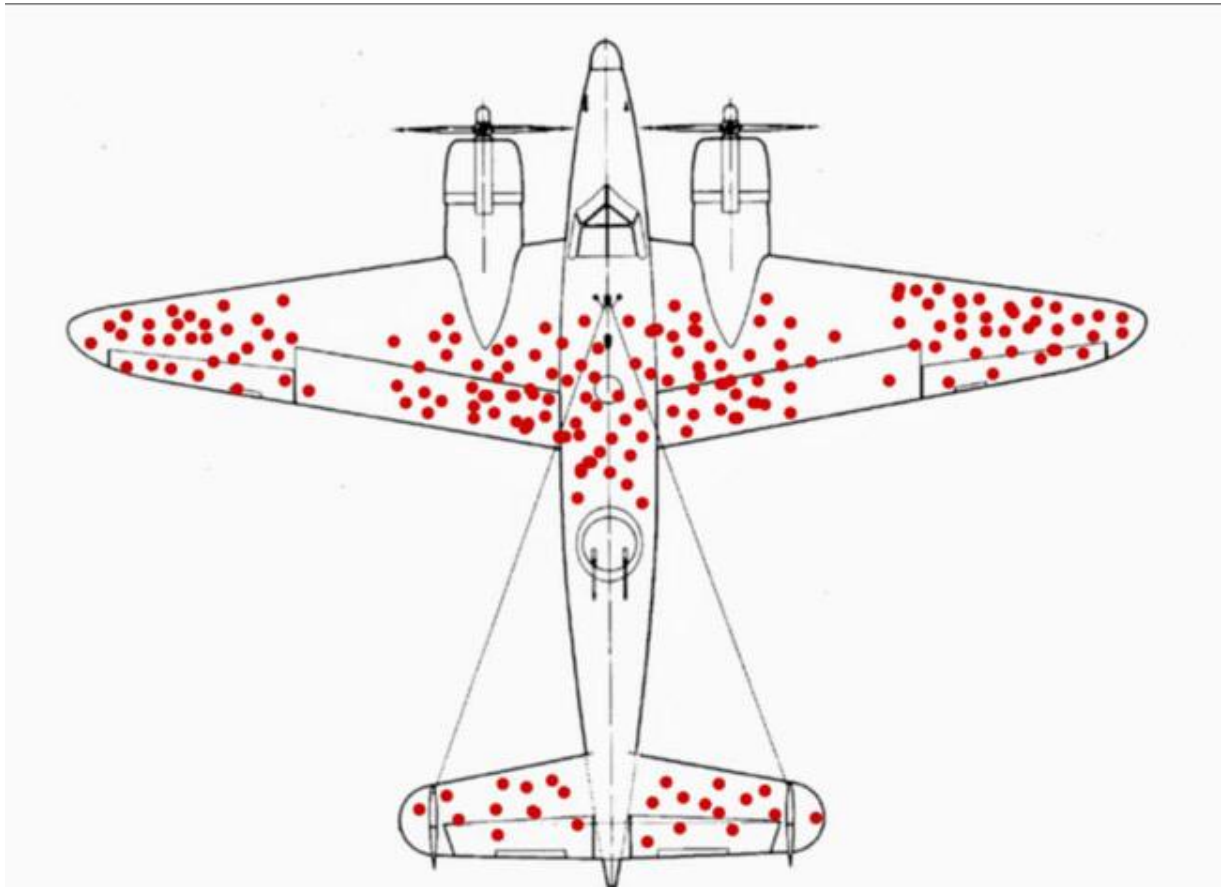
- Add more data
- Reduce the model complexity
- Add techniques against overfitting:
  - regularizations, normalizations, noise
- Remove unnecessary features

A classical Machine Learning flow



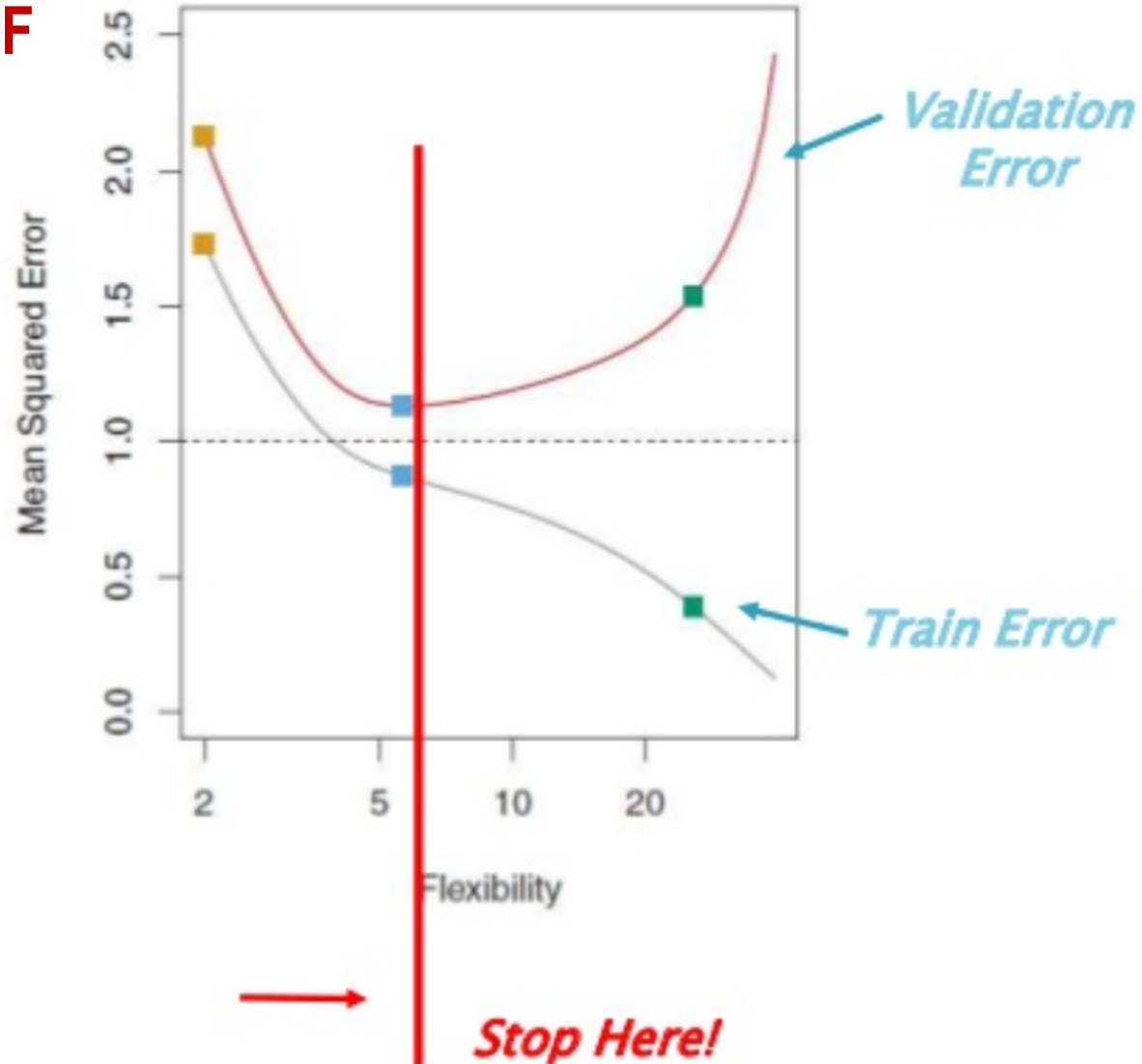
Databases used on training the model

Center for Naval Analyses to minimize the plane losses during next missions



Generalization

BIAS / VARIANCE TRADE-OFF



A classical Machine Learning flow

COMPONENTS

- Feature extraction - Today
- Machine Learning algorithms:
  - Classical approaches – Course 2
  - Deep Learning – Course 3
- Performance measurement – Course 4
- Hands-on – Course 5

FEATURE EXTRACTION

Feature extraction

The algorithms we used are very standard for Kagglers. […] We spent most of our efforts in feature engineering.

- Xavier Conort, on "Q&A with Xavier Conort" on winning the Flight Quest challenge on Kaggle

Better features means flexibility.

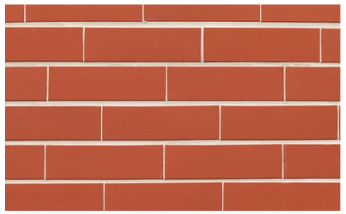
Better features means simpler models.

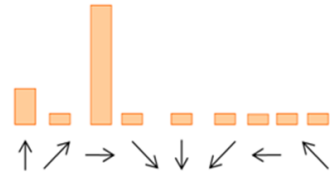
Better features means better results.

Feature extraction

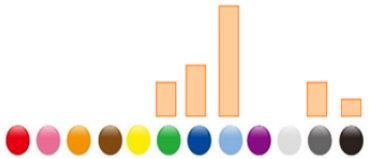
IMAGES

pixels

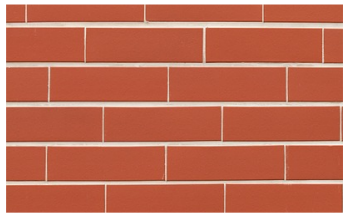


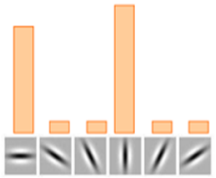


edges



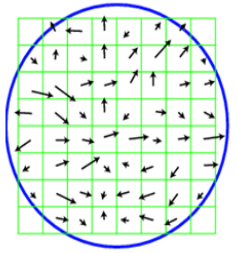
color





texture

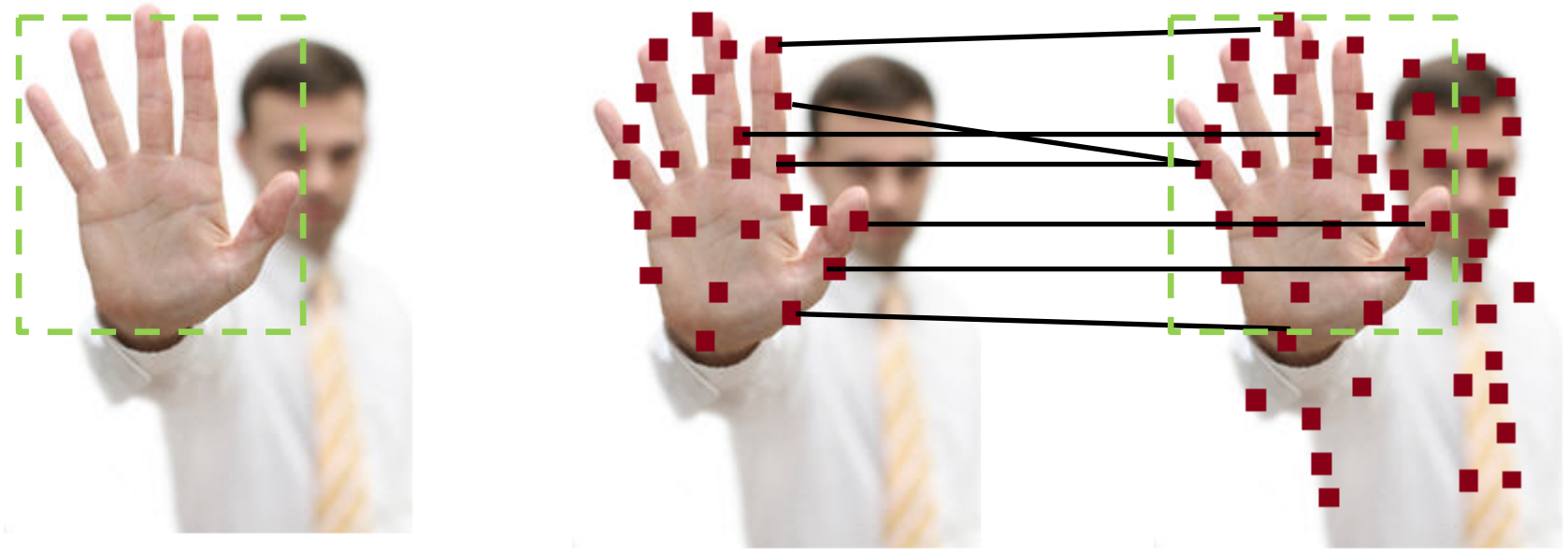
shape



keypoints

Feature extraction

KEYPOINTS



Feature extraction

KEYPOINTS

General algorithm

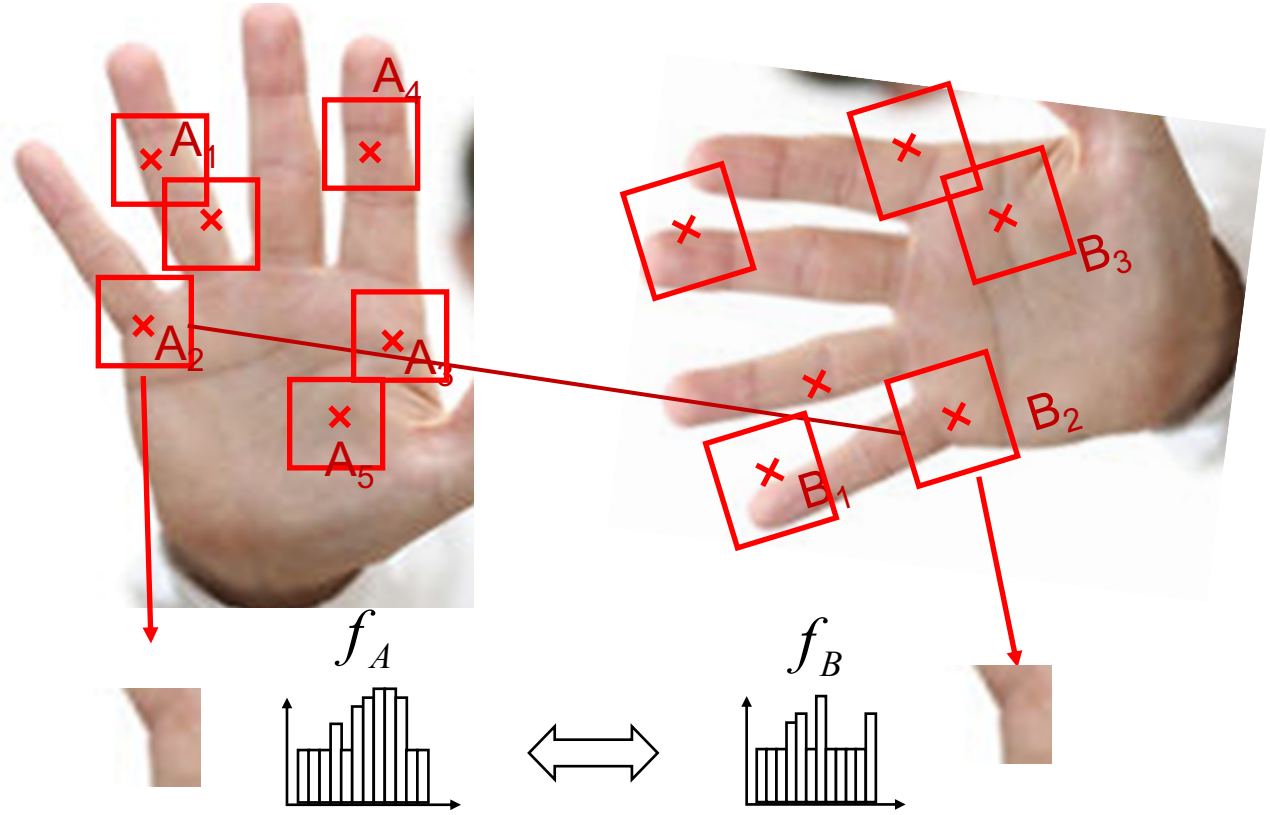
(1) detects the regions that contains the keypoints;

(2) for each keypoint we need to define a neighborhood and extract a descriptor;

(3) compute the distance between the keypoints extracted from the object template and the test dataset.

Feature extraction

KEYPOINTS



Feature extraction

KEYPOINTS

Objectives:

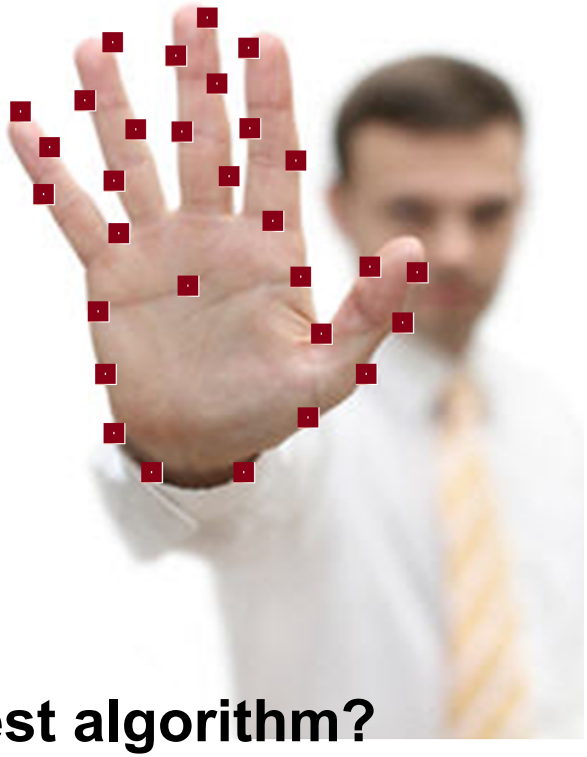
- Repetitivity:
  - invariant to translation, rotation, scale change;
  - invariant la oclusions;
  - Invariant to ilumination changes.
- Precise localization;
- Relevant content.

Feature extraction

KEYPOINTS

Dense extraction

Selective extraction



Which is the best algorithm?

Feature extraction

KEYPOINTS

Region dimension

Dimensiune

Low region area(10x10)

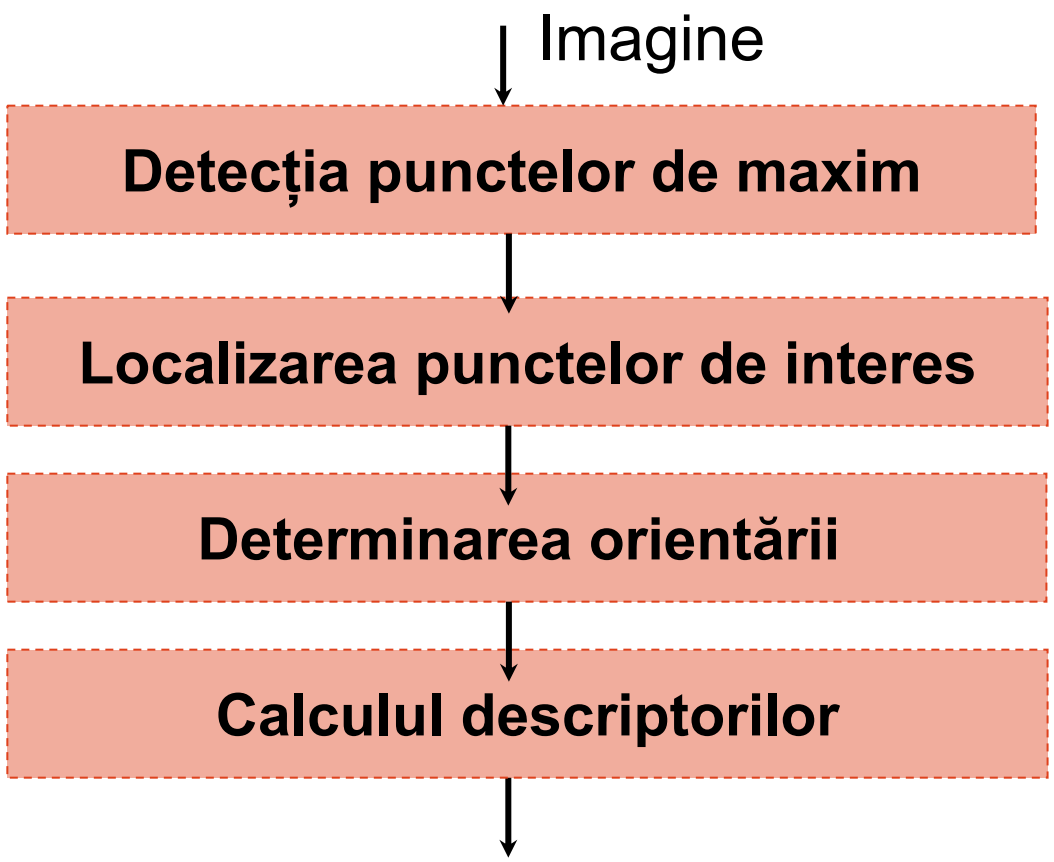
- Robust to oclusions;
- More interest points.

Large region area (40x40)

- More errors (one keypoint can be part of more objects);
- Description is more precise.

Feature extraction

KEYPOINTS **Algoritmul SIFT (Scale-invariant feature transform)**



- identificarea potențialelor puncte de interes;
- localizarea candidaților și eliminarea punctelor duplicate;
- identificarea orientării dominate;
- construcția unui descriptor bazat pe histograma de gradienți din vecinătatea punctelor de interes.

Feature extraction

KEYPOINTS

Algoritmul SIFT

Cum se pot detecta locații care sunt invariante la schimbări de scală ale imaginii?

Soluție: utilizarea funcției de diferențe de gausiene (DoG).

- Pentru o imagine *I(x,y)*, fie o reprezentare liniară a acesteia:

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y)$$

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}$$

- Se caută valorile de minim/maxim local pentru o serie de diferențe de gausiene:

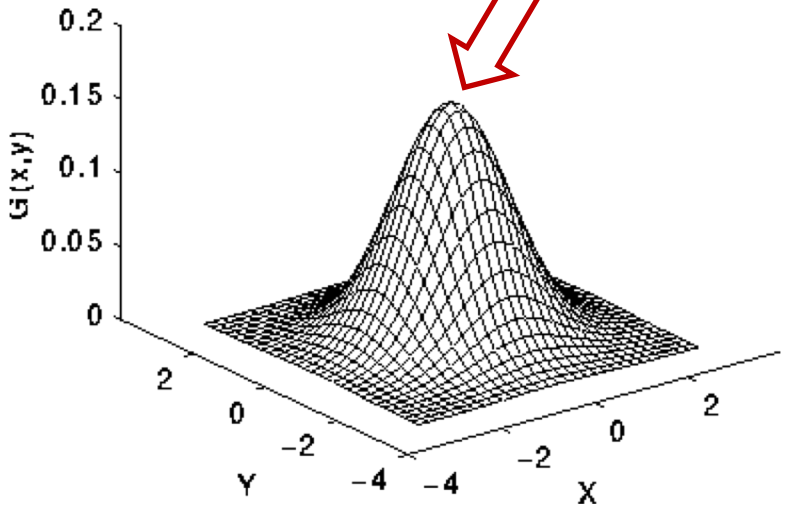
$$D(x, y, \sigma) = (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y)$$
$$= L(x, y, k\sigma) - L(x, y, \sigma)$$

Feature extraction

FILTRU GAUSSIAN - REAMINTIRE

σ = 1, W = 5

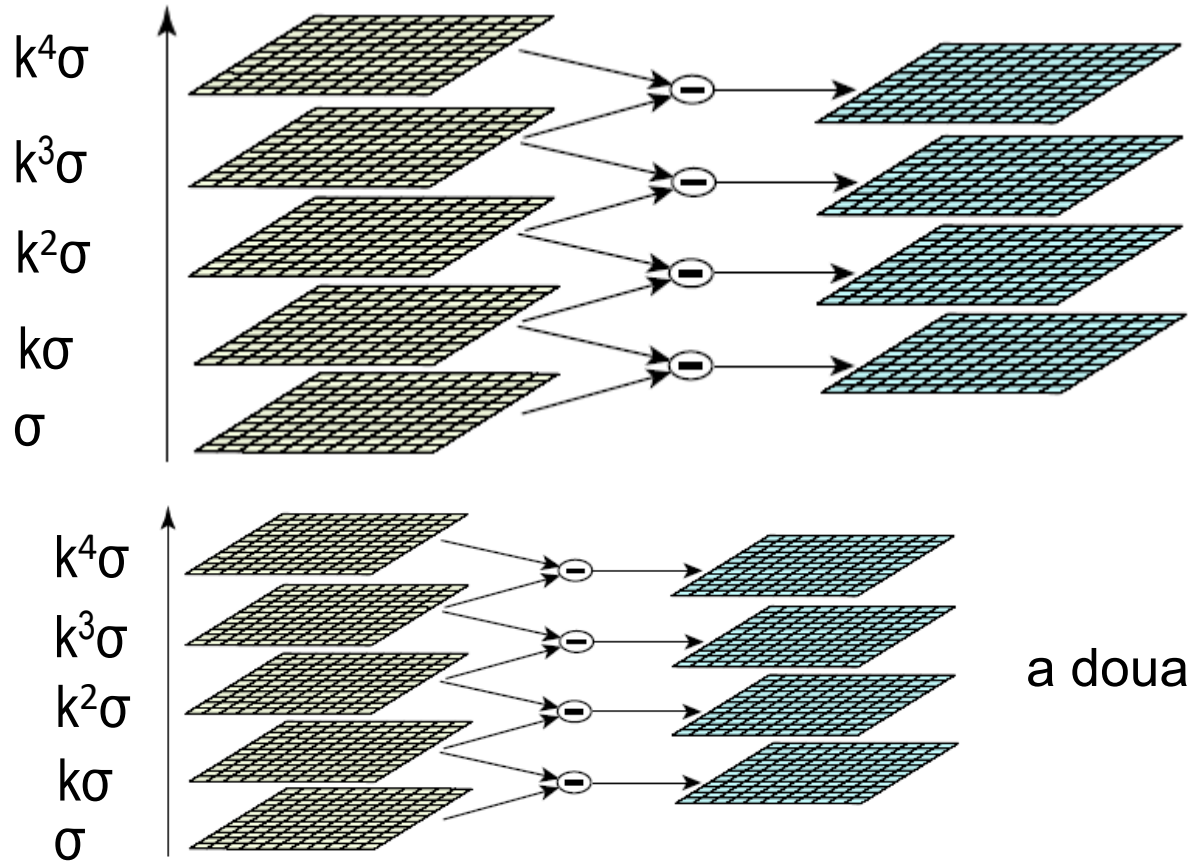
0.00296902	0.01330621	0.02193823	0.01330621	0.00296902
0.01330621	0.0596343	0.09832033	0.0596343	0.01330621
0.02193823	0.09832033	0.16210282	0.09832033	0.02193823
0.01330621	0.0596343	0.09832033	0.0596343	0.01330621
0.00296902	0.01330621	0.02193823	0.01330621	0.00296902



[https://github.com/imironica/IVOM-Demo/blob/master/IVOM_Demo/KeypointsDetector/compute_gaussian_filter.py]

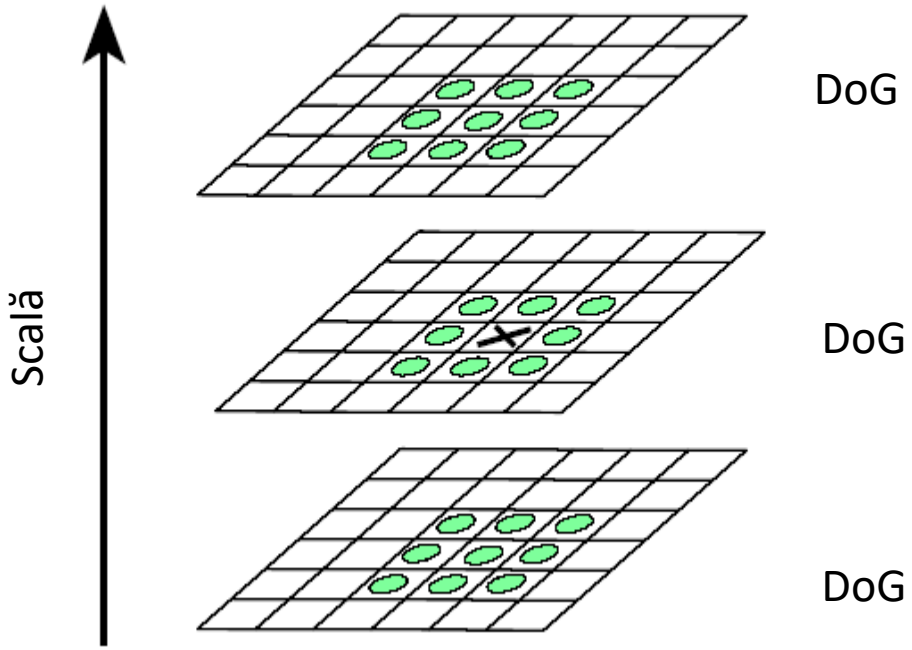
Feature extraction

Algoritmul SIFT - Detecția punctelor de extrem



Feature extraction

Algoritmul SIFT - Detecția punctelor de extrem



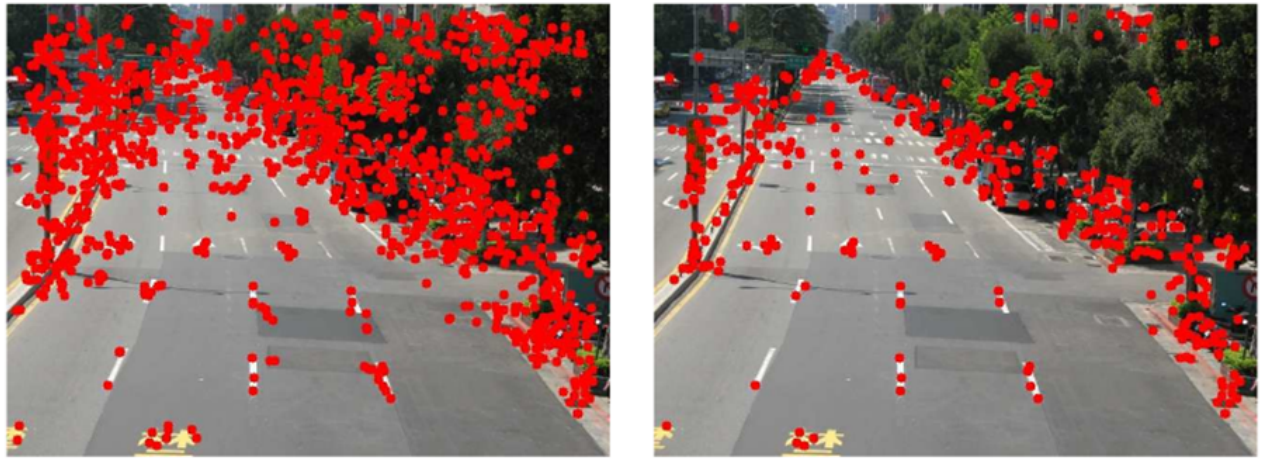
Dacă ***X*** este *cel mai mare* sau *cel mai mic* dintre toți vecinii, atunci ***X*** este denumit ***punct de interes (keypoint)***.

Feature extraction

Algoritmul SIFT - Detecția punctelor de extrem

(a) puncte de interes înainte de aplicarea pragului;

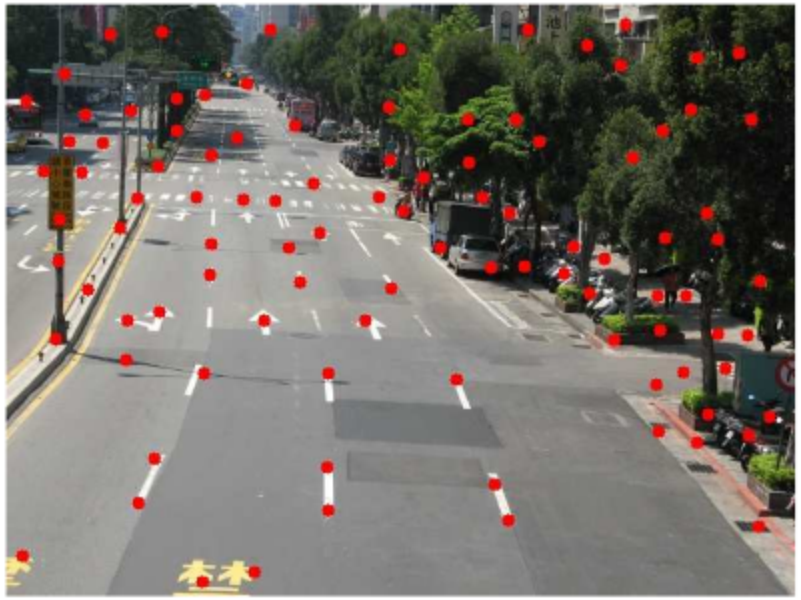
(b) puncte de interes după aplicarea pragului;



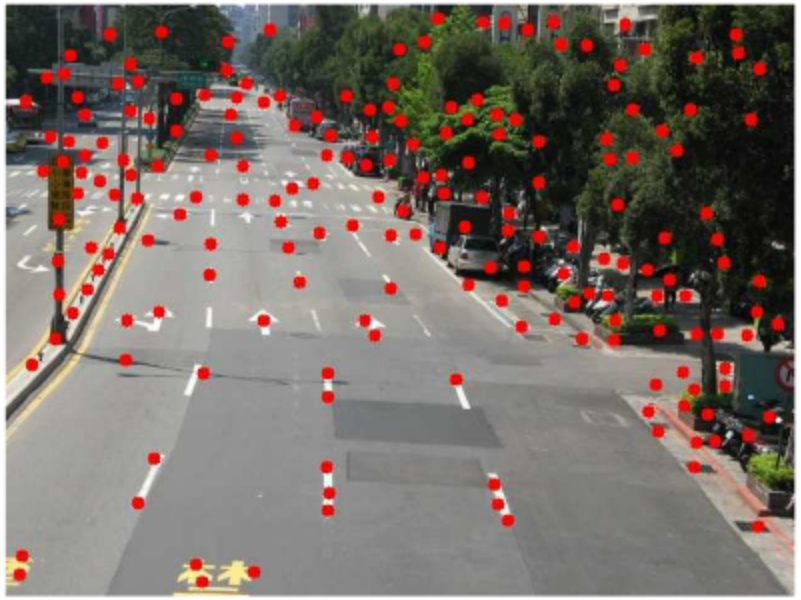
(a) (b)

În funcție de selecția pragurilor anterioare vom avea un număr mai mare / mai mic de puncte de interes.

Feature extraction



(a)



(b)

(a) Top 100 keypoints.
(b) Top 200 keypoints

Feature extraction

1. Se preia o fereastră de 16 x16 în jurul punctului de interes.
2. Se împarte în 4x4 celule.
3. Se calculează o histogramă de gradienți pe 8 orientări pentru fiecare celulă.

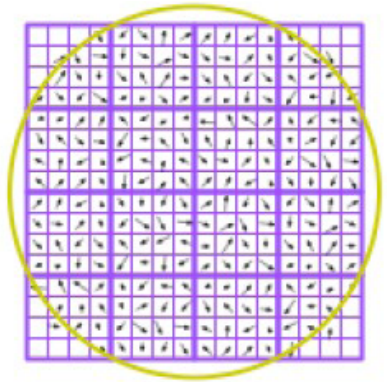
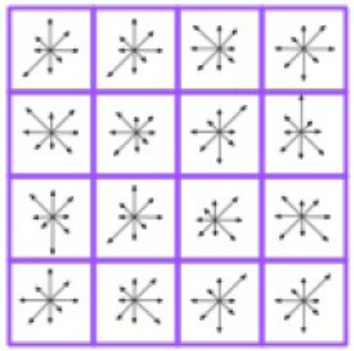


Image gradients



Keypoint descriptor

16 histograme x 8 orientări
= 128 trăsături

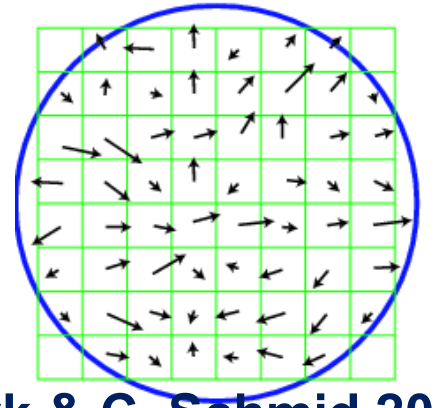
Feature extraction

PCA-SIFT

- Algoritmul este identic cu SIFT, mai puțin pasul 4 (calculul descriptorului);

- În loc să se utilizeze histogramele ponderate din algoritmul SIFT clasic, se calculează gradienții (verticali / orizontali) locali pe o suprafață de 41x41 pixeli (2 pixeli reprezintă conturul);

- Se concatenează toate valorile într-un singur vector: 2 x 39 x 39 = 3042 elemente.

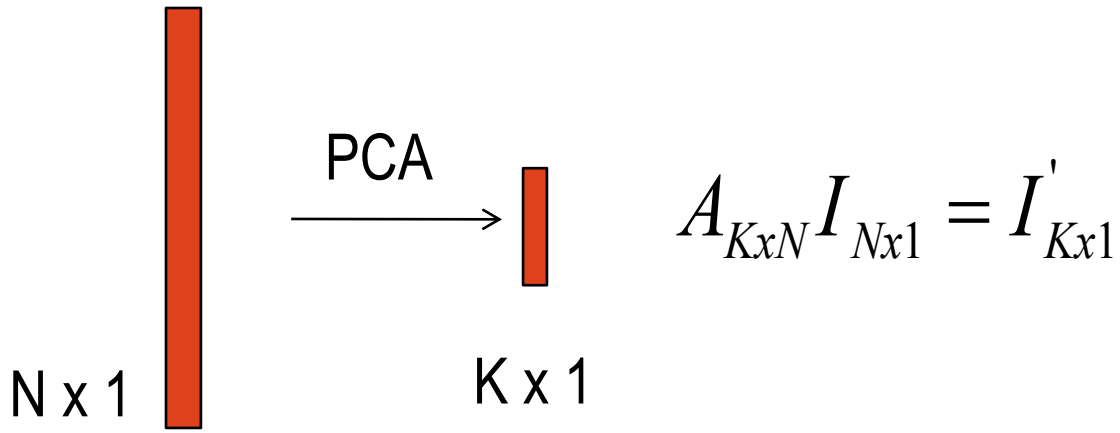


[K. Mikolajczyk & C. Schmid 2005]

Feature extraction

PCA-SIFT

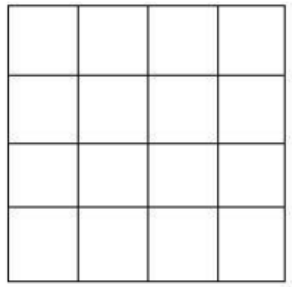
- Se reduce dimensionalitatea vectorului folosind Analiza Componentelor Principale (Principal Component Analysis - PCA) - de ex: de la 3042 la 64 / 36 / 20;

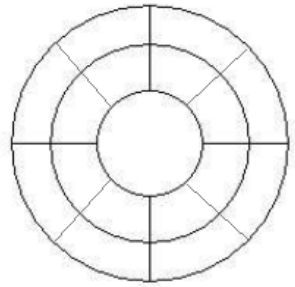


$$A_{KxN} I_{Nx1} = I'_{Kx1}$$

Feature extraction

- GLOH reprezintă un descriptor similar cu PCA-SIFT, singura diferență fiind calculul descriptorului final:
- Se împarte spațiul din jurul punctului de interes în coordonate log-polare;
- Lungime descriptor: (2 x 8 + 1) * 16 = 272;
- Se aplică PCA și se reține 128 elemente.





- **Proprietăți:**
  - Acuratețe mai mare de clasificare;
  - Viteză mai mică;
  - Mai rezistent la zgomot.

[Mikolajczyk & Schmid ‘05]

III. Descriptori locali

UTILIZAREA PUNCTELOR DE INTERES

SURF

Algoritmul **SURF (**Speeded Up Robust Features) îmbunătățește viteza de calcul prin:

(1) utilizarea matricei de aproximare Hessiană

$$H(x,\sigma) = \begin{bmatrix} L_{xx}(\sigma) & L_{xy}(\sigma) \\ L_{yx}(\sigma) & L_{yy}(\sigma) \end{bmatrix}$$

(2) a imaginii integrale în calculul descriptorului.

Ce este o imagine integrală?

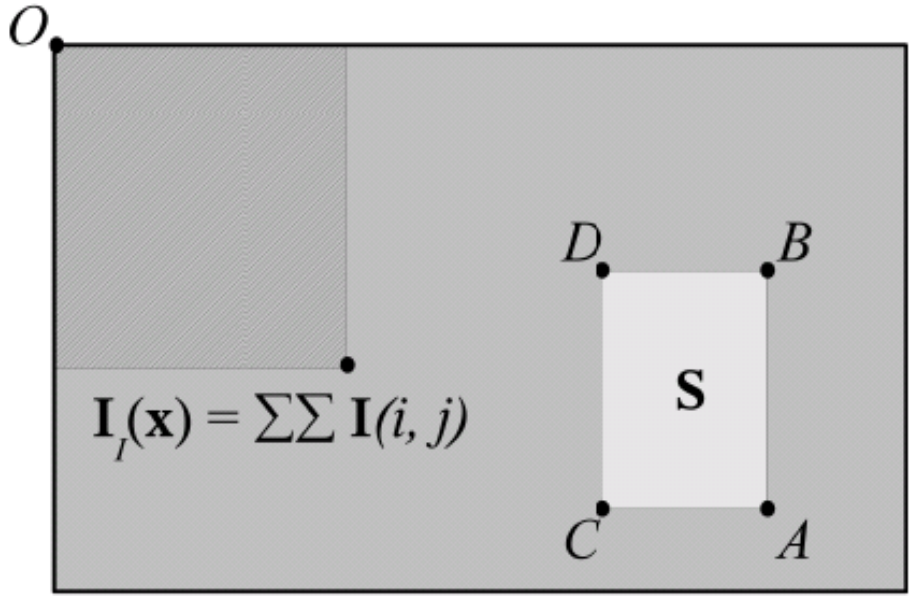
[H. Bay '08]

III. Descriptori locali

UTILIZAREA PUNCTELOR DE INTERES

SURF – Imaginea integrală

Imaginea integrală $I_\Sigma(x,y)$ a unei imagini I(x, y) reprezintă suma tuturor pixelilor din I(x,y) dintr-o regiune dreptunghiulară.



$$I_\Sigma(x,y) = \sum_{i=0}^{i\le x}\sum_{j=0}^{j\le y} I(i,j)$$

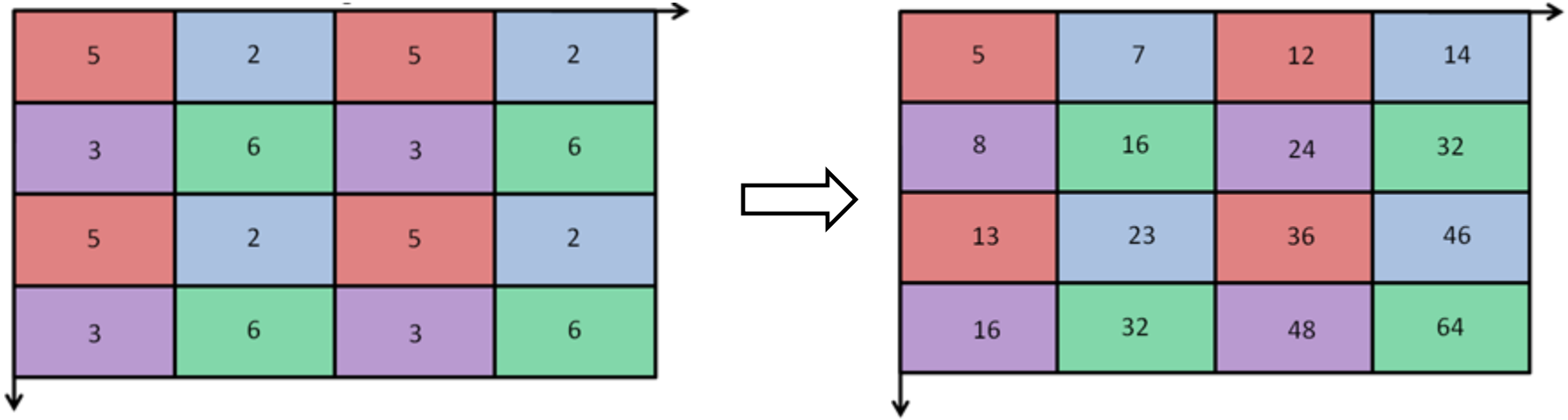
Prin utilizarea imaginii integrale este nevoie de doar patru valori pentru calculul sumei pixelilor dintr-o suprafată dreptunghiulară

$$\mathbf{S}=A-B-C+D$$

III. Descriptori locali

UTILIZAREA PUNCTELOR DE INTERES

SURF – Imaginea integrală



Imagine

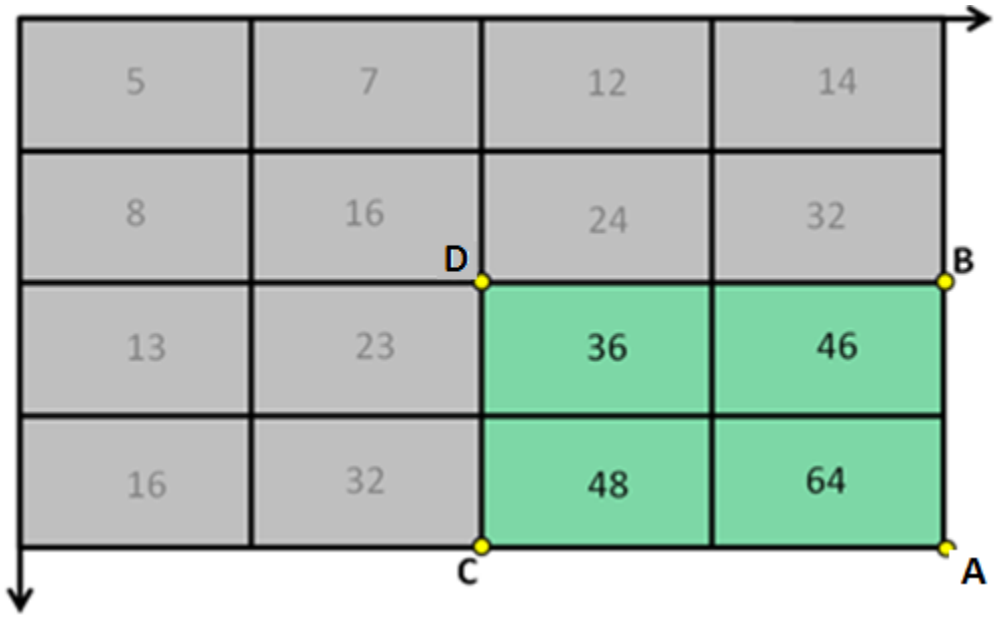
Imagine integrală

$$\mathbf{S}=A-B-C+D$$

III. Descriptori locali

UTILIZAREA PUNCTELOR DE INTERES

SURF – Imaginea integrală



$$\mathbf{S}=A-B-C+D$$

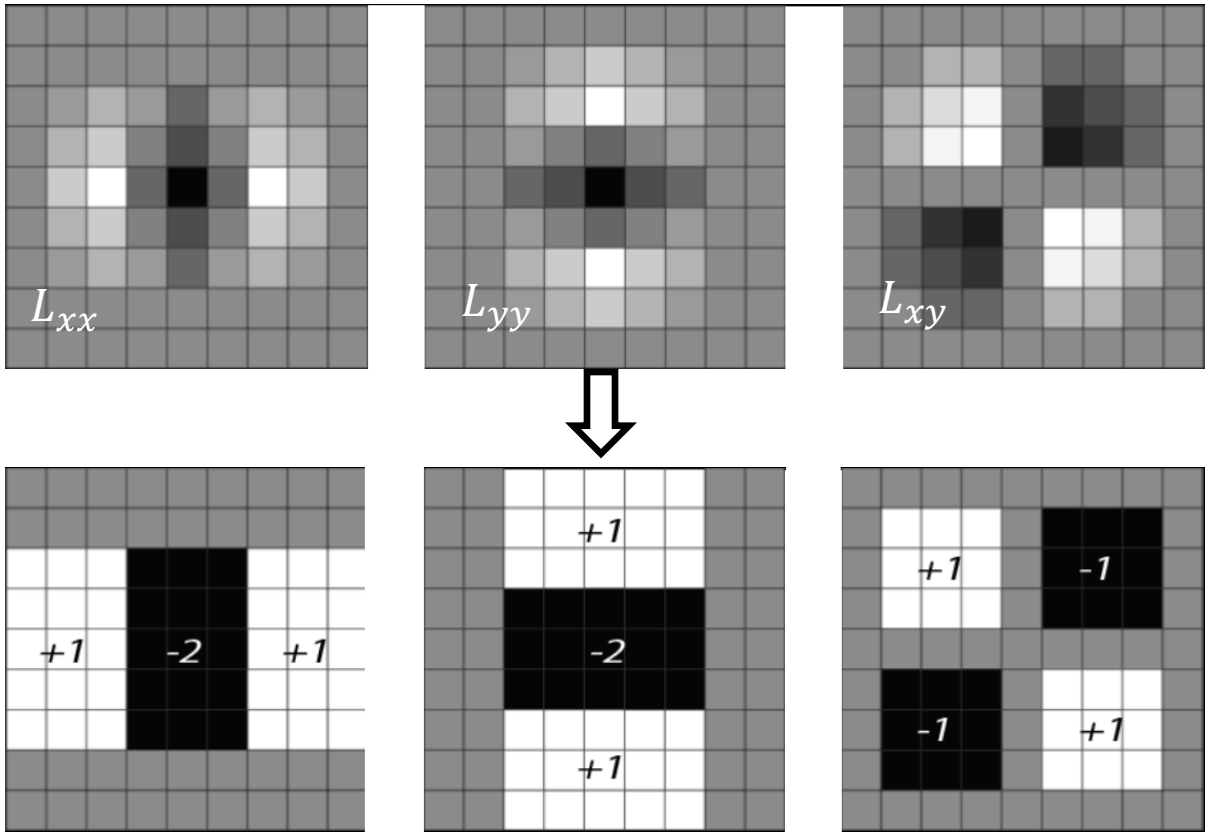
S = 64-32-32+16 = 16

III. Descriptori locali

UTILIZAREA PUNCTELOR DE INTERES

SURF – Aproximarea matricei Hessiene

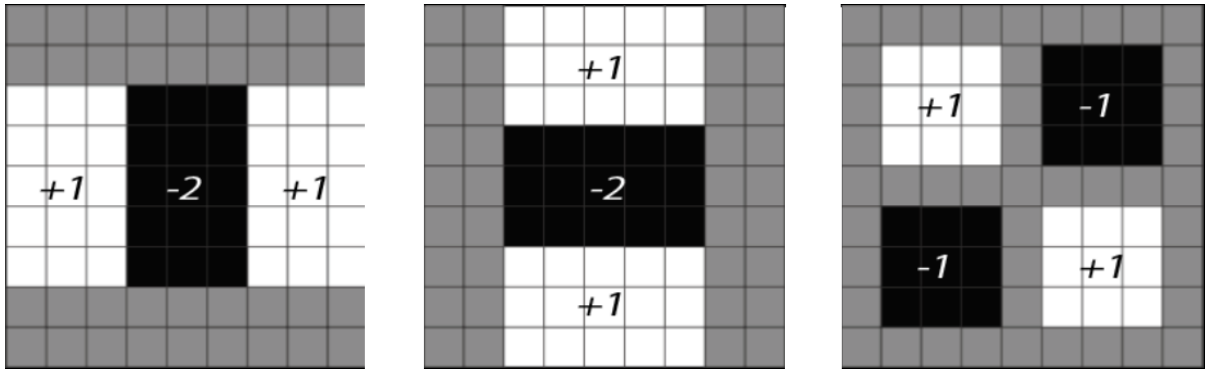
Pentru σ=1.2 (9x9), funcția LoG poate fi aproximată din:



III. Descriptori locali

UTILIZAREA PUNCTELOR DE INTERES

SURF – Aproximarea matricei Hessiene



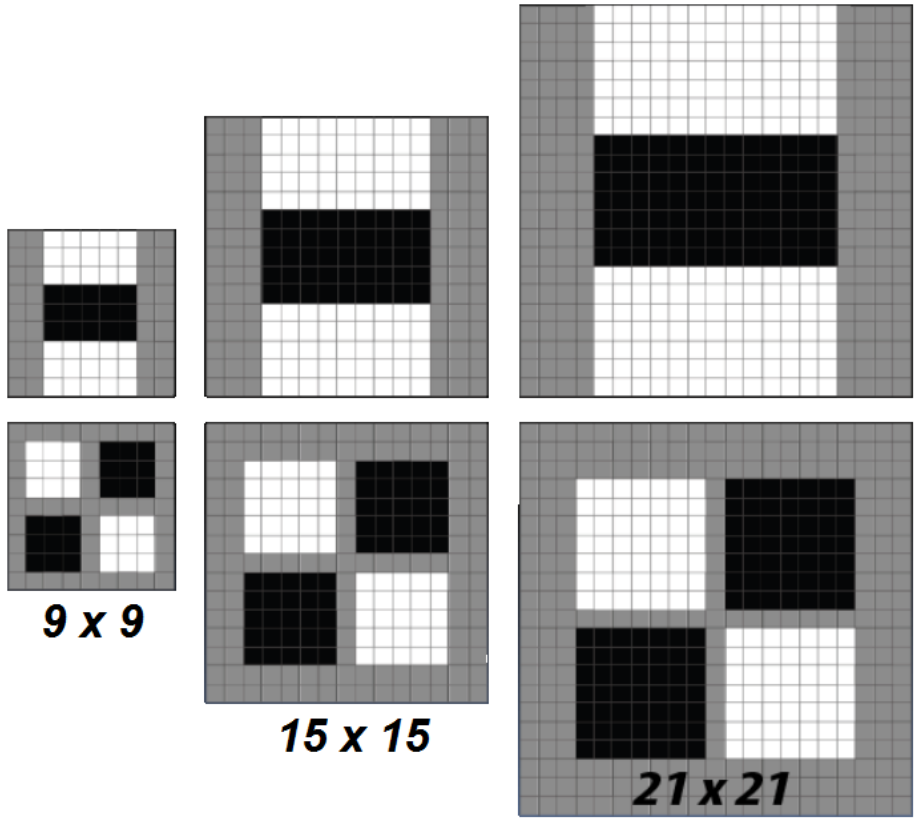
Acestea pot fi foarte ușor calculate utilizând principiul de imagine integrală.

III. Descriptori locali

UTILIZAREA PUNCTELOR DE INTERES

SURF – Aproximarea matricei Hessiene

Similar cu algoritmul SIFT, se aplică filtrul se calculează la mai multe octave (3 sau 4 în funcție de implementare):

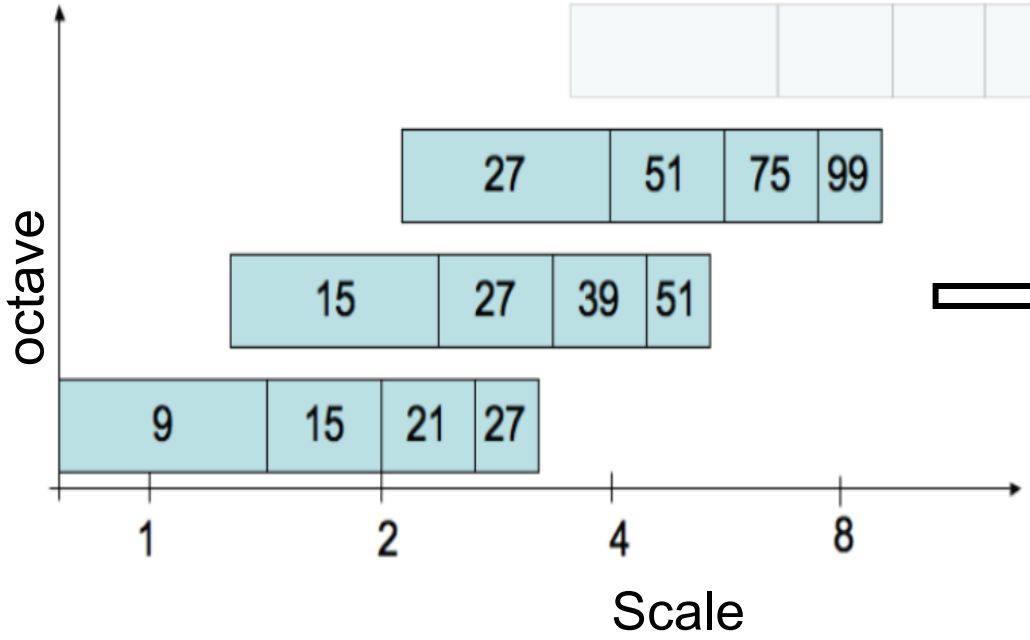


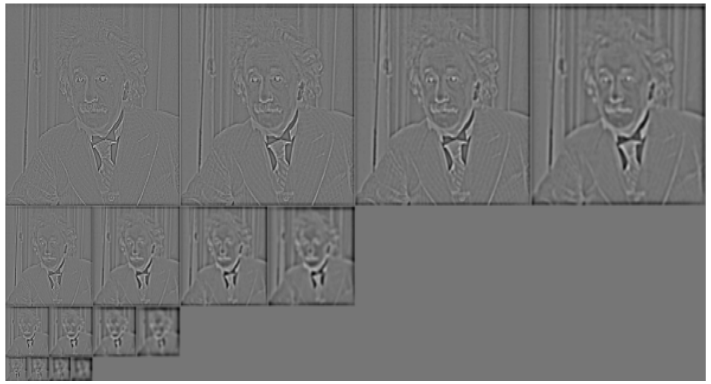
III. Descriptori locali

UTILIZAREA PUNCTELOR DE INTERES

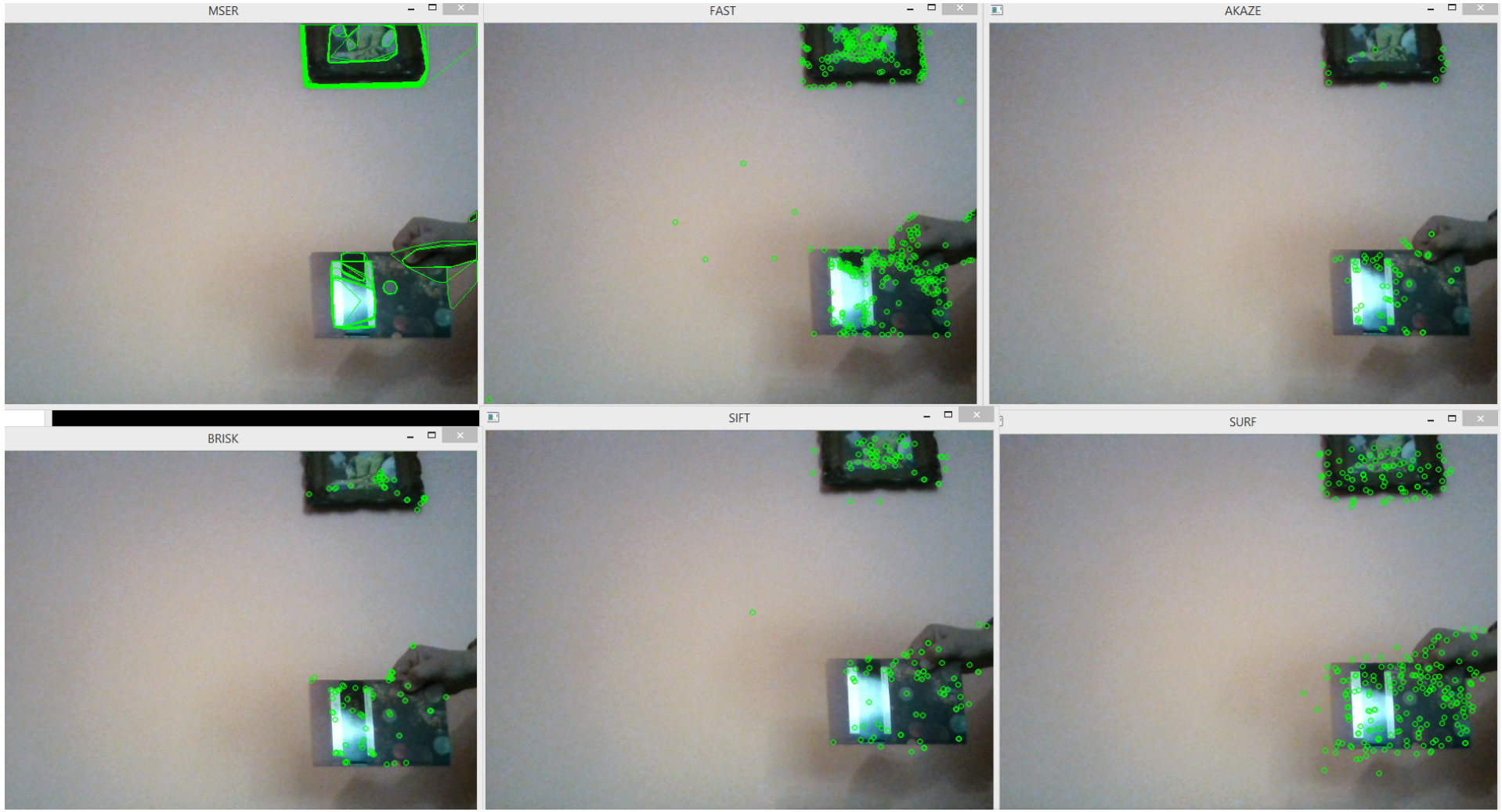
SURF – Aproximarea matricei Hessiene

Similar cu algoritmul SIFT, se aplică filtrul se calculează la mai multe scale (3 sau 4 în funcție de implementare):





Datorită imaginilor integrale, filtrele de orice dimensiune pot fi aplicate cu aceeași viteză!


MSER
FAST
AKAZE
BRISK
SIFT
SURF

III. Descriptori locali

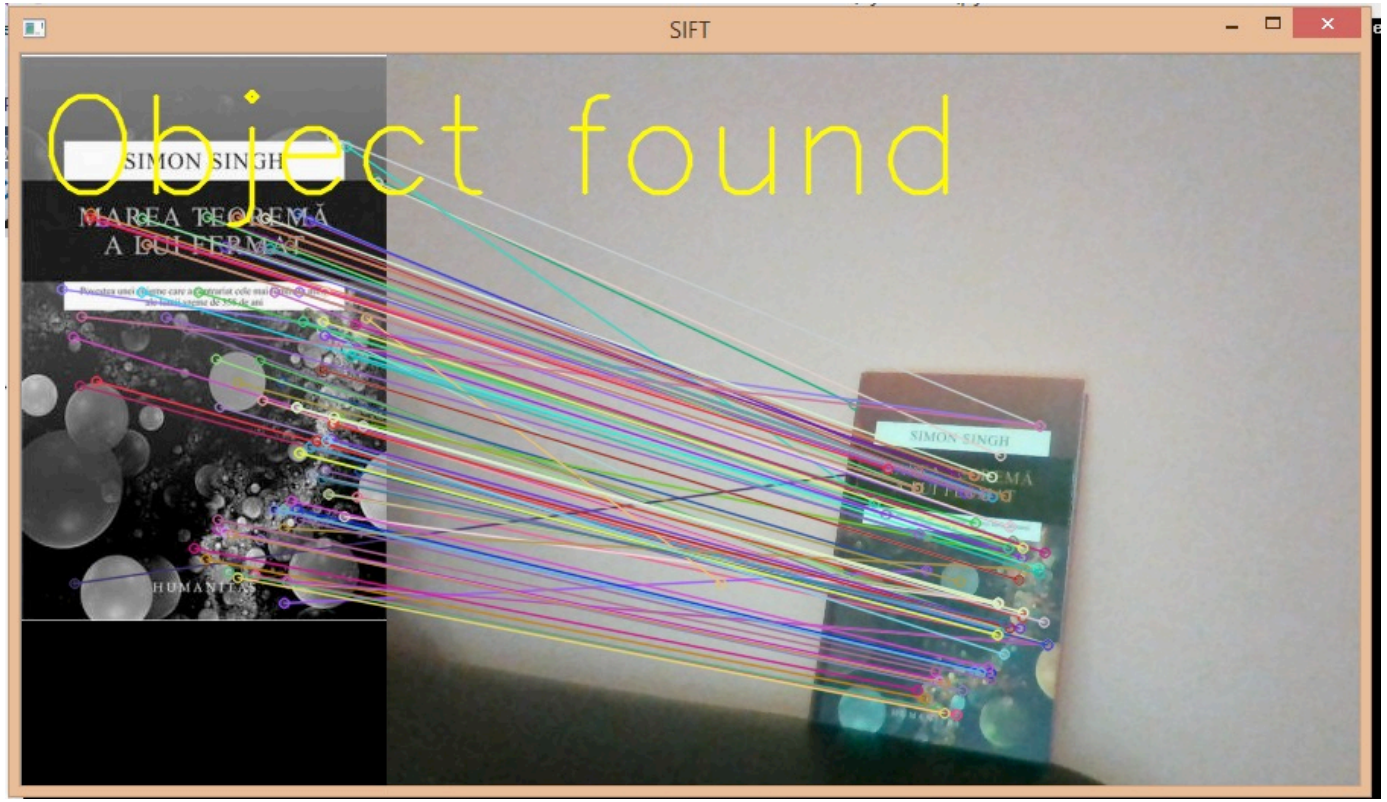
UTILIZAREA PUNCTELOR DE INTERES

Demo

[https://github.com/imironica/IVOM-Demo/tree/master/IVOM_Demo/KeypointsDetector]

III. Descriptori locali

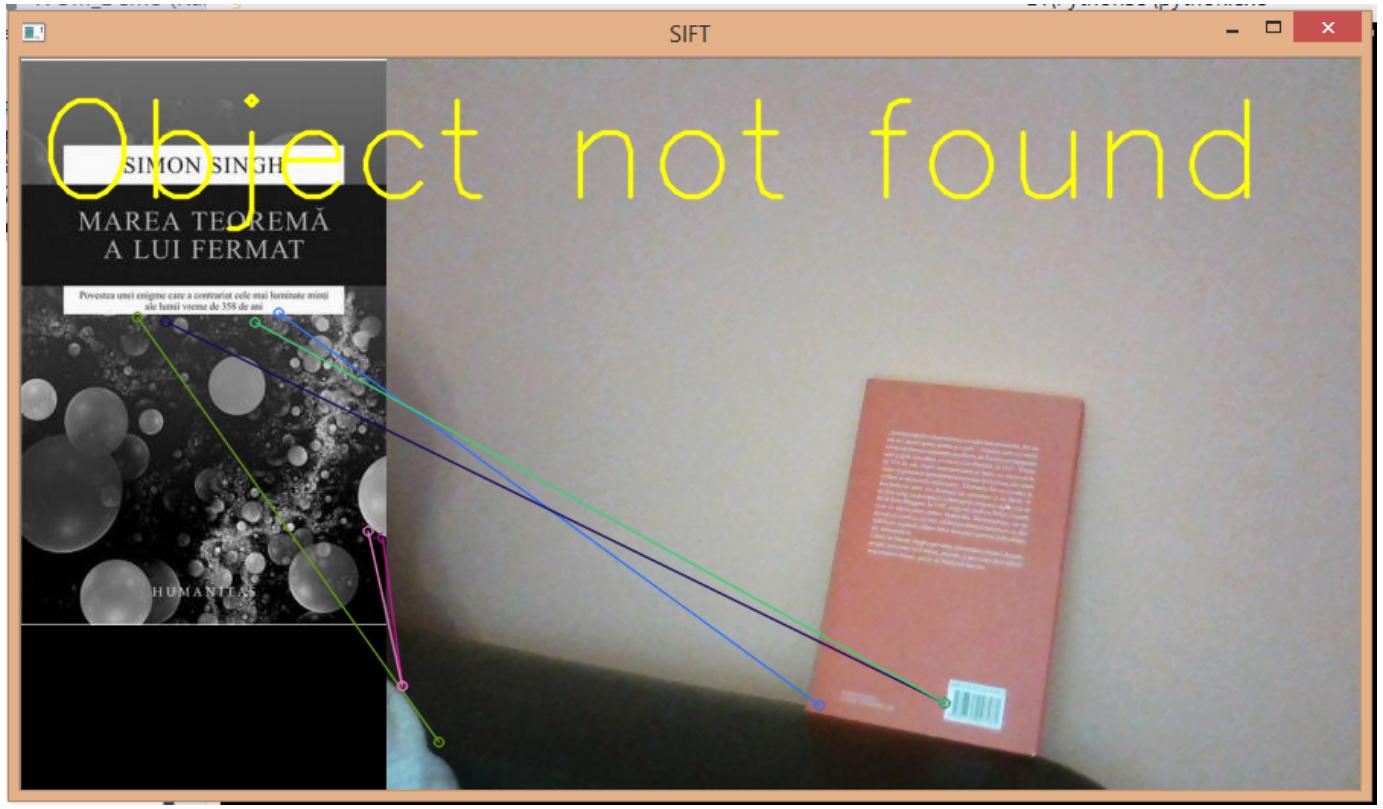
UTILIZAREA PUNCTELOR DE INTERES



[https://github.com/imironica/IVOM-Demo/tree/master/IVOM_Demo/KeypointsDetector]

III. Descriptori locali

UTILIZAREA PUNCTELOR DE INTERES



[https://github.com/imironica/IVOM-Demo/tree/master/IVOM_Demo/KeypointsDetector]

III. Descriptori locali

AGREGAREA PUNCTELOR DE INTERES

Căutarea cu puncte de interes

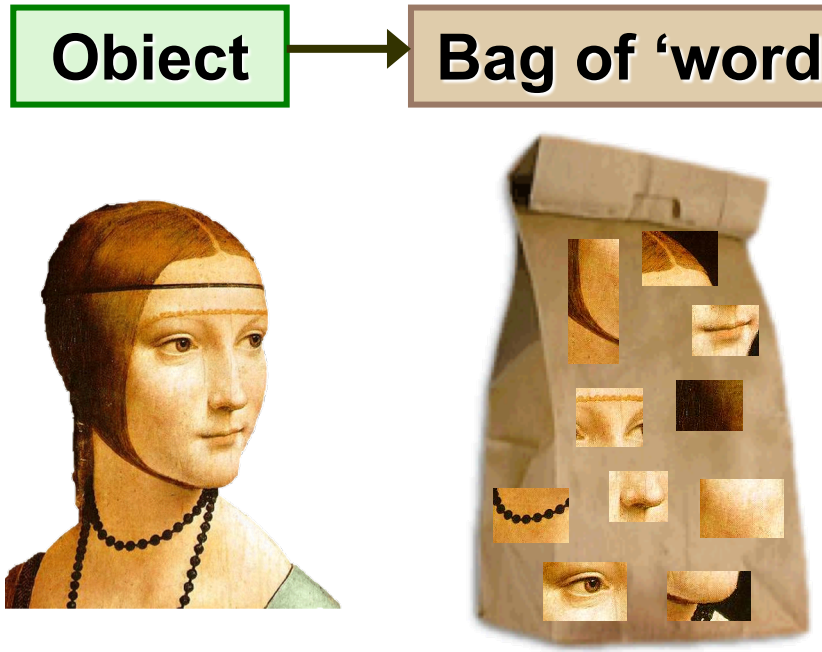
Algoritmul de căutare poate fi unul destul de consumator de resurse de calcul.

Posibilă soluție:

- Agregarea punctelor de interes într-un descriptor global.

III. Descriptori locali

MODELUL BAG-OF-WORDS



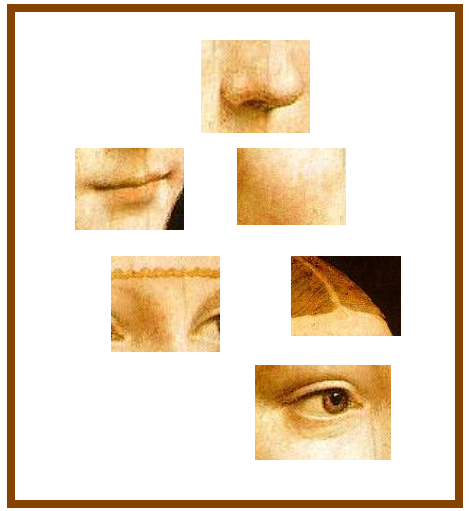
- Extragere trăsături (puncte de interes / descriptori de mișcare)
- Învățarea unui dicționar (vizual / de mișcare)
- Cuantizarea trăsăturilor prin utilizarea vocabularului
- Reprezentarea cadrelor prin utilizarea frecvenței de apariție cuvintelor

[Slide-uri adaptate din prezentările lui Rob Fergus, Svetlana Lazebnik și Noah Snavely]

III. Descriptori locali

MODELUL BAG-OF-WORDS

- Extragere trăsături (puncte de interes / descriptori de mișcare)



III. Descriptori locali

MODELUL BAG-OF-WORDS

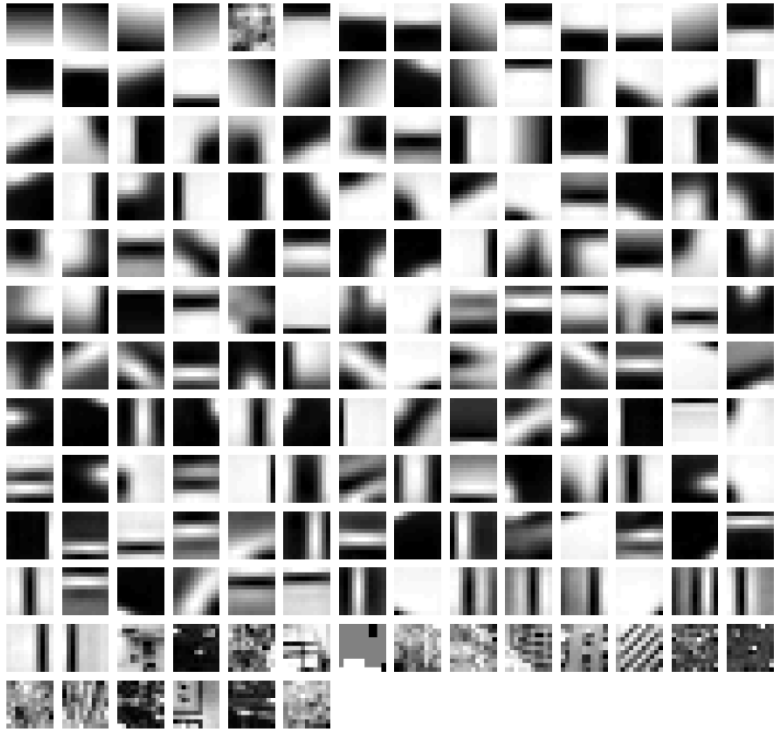
- Învățarea unui dicționar (vizual / de mișcare)

- K-means
- Clusterizare ierarhică
- Gaussian Mixture Model
- Arbori aleatori

III. Descriptori locali

MODELUL BAG-OF-WORDS

- Învățarea unui dicționar vizual



III. Descriptori locali

MODELUL BAG-OF-WORDS

- Cat de mare trebuie să fie un dicționar?

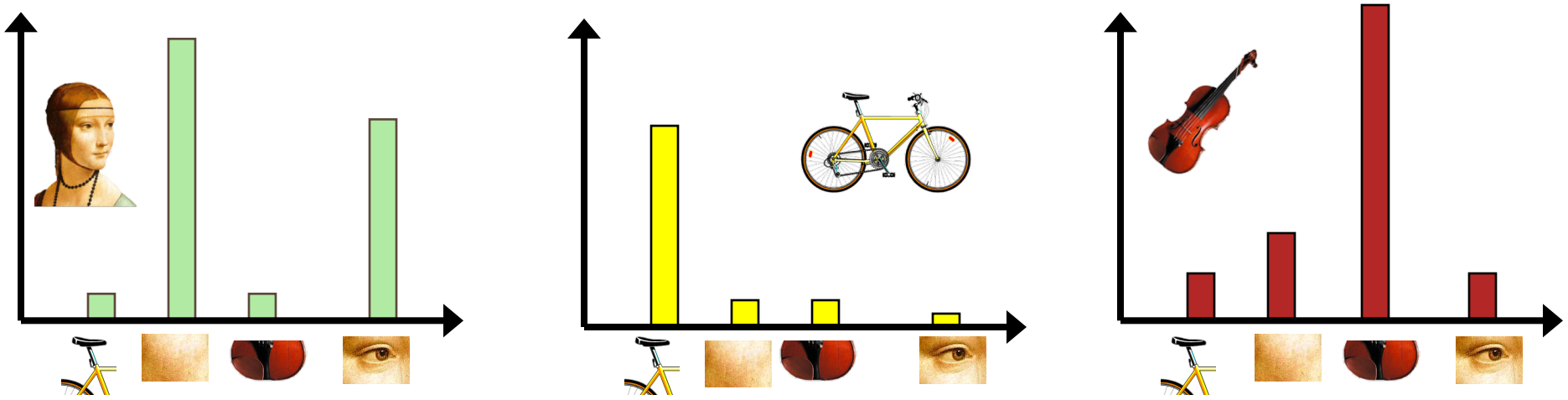
  - Prea mic: numărul de cuvinte vizuale nu vor fi reprezentative pentru toate conceptele
  - Prea mare: supra-învățare (overfitting)

  Există metode care propun de la câteva sute de cuvinte la sute de mii de cuvinte.

III. Descriptori locali

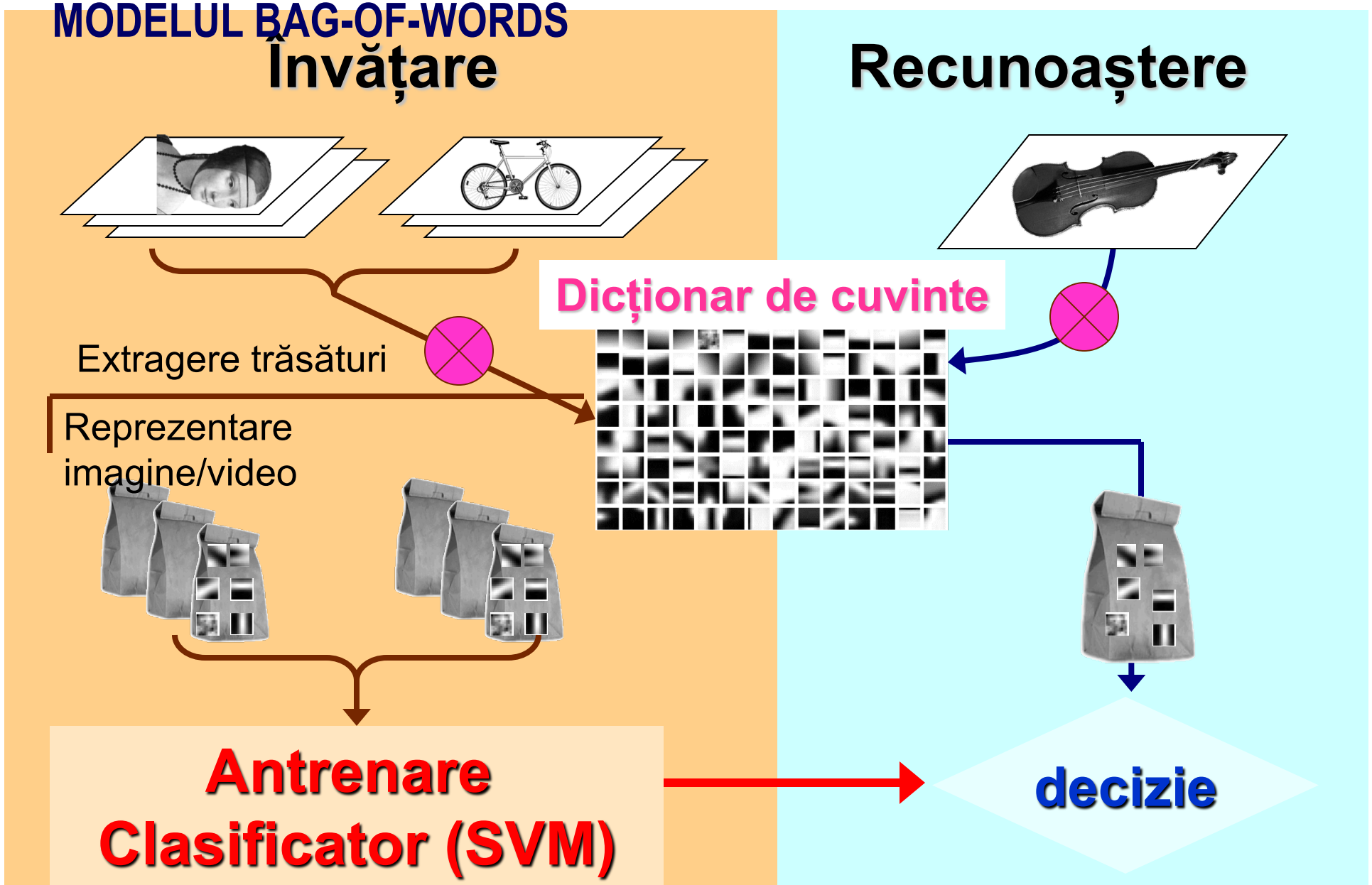
MODELUL BAG-OF-WORDS

- Reprezentarea cadrelor prin utilizarea frecvenței de apariție cuvintelor



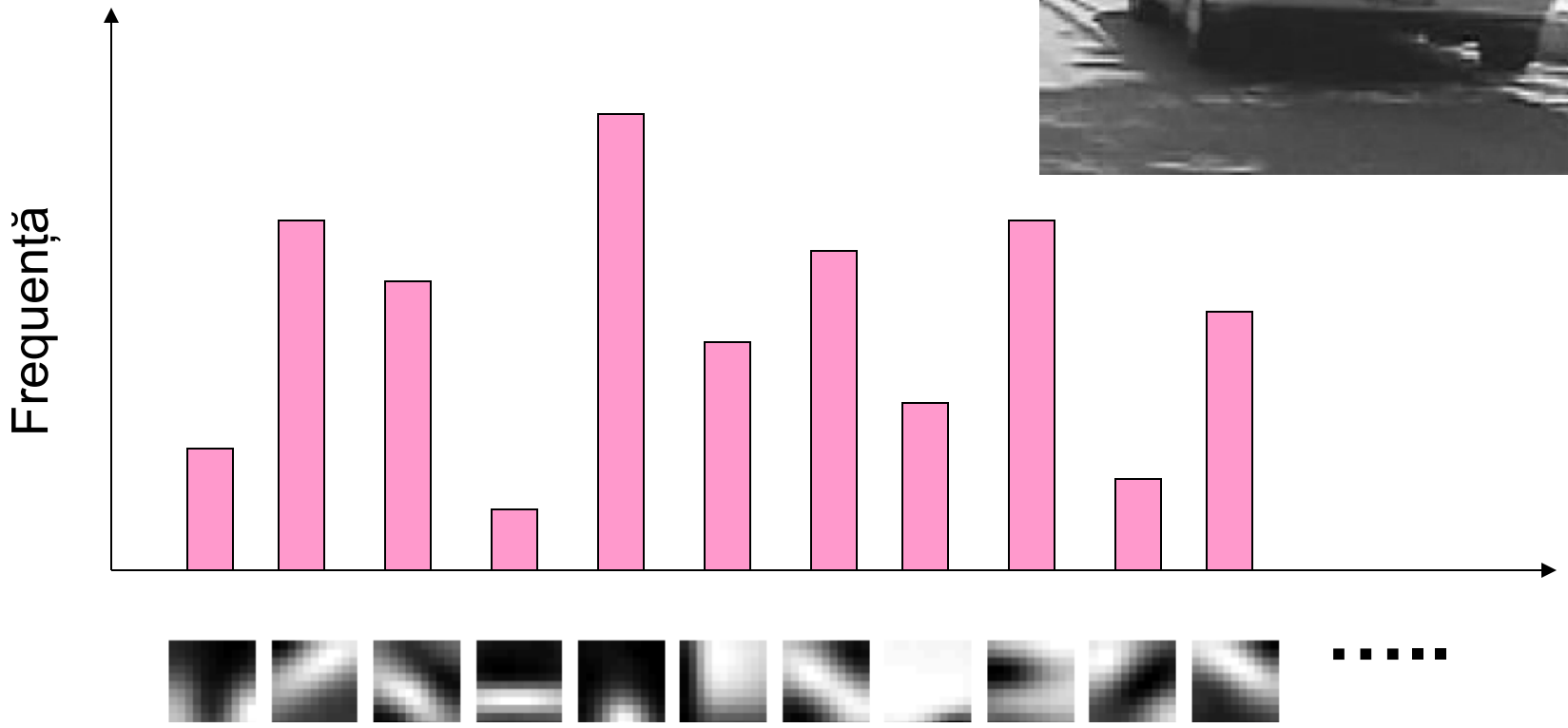
III. Descriptori locali

MODELUL BAG-OF-WORDS



III. Descriptori locali

MODELUL BAG-OF-WORDS



III. Descriptori locali

MODELUL BAG-OF-WORDS

Descriptori - Bag of Words

- Au rezultate bune atunci când obiectele sunt asemănătoare

- Dar ce facem cu scaunele?

III. Descriptori locali

MODELUL BAG-OF-WORDS

Dezavantaje model „Bag of Words”

- nu există nici o metodă riguroasă de reprezentare a distribuției spațiale dintre anumite perechi de cuvinte.
- există multe cuvinte care nu sunt relevante
- procesul de cuantizare a cuvintelor generează zgomot de cuantizare.
- costul computațional crește foarte mult odată cu dimensiunea vocabularului de cuvinte.

III. Descriptori locali

MODELUL FISHER KERNEL

FK vs BoW

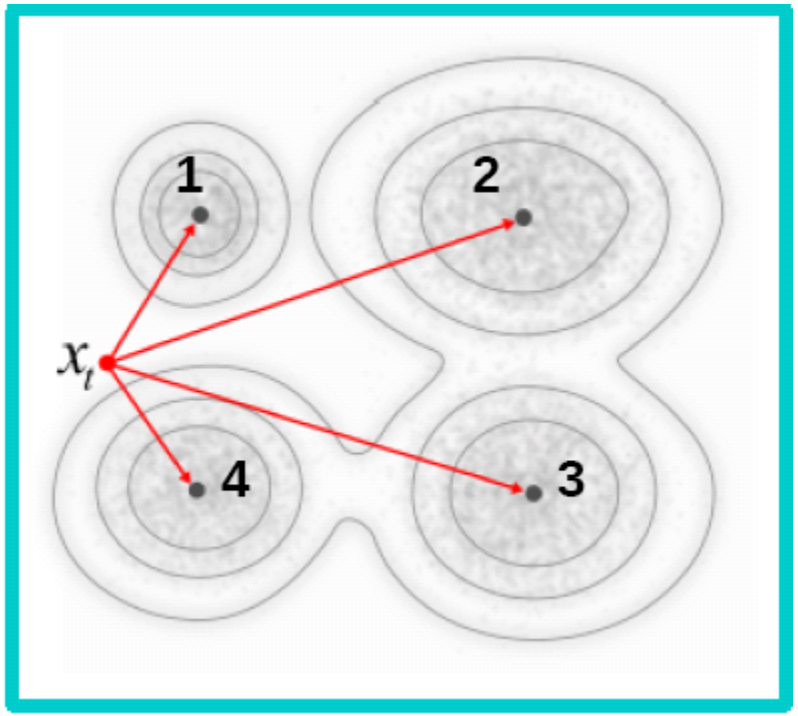
Bag of Words

conține apartenența

fiecărui punct proeminent către

un element al unui dicționar

(histogramă de cuvinte)



Rezultat: D = [0;0;0;1];

Dimensiune: K

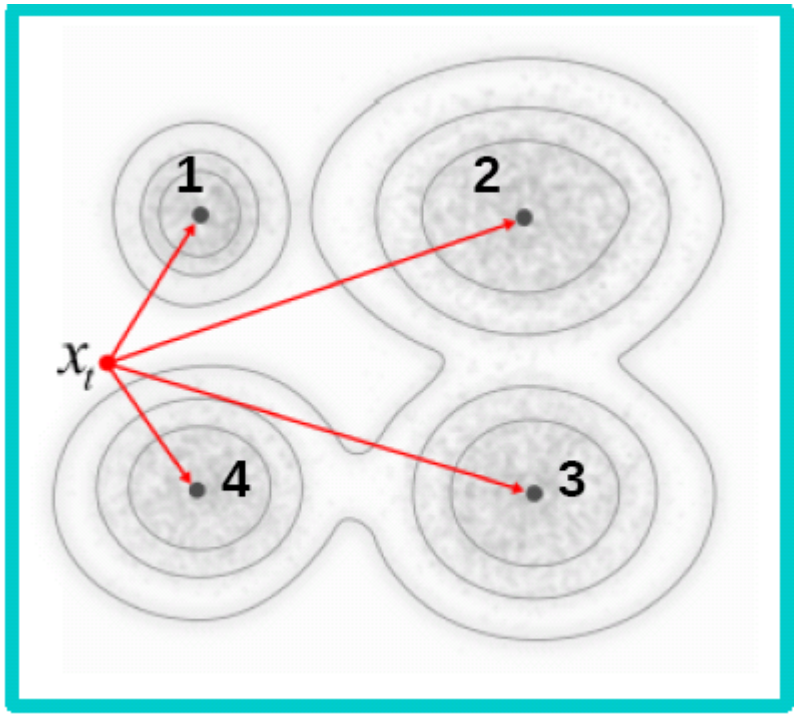
III. Descriptori locali

MODELUL FISHER KERNEL

FK vs BoW

Fisher Kernels

Calculează probabilitățile de apartenență la un cuvânt din dicționar



Rezultat: D = [0.3;0.1;0.1;0.5];

- calculează gradientul mediei și a varianței probabilităților de apartenență la un cuvânt din dicționar.

Dimensiune: 2*D*K

III. Descriptori locali

MODELUL FISHER KERNEL

Îmbunătățiri FK

Normalizare L2

- elimină erorile ce apar din diferența de scală a obiectelor

Normalizare de putere (Power Normalization)

- eliminarea efectului de matrice rară (majoritatea elementelor din FK au valori foarte mici)

$$f(z) = \text{sign}(z)|z|^{\alpha}$$

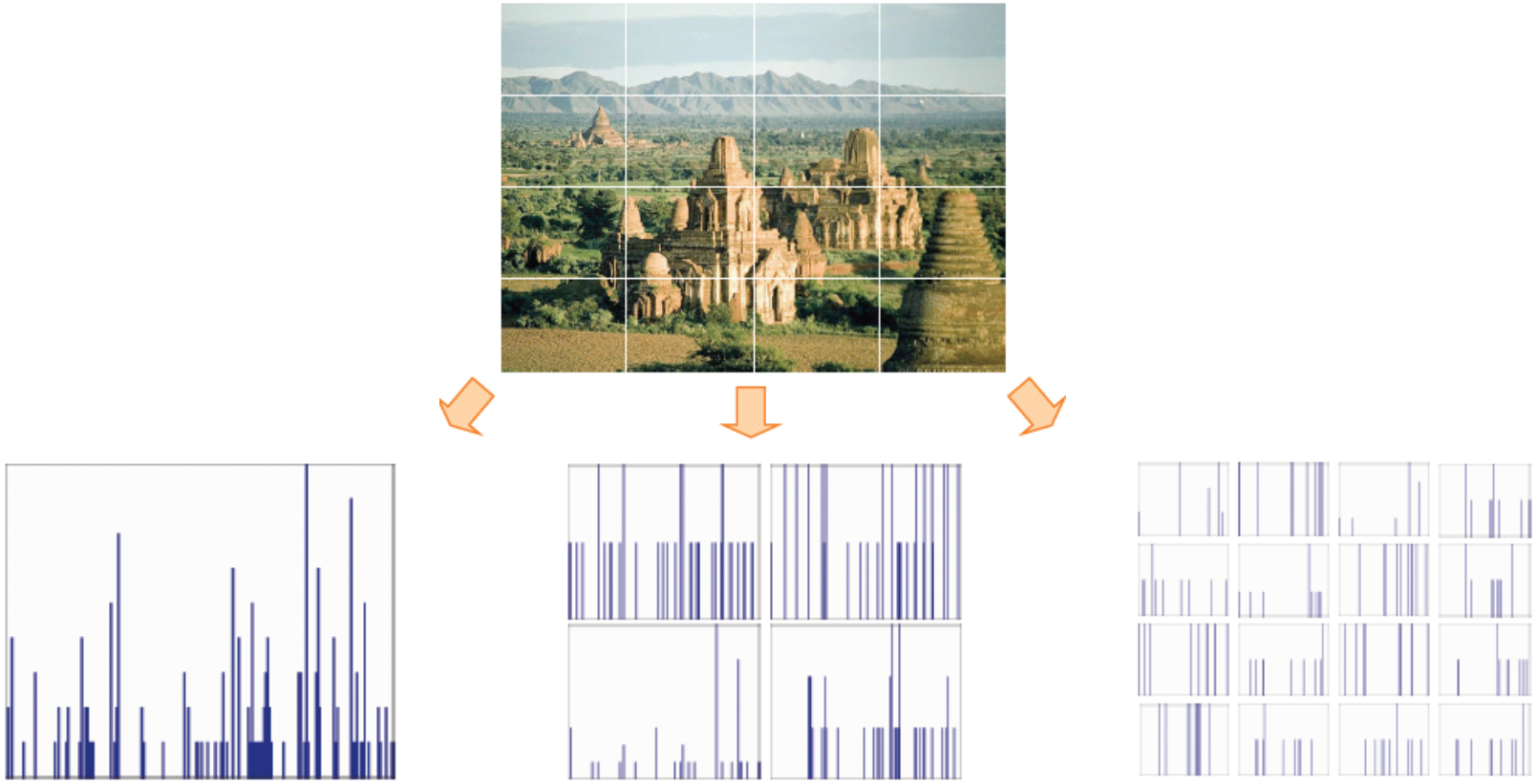
Aplicare Piramide Spațiale

- utilizează informația geometrică a obiectelor

III. Descriptori locali

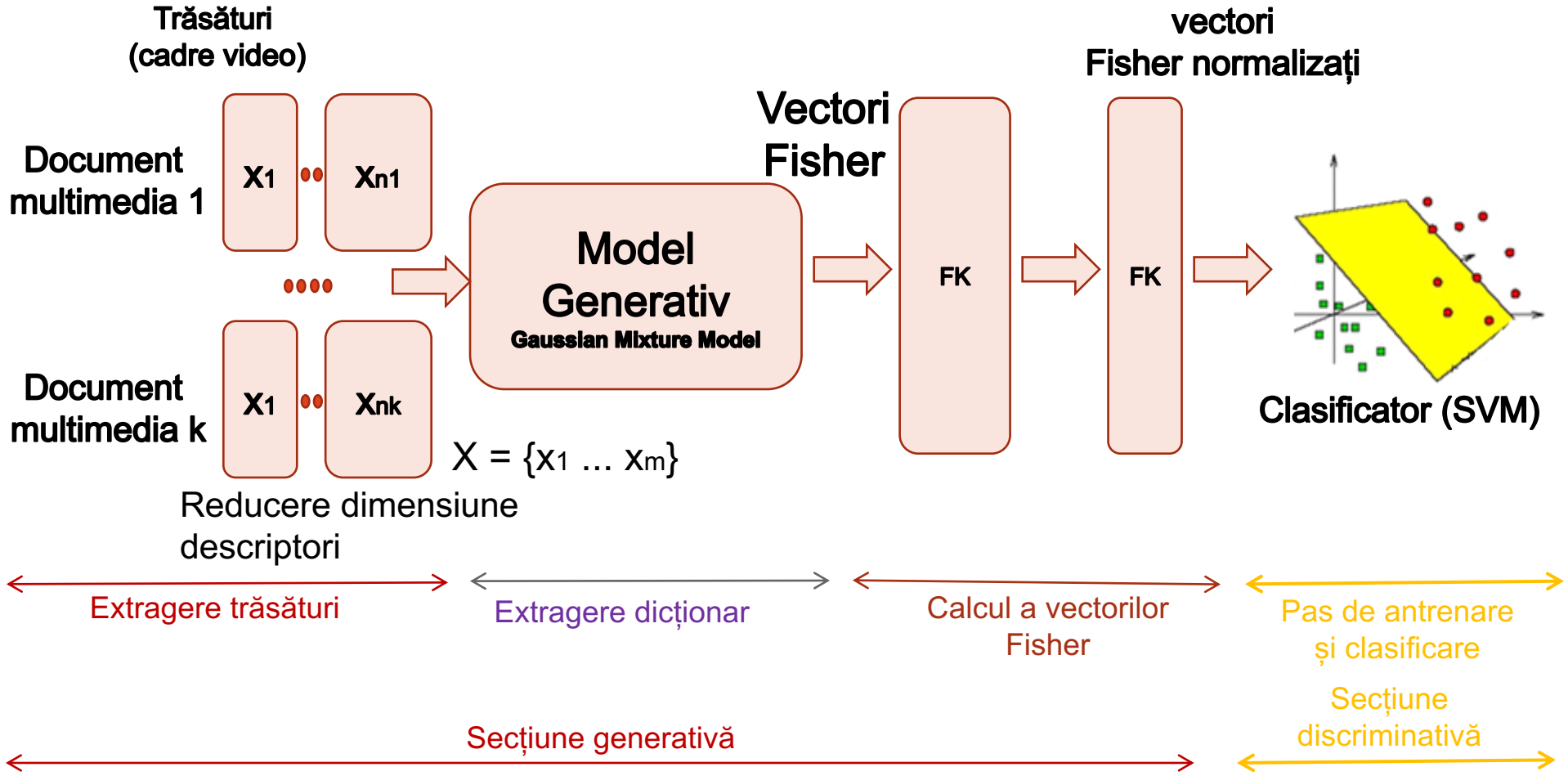
MODELUL FISHER KERNEL

Piramide spațiale



Modelul Fisher kernel

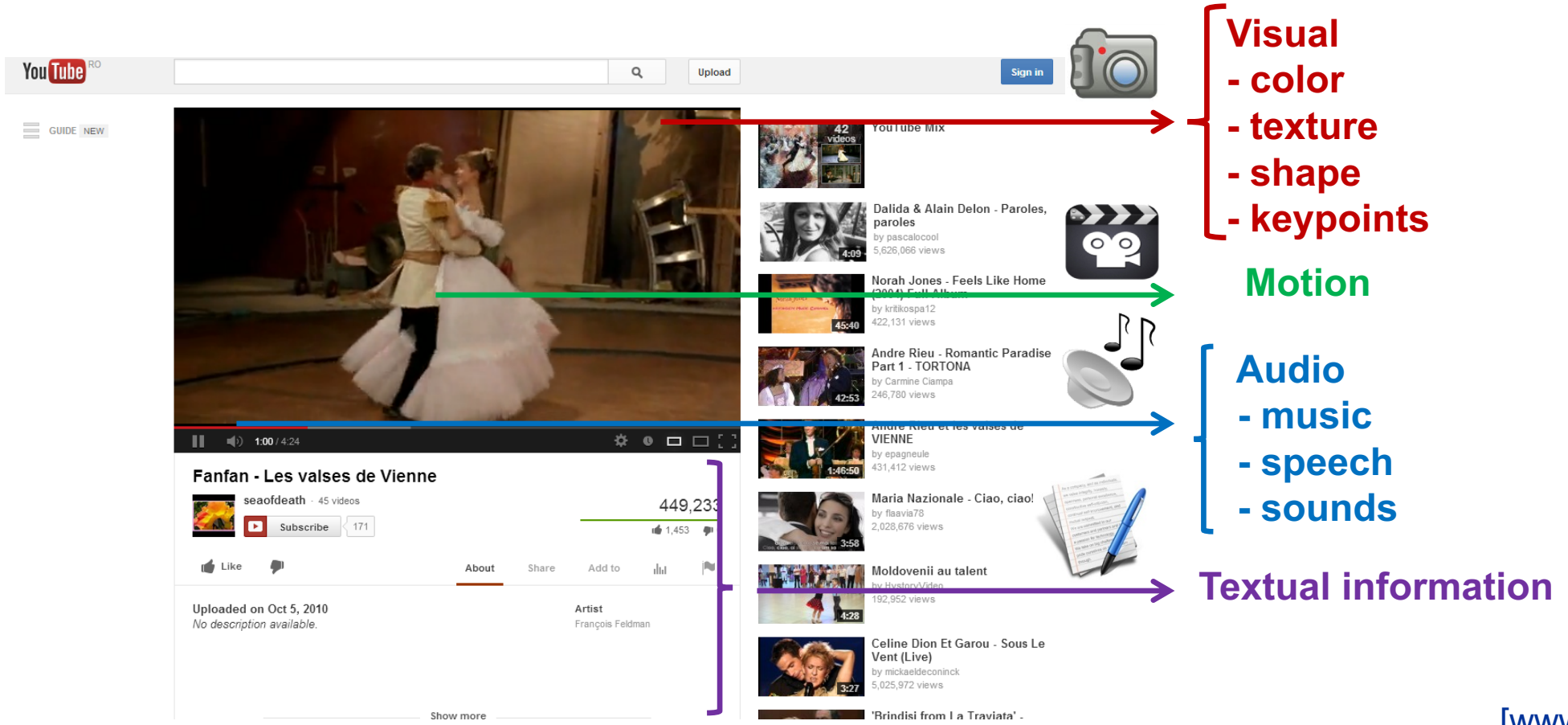
Arhitectura reprezentării „Fisher kernel”



[Mironică et al., ICMR'13 ACM]

Feature extraction

VIDEO CLASSIFICATION

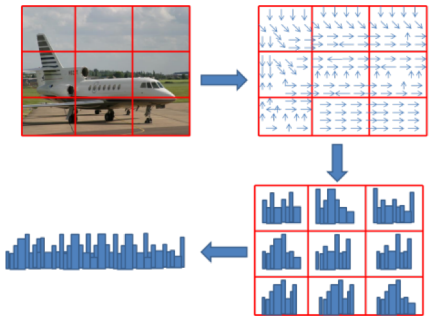


[www.youtube.com]

Feature extraction

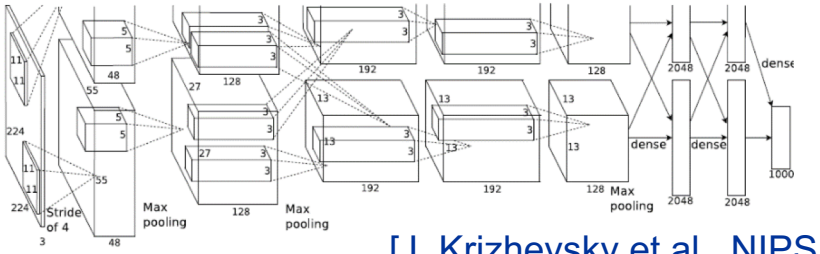
Visual features

Histograms of Oriented Gradients



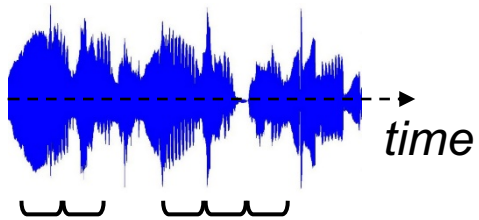
[CITS 2009, O. Ludwig,et all]

Convolutional Neural Networks



[J. Krizhevsky et al., NIPS, 2013]

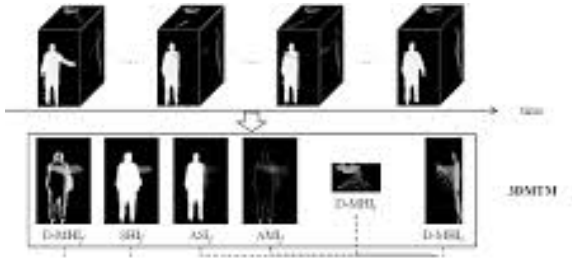
Audio features



[Tzanetakis al., Ismir, 2011]

Motion features

3D-HoG / 3D-HOF

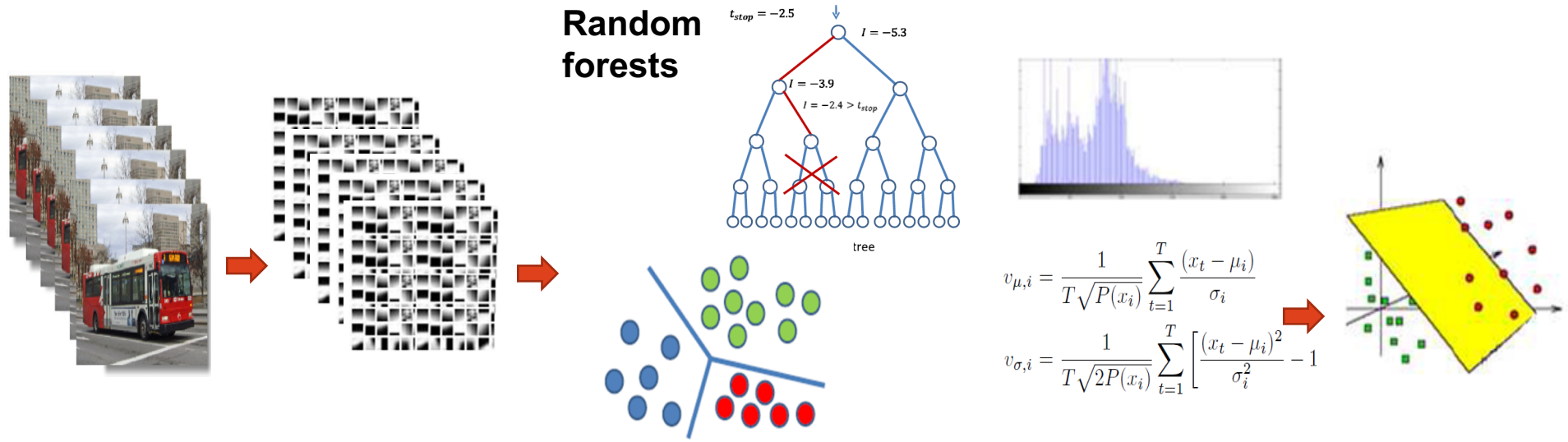


[J. Uijlings et al., IJMIR, 2014]

Feature extraction

VIDEO CLASSIFICATION

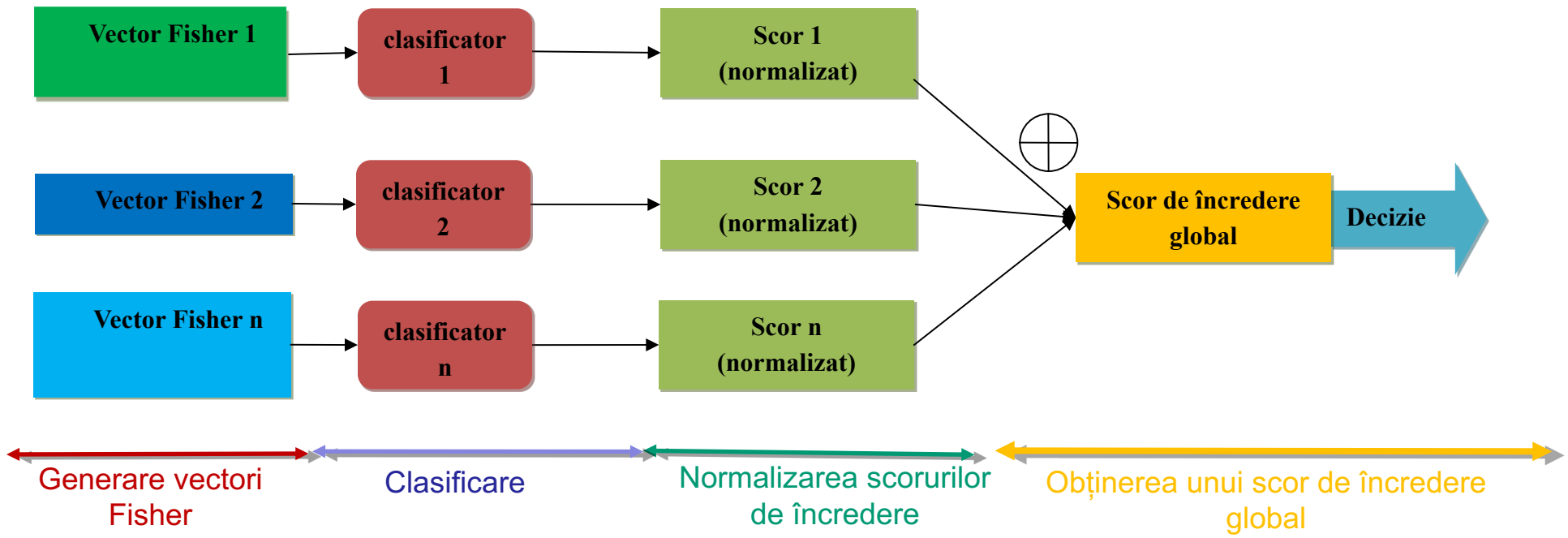
Fast automatic indexing of video databases;



Extraction of multimodal features → Generate a video dictionary → **Compute VLAD descriptors** → **Nonlinear SVM classifiers**

[Mironică et al., MTAP, 2016]

Fuziunea trăsăturilor – „Late Fusion”



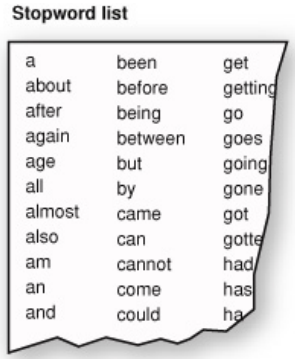
[Mironică et al., CBMI 2013, IEEE/ACM]

Feature extraction

TEXT CLASSIFICATION

Preprocessing steps

- Remove the stop words



- Word stemming

  “swimmers”, “swimming” -> swim

  “stems", "stemmer", "stemming", "stemmed" -> "stem".

- Extract the dictionary

[http://wiki.com]

Feature extraction

TEXT CLASSIFICATION

Bag of Words

(1) John likes to watch movies. Mary likes movies too.
(2) John also likes to watch football games.

Bag of Words

```
BoW1 = {"John":1,"likes":2,"to":1,"watch":1,"movies":2,"Mary":1,"too":1};
BoW2 = {"John":1,"also":1,"likes":1,"to":1,"watch":1,"football":1,"games":1};
```

(1) [1, 2, 1, 1, 2, 1, 1, 0, 0, 0]
(2) [1, 1, 1, 1, 0, 0, 0, 1, 1, 1]

[http://wiki.com]

Feature extraction

TEXT CLASSIFICATION

Term Frequency – Inverse Document Frequency

- A popular weighting schema that normalize the word frequency:

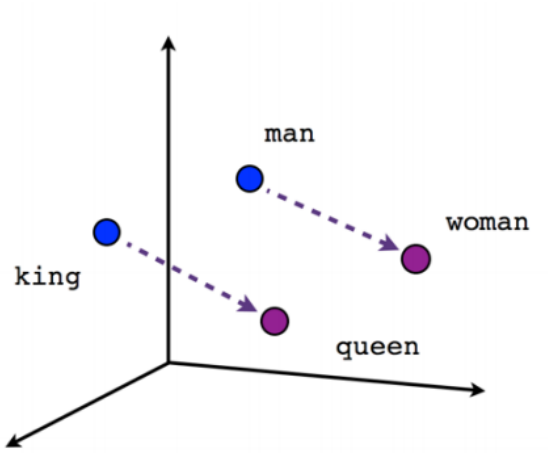
$$tfidf(w) = tf.\log(\frac{N}{df(w)})$$

- The word is more important if it appears several times in a target document
- The word is more important if it appears in less documents

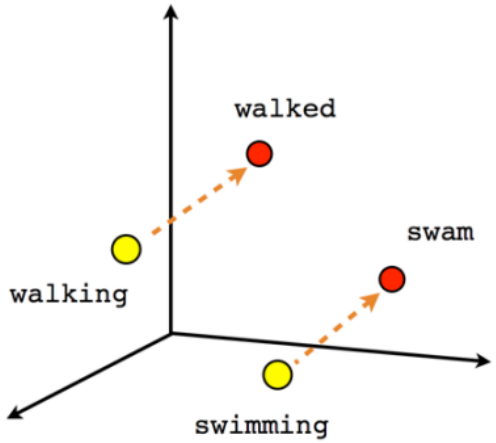
Feature extraction

TEXT CLASSIFICATION

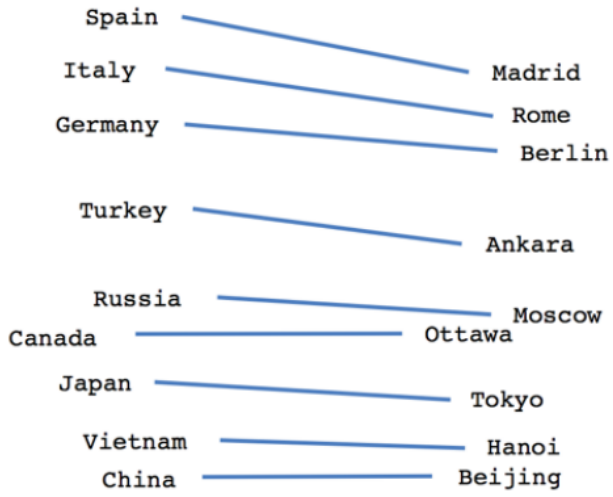
Word Embeddings



Male-Female



Verb tense



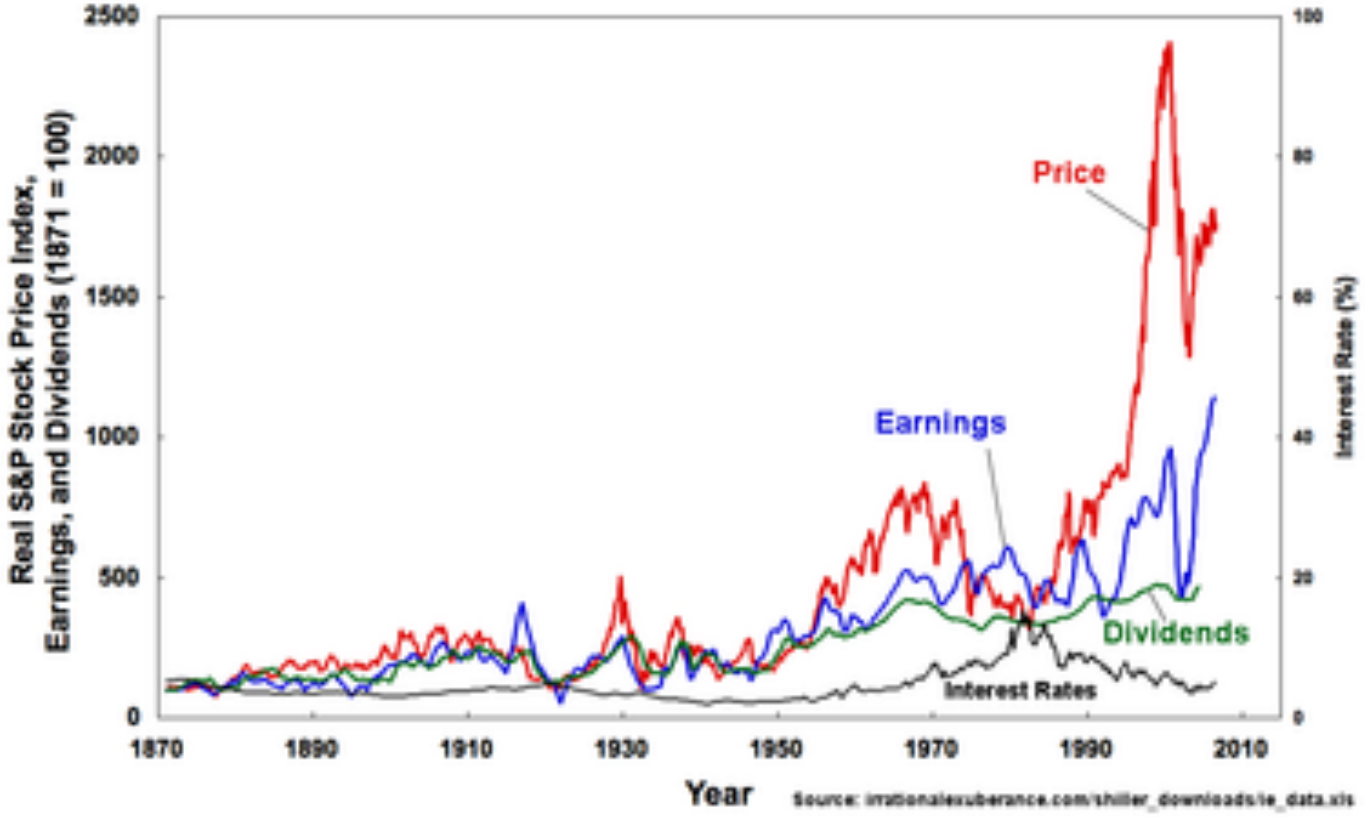
Country-Capital

[GloVe (Pennington et al., 2014)]

Feature extraction

STOCK MARKET PREDICTION

- Values of stocks (open, close, max, min)
- Bid/Ask volume
- Bid/Ask volume misbalance
- Price volatility
- Aggregated parameters on the last hours / days / weeks
- Correlations
- News & Tweets & Facebook (sentiment analysis)



[http://wiki.com]

Feature extraction

CREDIT DEFAULT PREDICTION

- Credit history
- Purpose of the credit
- Savings account/bonds
- Personal status and sex
- Other debtors / guarantors
- Present residence since
- Age in years
- Housing (rent / own / for free)
- Job
- Residency

[http://jantoo.com]

Feature engineering

CREDIT DATASET

Status of existing account	Credit history	Country	Age	Salary	Telephone
0 <= ... < 200 EUR	no credits taken	Germany	22	1000	none
>= 2000 EUR	all credits paid back duly	France	38	500	none
0 <= ... < 200 EUR	delay in paying off in the past		28		yes
200 EUR <= ... < 2000 EUR		Thailand	55	100000	yes
>= 2000 EUR	no credits taken		55	1000	

Feature engineering – missing data

Status of existing account	Credit history	Country	Age	Salary	Telephone
0 <= ... < 200 EUR	no credits taken	Germany	22	1000	none
>= 2000 EUR	all credits paid back duly	France	38	500	none
0 <= ... < 200 EUR	delay in paying off in the past	?	28	?	yes
200 EUR <= ... < 2000 EUR	?	Thailand	55	100000	yes
>= 2000 EUR	no credits taken	?	55	1000	?

Feature engineering – missing data

Alternative 1: remove these lines

Alternative 2: dummy methods
Use aggregation operators for numeric categories:
- Average, Max, Min, Median value

Categorical features:
- Most used value
- Other category

Alternative 3: A value estimated by another predictive model.

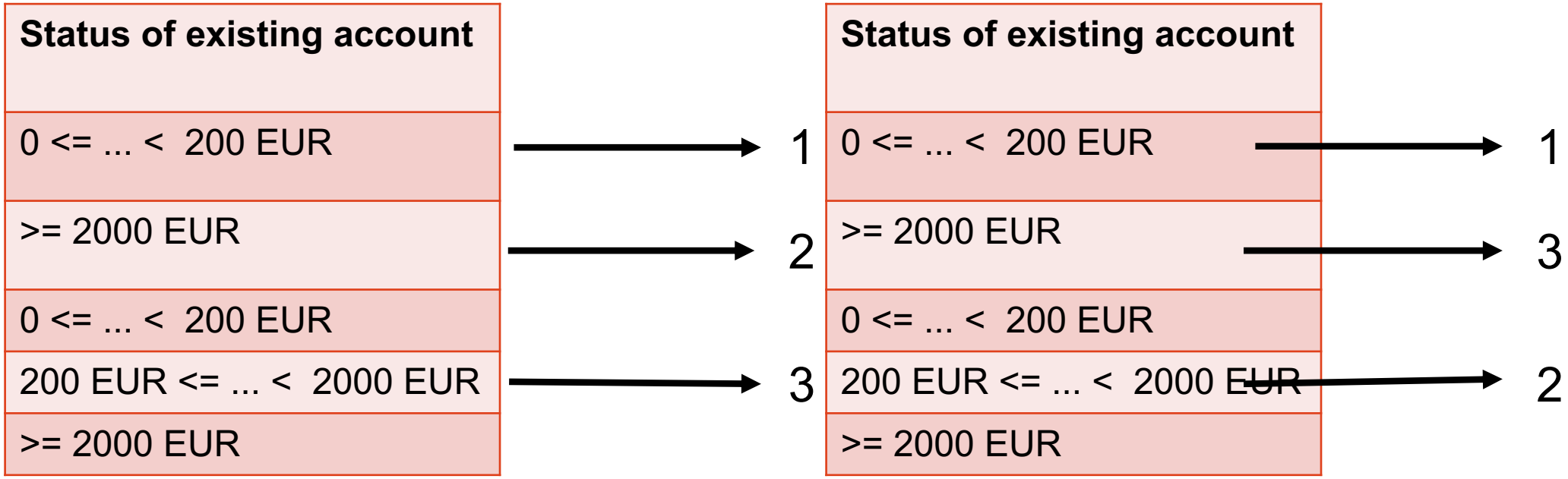
Feature engineering – categorical data

Telephone
none
none
yes
yes
none

Telephone
0
0
1
1
0

Feature engineering – categorical data

Label Encoder: It is used to transform non-numerical labels to numerical labels. Numerical labels are always between **0** and **n_classes-1**.



Feature engineering – categorical data

Dummy Encoder: Dummy coding is a commonly used method for converting a categorical input variable into continuous variable. ‘Dummy’, as the name suggests is a duplicate variable which represents one level of a categorical variable. Presence of a level is represented by 1 and absence is represented by 0.

Credit history
no credits taken
all credits paid back duly
delay in paying off in the past
all credits paid back duly
no credits taken

→

no credits taken	all credits paid back duly	delay in paying off in the past
1	0	0
0	1	0
0	0	1
0	1	0
1	0	0

Feature engineering – categorical data

Country
Germany
France
Australia
Thailand
US

Germany	France	250 columns	US
1	0		0
0	1		0
0	0		0
0	0		0
0	0		1

Feature engineering – categorical data

Country
Germany
France
Australia
Thailand
US

Region
Europe
Europe
Oceania
Asia
America

Europe	Oceania	Asia	America
1	0	0	0
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1

Feature engineering – different scales

Feature normalization

Age	Salary
22	1000
38	500
28	1000000
55	100000
55	1000

[18 - 80] [0 - 1000000]

→

Age	Salary
0.22	0.001
0.38	0.0005
0.28	1
0.55	0.1
0.55	0.001

Robust scaler

$$x' = \frac{x_i - Q_1(x)}{Q_3(x) - Q_1(x)}$$

Rescaling

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

Mean normalization

$$x' = \frac{x - \text{mean}(x)}{\max(x) - \min(x)}$$

Standardization

$$x' = \frac{x - \bar{x}}{\sigma}$$

Scaling to unit length

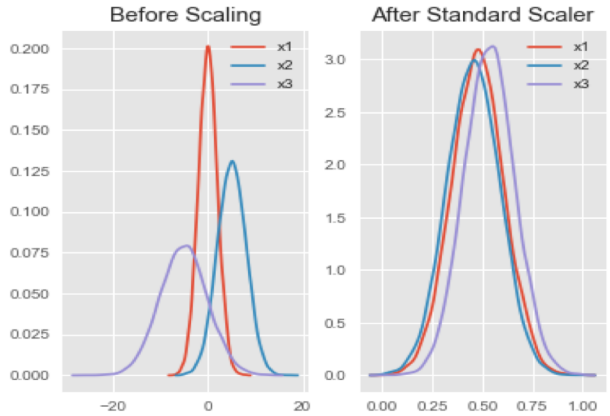
$$x' = \frac{x}{||x||}$$

Feature engineering – different scales

Feature normalization

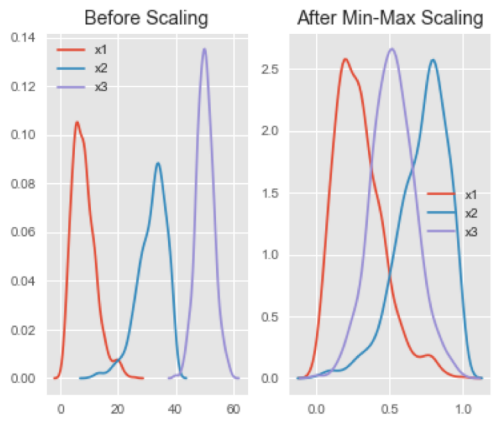
Standardization

$$x' = \frac{x - \bar{x}}{\sigma}$$



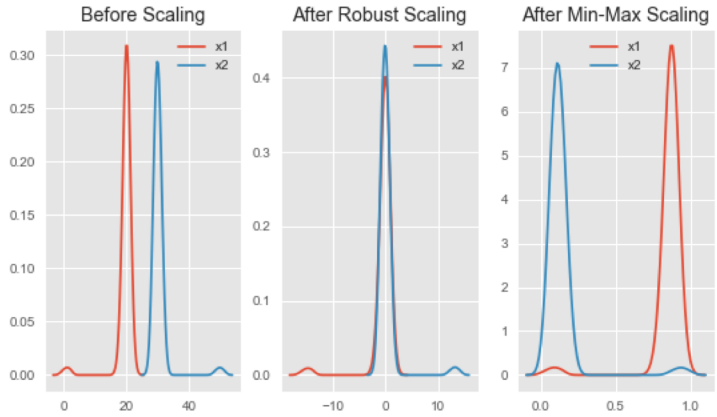
Rescaling

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

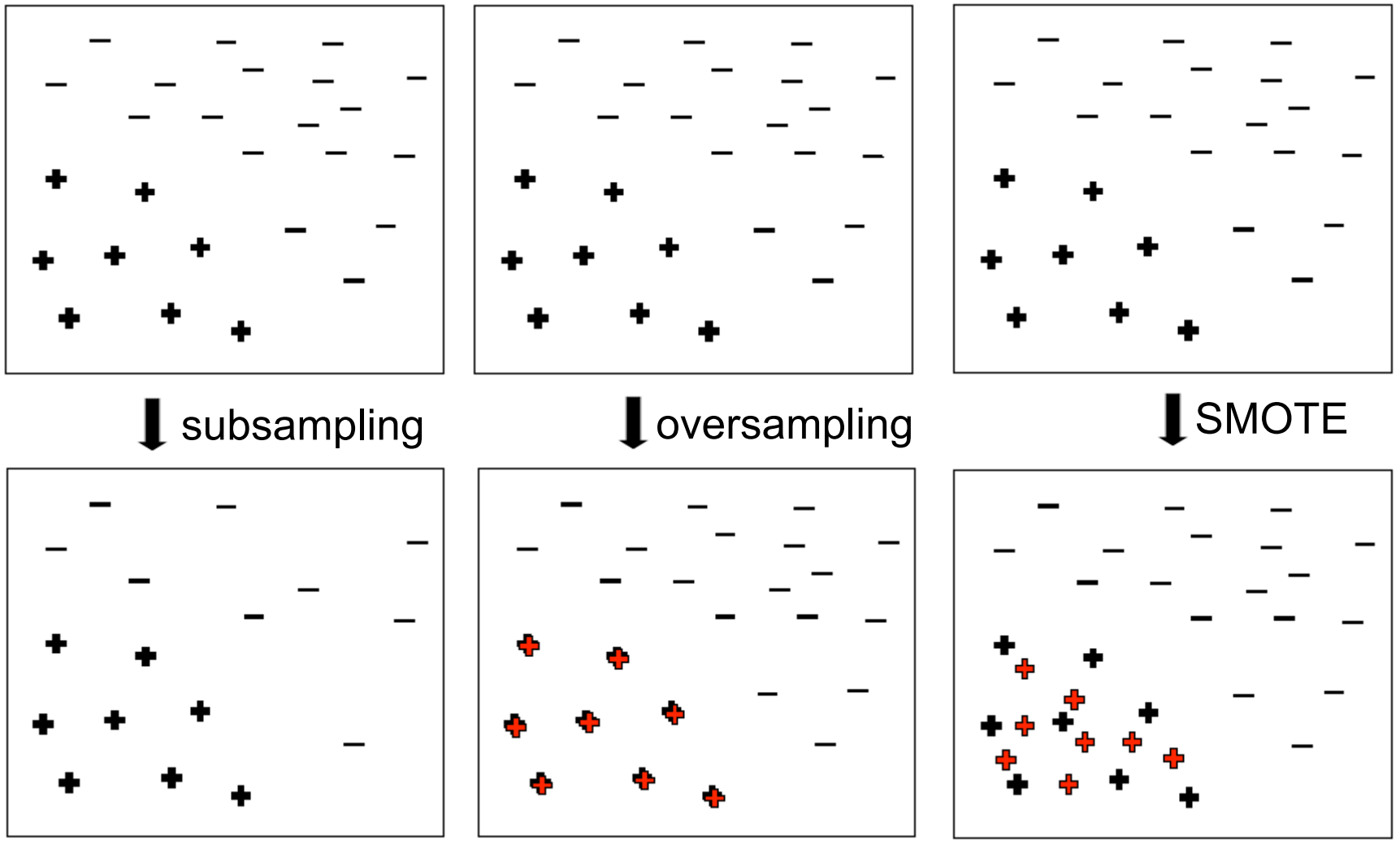


Robust scaler

$$x' = \frac{x_i - Q_1(x)}{Q_3(x) - Q_1(x)}$$

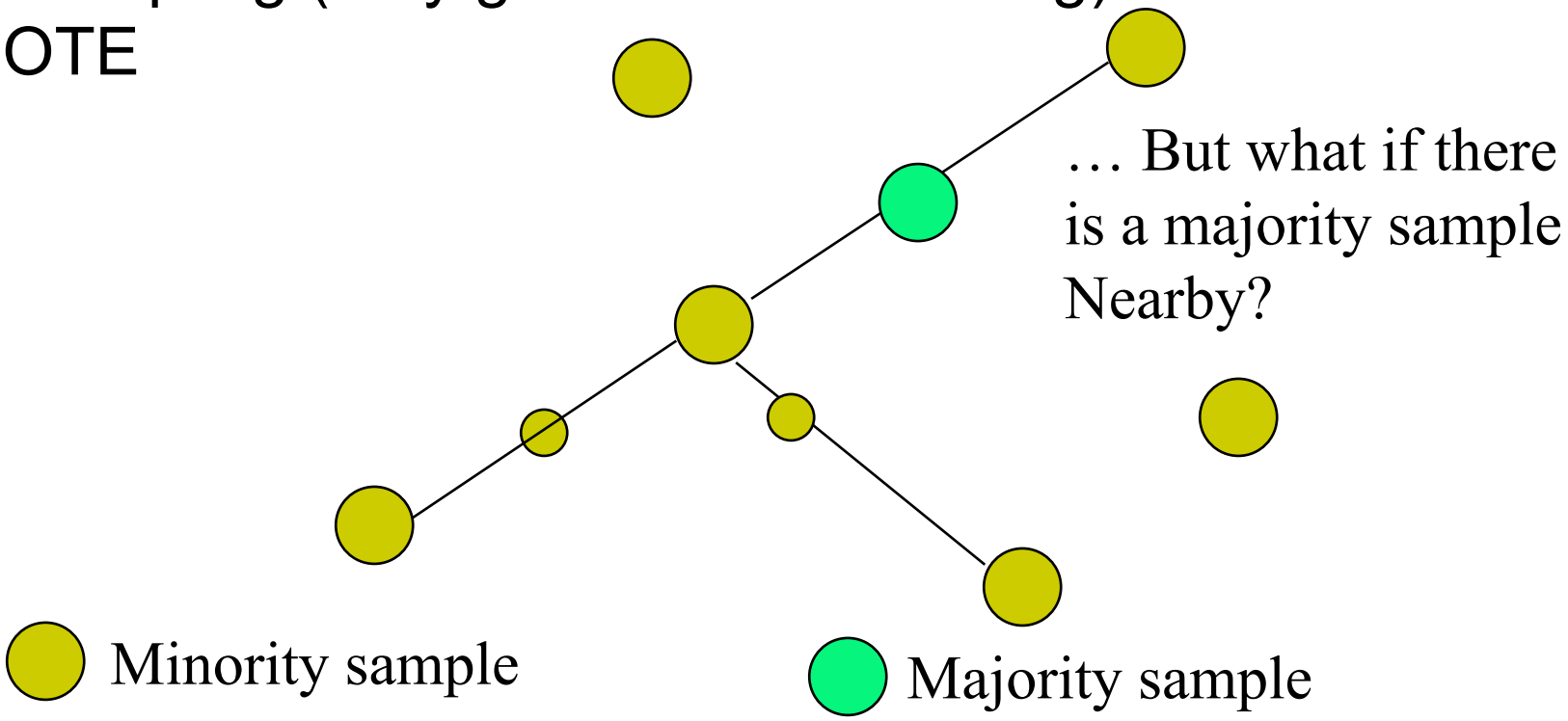


Feature engineering – imbalanced data

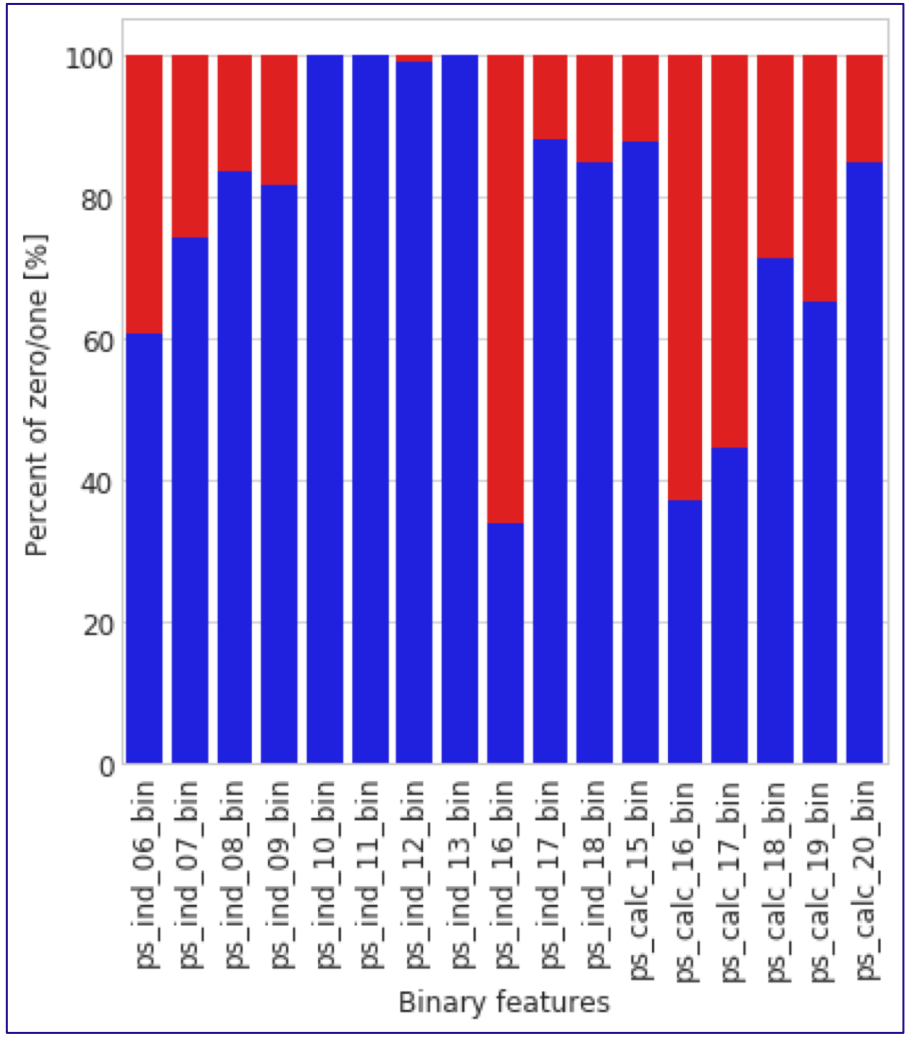


Feature engineering – imbalanced data

- Oversampling (may generate overfitting)
- Subsampling (may generate underfitting)
- SMOTE



Feature engineering – unnecessary features

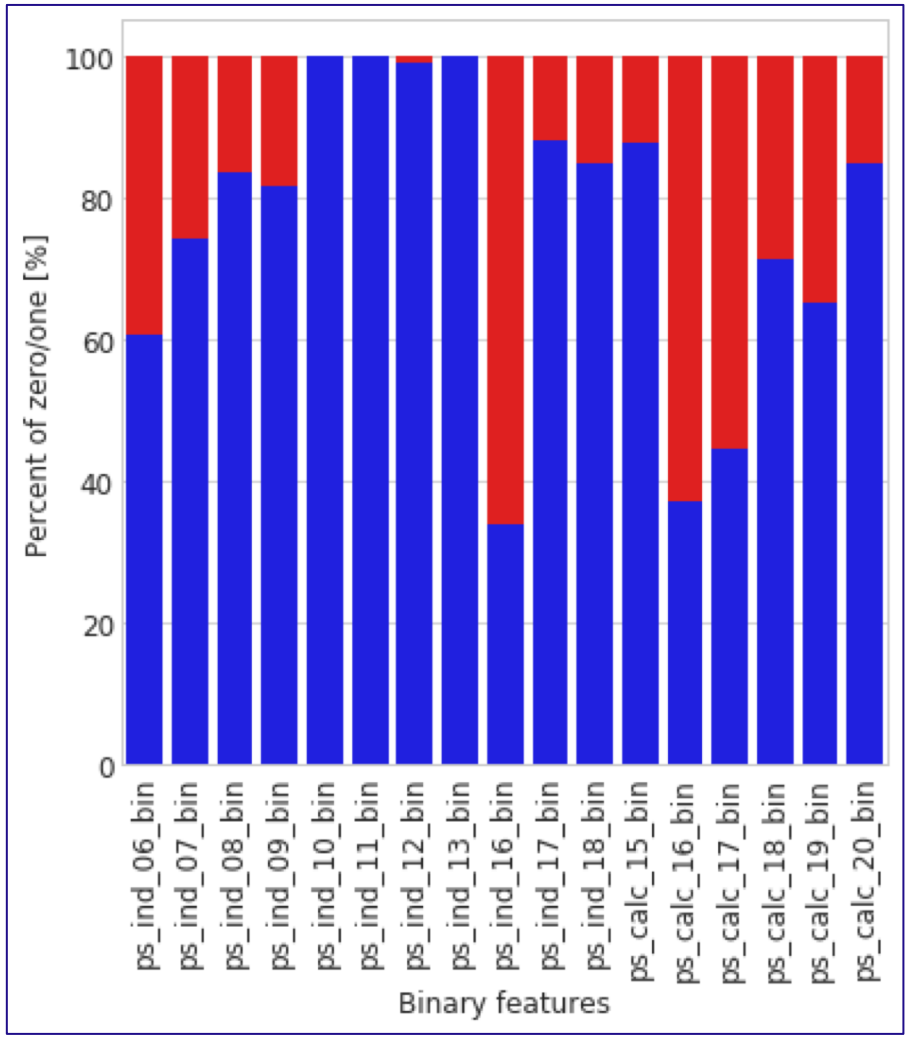


Binary features

Distribution of features values with
target = 1 and **target = 0**

[G. Preda, I. Mironică AAD Conference 2018]

Feature engineering – most important features



Binary features

Distribution of features values
with
target = 1 and **target = 0**

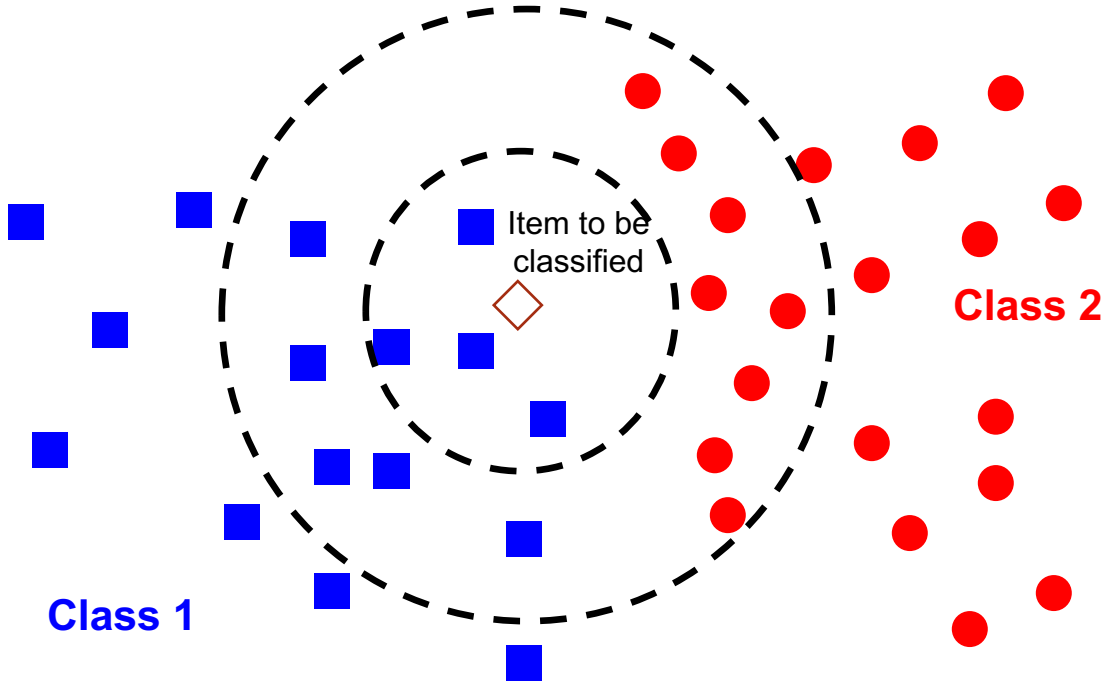
[G. Preda, I. Mironică AAD Conference 2018]

Session 2

CLASSICAL

MACHINE LEARNING ALGORITHMS

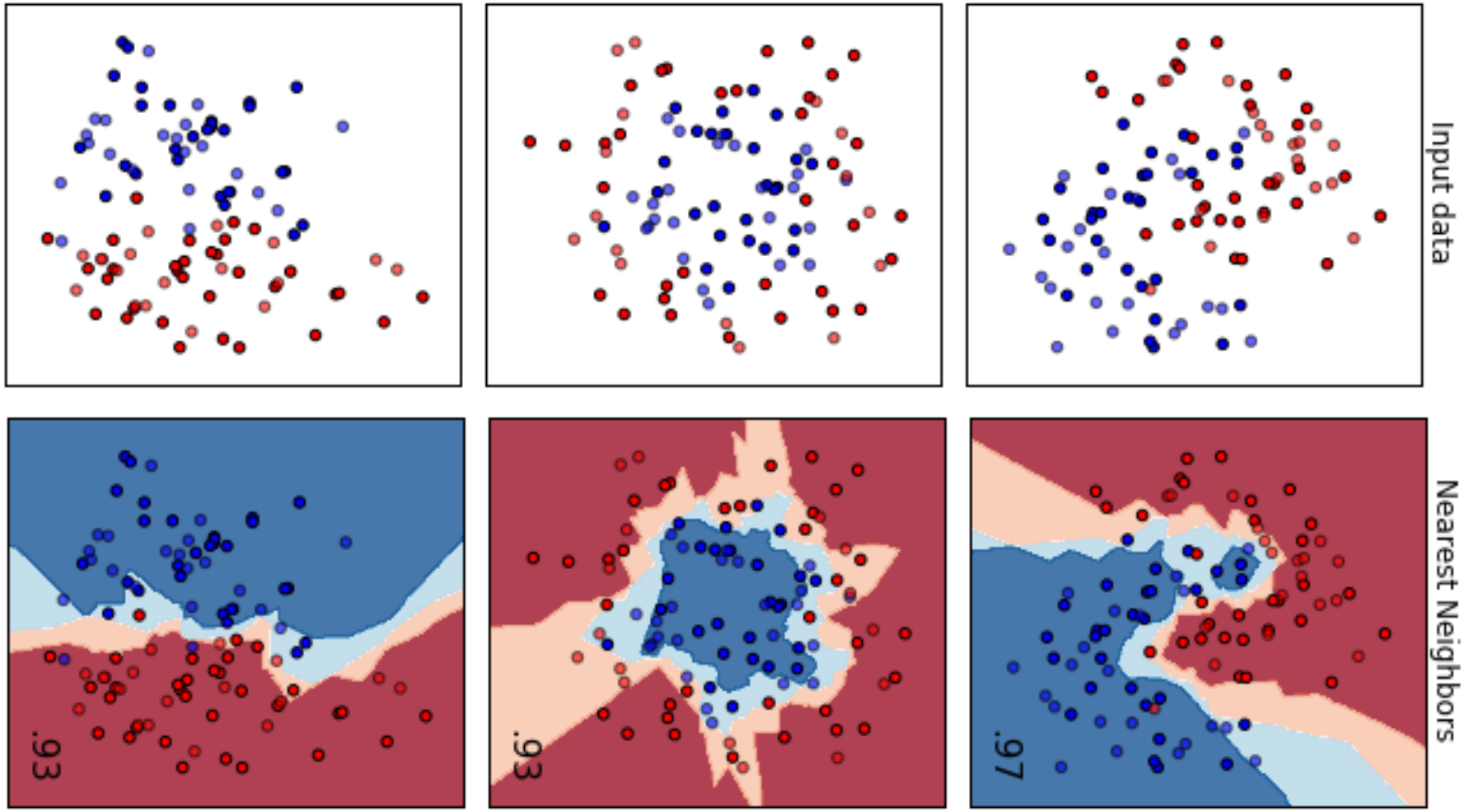
Nearest neighbor



K-NN computes the distance between the item that need to be classified to the most close objects from the database.

The item it will be classified according to the class of the neighbors.

Nearest neighbor



Nearest neighbor

Parameters to consider

- Number of neighbors

  If *k* is too small, sensitive to noise points
  If *k* is too large, neighborhood may include points from other classes

- The way of computing the metric

Nearest neighbor

ADVANTAGES

Small amount of data to train your model

Training is very fast (almost zero)

DISADVANTAGES

You need to keep in memory all the data

Slow: you need to compare your feature with all the stored database

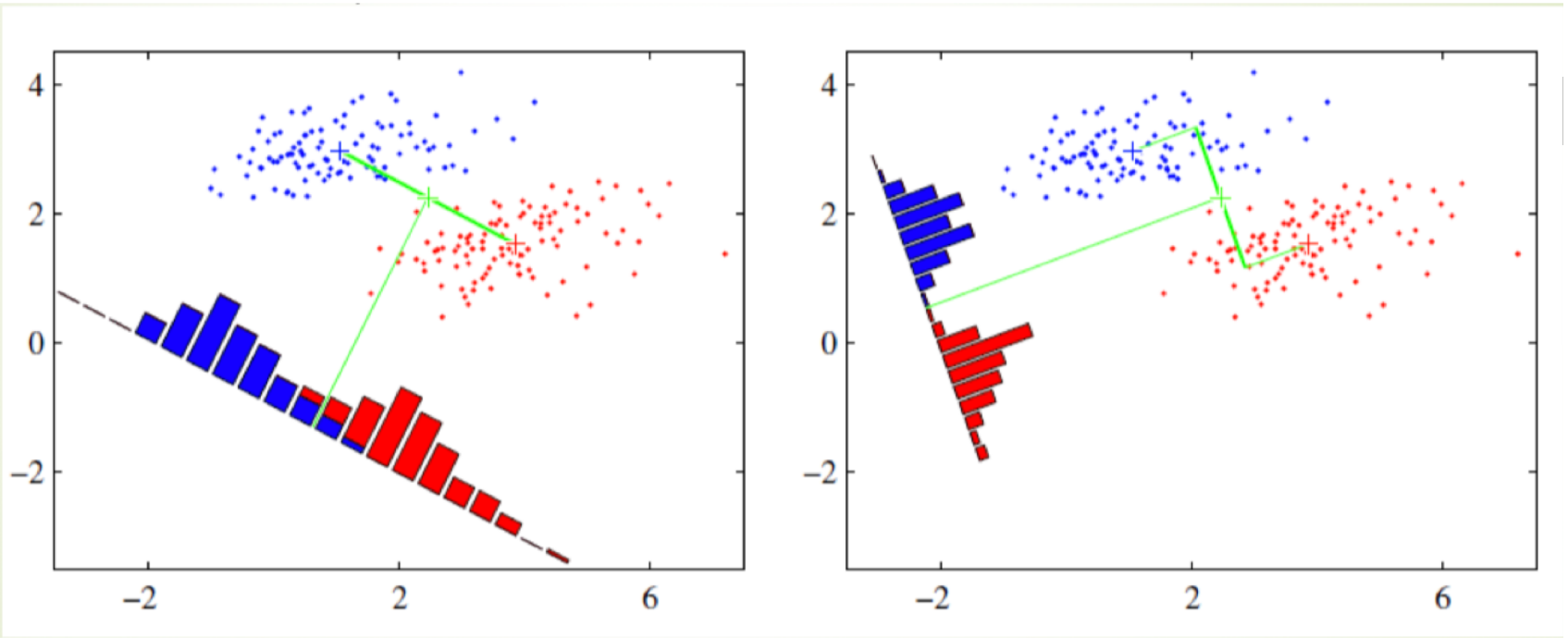
Highly depends on the metric used

You cannot use it for complex algorithms

Linear classification

FISHER METHOD

LDA assumes that the conditional probability density functions p(x|y=0) and p(x|y=1) are both normally distributed with mean and covariance.



$$\vec{w} \cdot \vec{x} > c$$

$$\vec{w} = \Sigma^{-1}(\vec{\mu}_1 - \vec{\mu}_0)$$

$$c = \frac{1}{2}(T - \vec{\mu}_0{}^T \Sigma_0^{-1} \vec{\mu}_0 + \vec{\mu}_1{}^T \Sigma_1^{-1} \vec{\mu}_1)$$

Linear classification

LINEAR PERCEPTRON METHOD

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 \qquad h(x) = \sum_{i=0}^{n} \theta_i x_i = \theta^T x,$$

Use Gradient Descent to update the weights:

- Choose some initial weights
- Define cost function

$$J(\theta) = \frac{1}{2} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2.$$

- Update weights until convergence

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta).$$

$$\begin{aligned}
\frac{\partial}{\partial \theta_j} J(\theta) &= \frac{\partial}{\partial \theta_j} \frac{1}{2} (h_\theta(x) - y)^2 \\
&= 2 \cdot \frac{1}{2} (h_\theta(x) - y) \cdot \frac{\partial}{\partial \theta_j} (h_\theta(x) - y) \\
&= (h_\theta(x) - y) \cdot \frac{\partial}{\partial \theta_j} \left(\sum_{i=0}^{n} \theta_i x_i - y \right) \\
&= (h_\theta(x) - y)\, x_j
\end{aligned}$$

Logistic regression

Logistic regression's output represents the probability of an independent variable to belong to a certain category

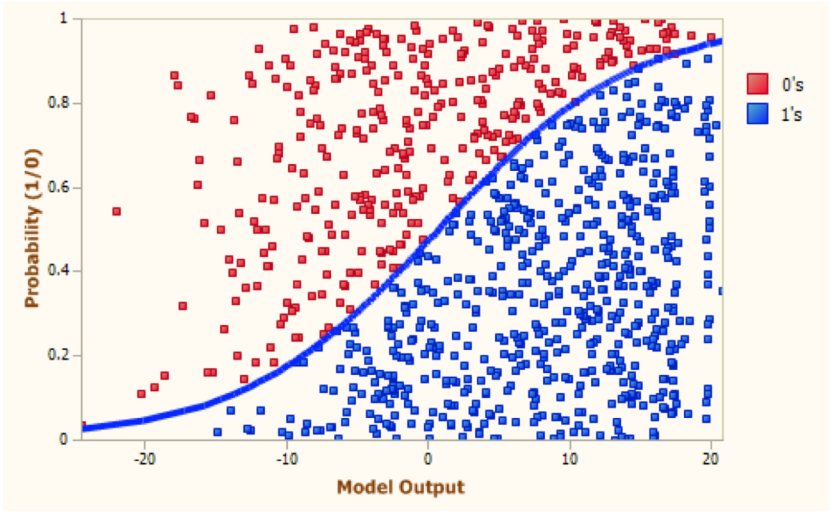
$$h_\theta(x) = g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}},$$

$$g(z) = \frac{1}{1 + e^{-z}}$$

$$\begin{aligned} g'(z) &= \frac{d}{dz}\frac{1}{1+e^{-z}} \\ &= \frac{1}{(1+e^{-z})^2}(e^{-z}) \\ &= \frac{1}{(1+e^{-z})} \cdot \left(1 - \frac{1}{(1+e^{-z})}\right) \\ &= g(z)(1-g(z)). \end{aligned}$$

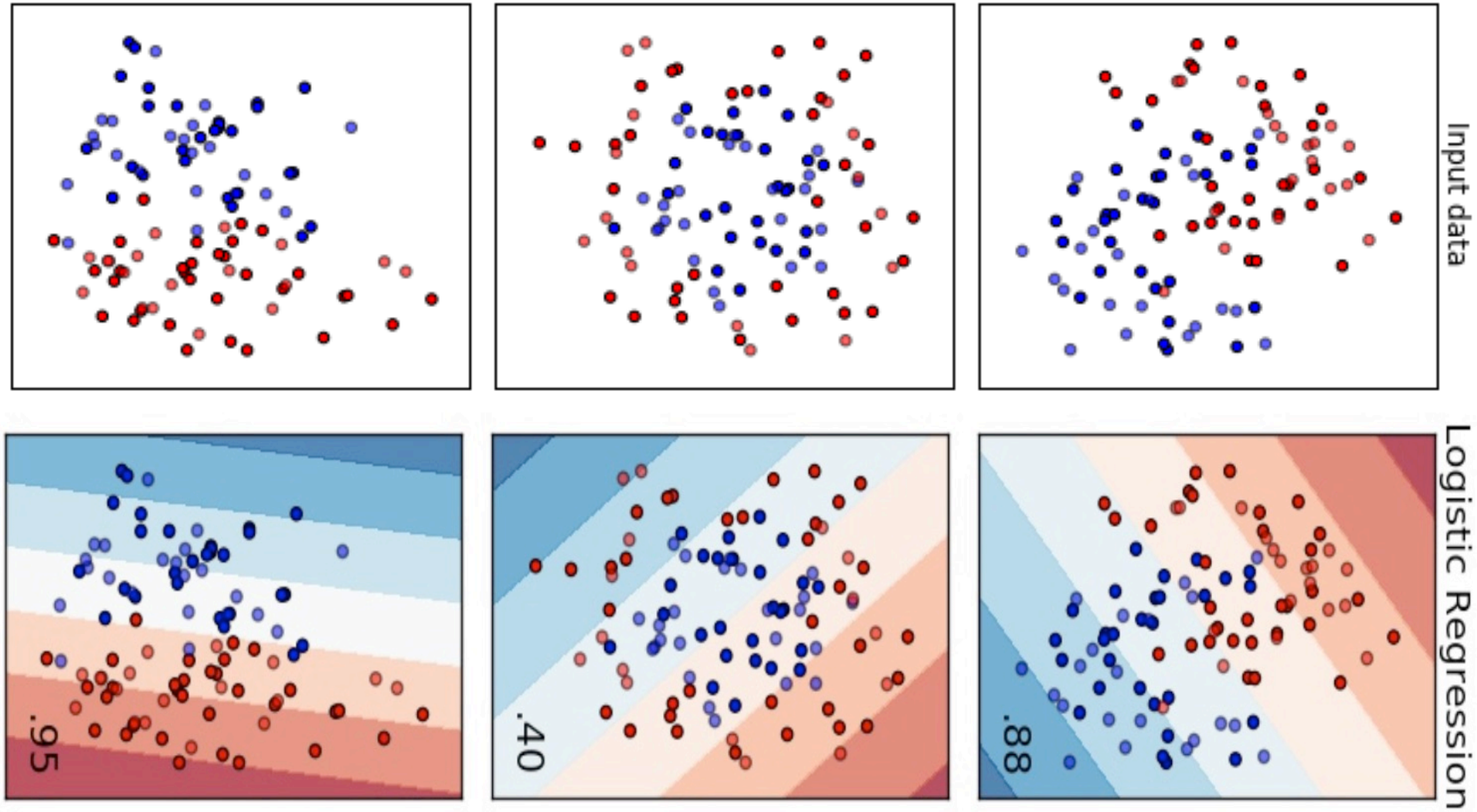
$$J(\theta) = -\left[\frac{1}{m}\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1-y^{(i)}) \log(1 - h_\theta(x^{(i)}))\right]$$

Use Gradient Descent to update the weights.

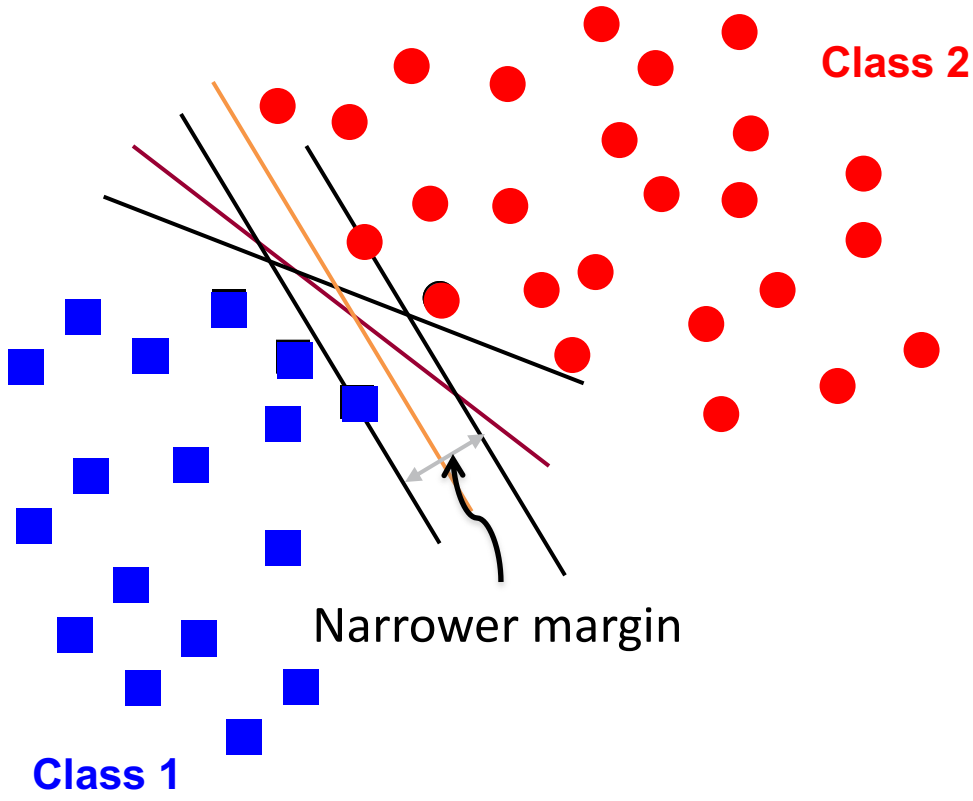


Logistics functions

Logistic regression

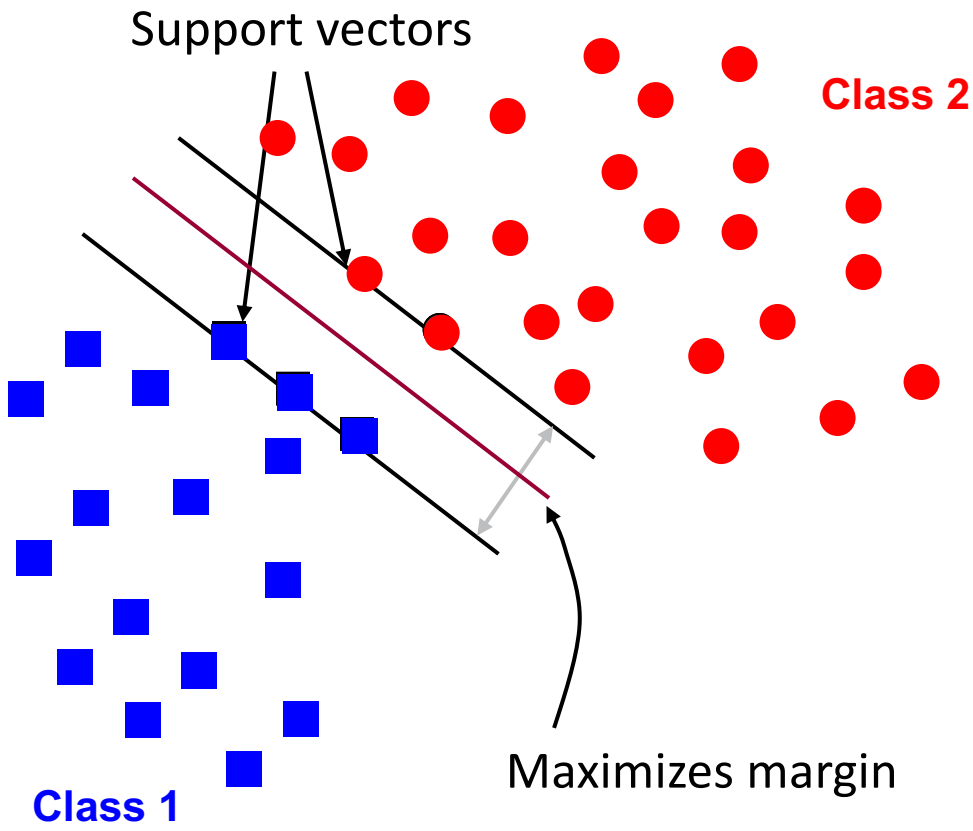


Support vector machines



SVM constructs a hyperplane that is used to separate different classes.

Support vector machines

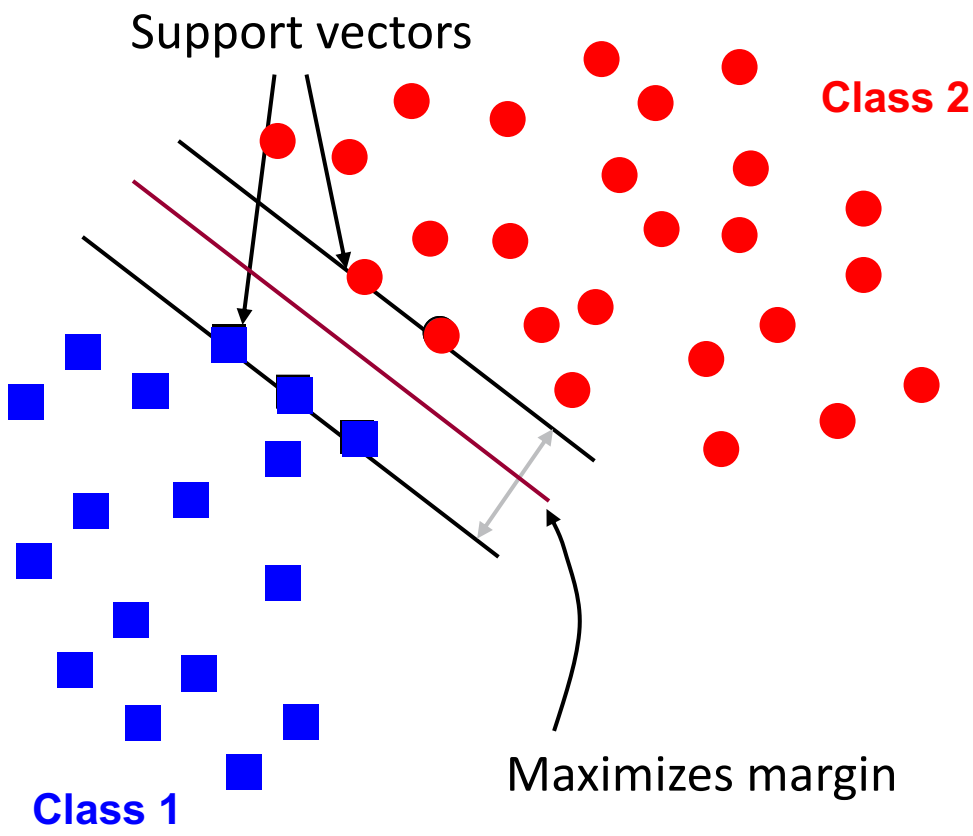


- The hyperplane is chosen where the distance between distinct categories is maximized.
- **Hard-margin** vs **Soft-margin**
- Function that needs to be minimized:

$$\left[\frac{1}{n}\sum_{i=1}^{n}\max\left(0, 1 - y_i(\vec{w}\cdot\vec{x}_i - b)\right)\right] + \lambda\|\vec{w}\|^2$$

Hinge loss function — Regularization

Support vector machines



$$\left[\frac{1}{n}\sum_{i=1}^{n}\max\left(0, 1 - y_i(\vec{w}\cdot\vec{x}_i - b)\right)\right] + \lambda\|\vec{w}\|^2$$

Hinge loss function — Regularization

Partial derivates:

$$\frac{\delta}{\delta w_k}\lambda \parallel w \parallel^2 = 2\lambda w_k$$

$$\frac{\delta}{\delta w_k}\left(1 - y_i\langle x_i, w\rangle\right)_+ = \begin{cases} 0, & \text{if } y_i\langle x_i, w\rangle \geq 1 \\ -y_i x_{ik}, & \text{else} \end{cases}$$

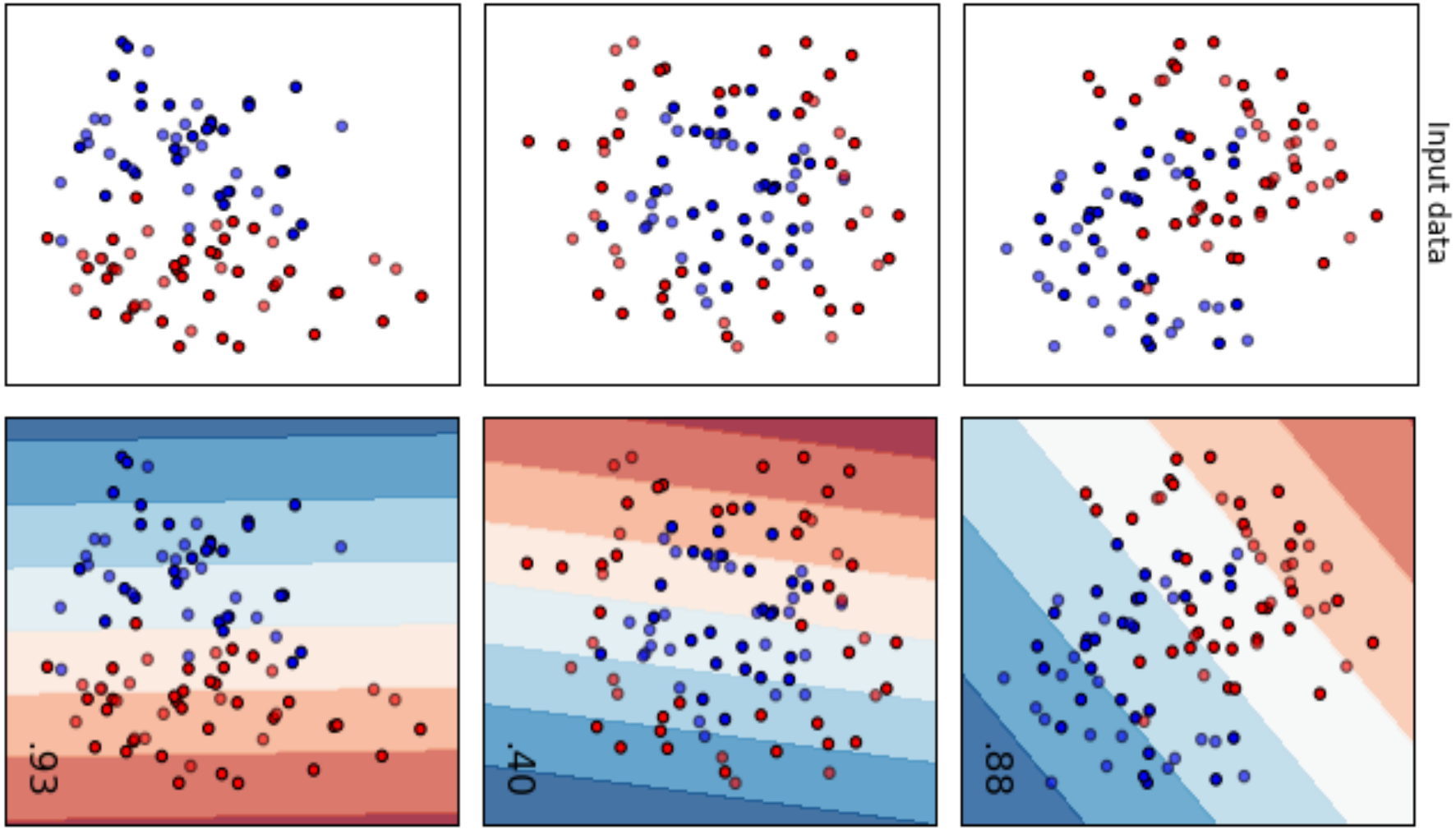
Gradient Update — No misclassification

$w = w - \alpha \cdot (2\lambda w)$

Gradient Update — Misclassification

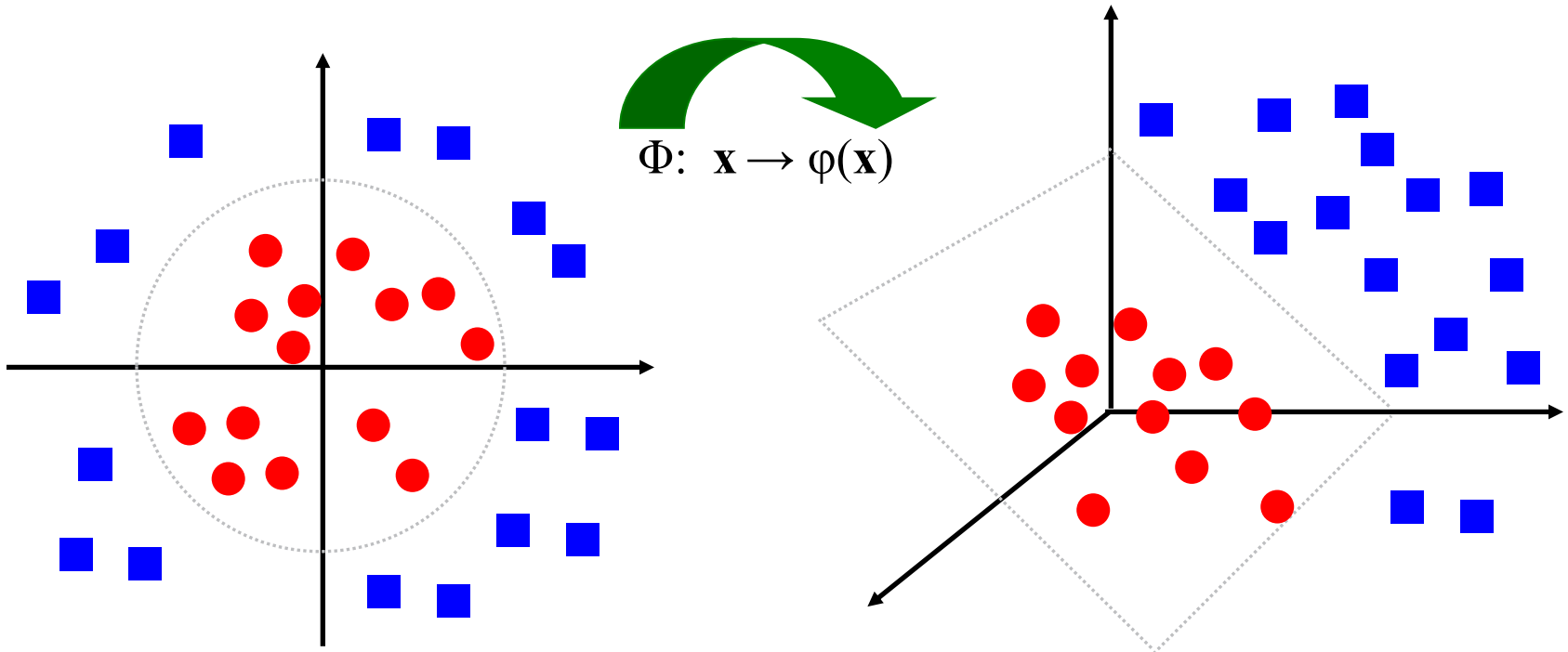
$w = w + \alpha \cdot (y_i \cdot x_i - 2\lambda w)$

Support vector machines



Nonlinear Support vector machines

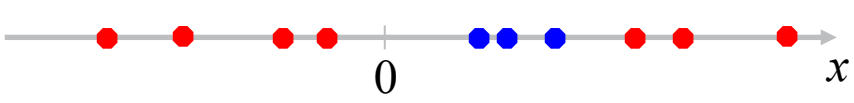
General idea: the original feature space can always be mapped to some higher-dimensional feature space where the training set is separable:



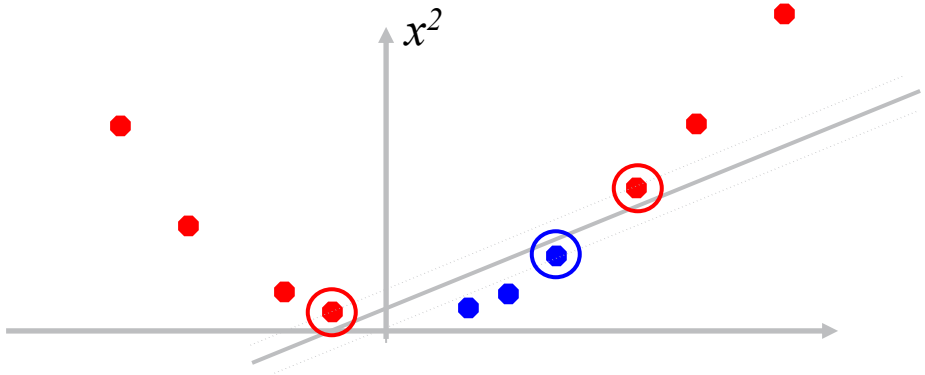
(x,y) -> (x, y, x*y)

Nonlinear Support vector machines

Non linear problem



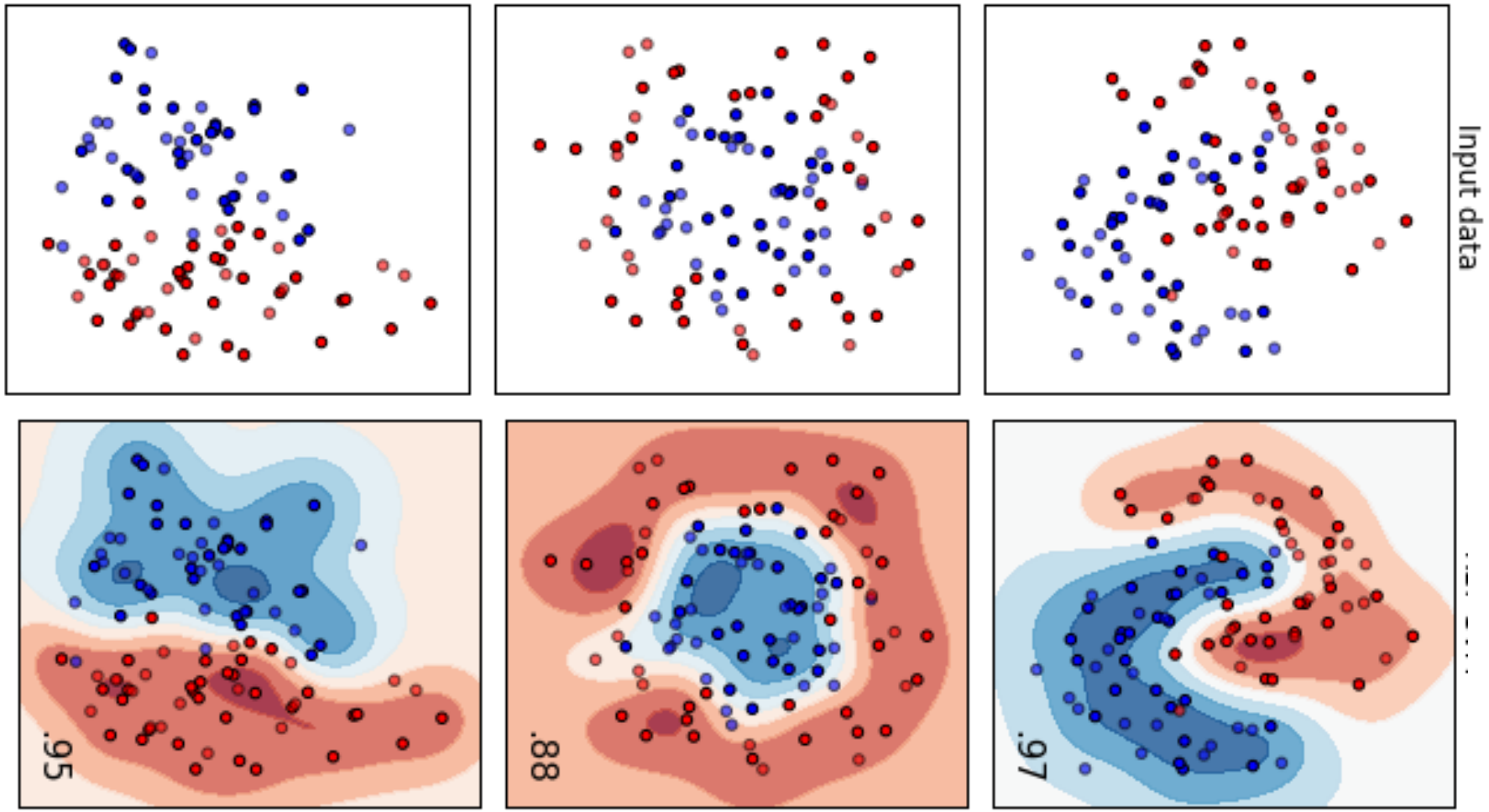
Map a multidimentional function $\varphi(x) = (x, x^2)$



$$\varphi(x) \cdot \varphi(y) = (x, x^2) \cdot (y, y^2) = xy + x^2 y^2$$

$$K(x, y) = xy + x^2 y^2$$

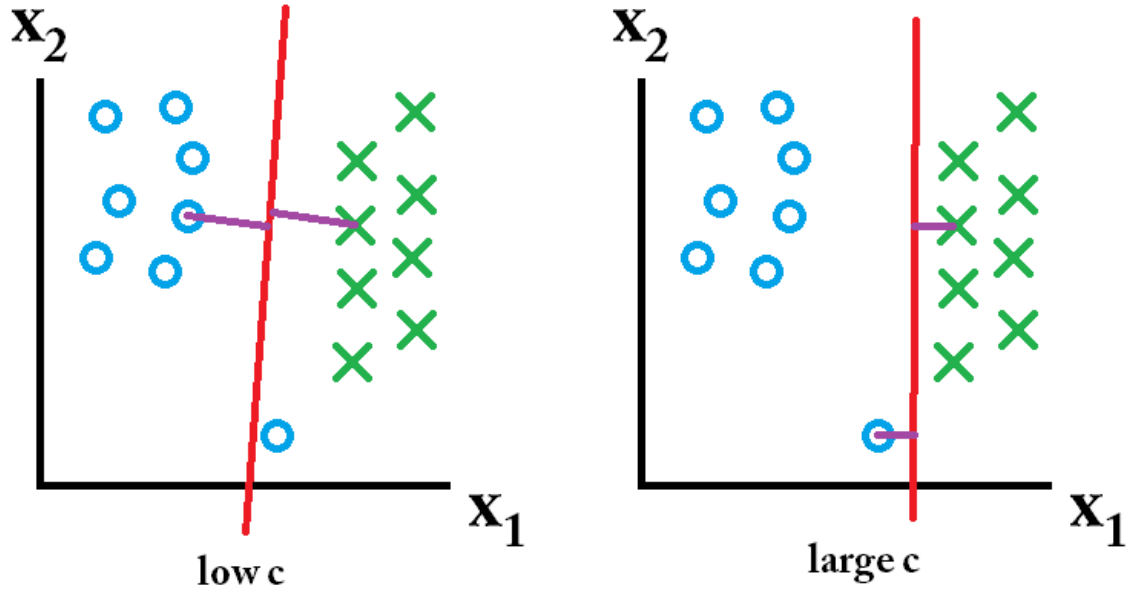
RBF Support vector machines



RBF Support vector machines

PARAMETERS TO CONSIDER

C - The way of computing the hyperplane:
- if it is two tight (C has a higher value) – the model will overfit
- if it is too general – the model will underfit



Gamma and the kernel type

Support vector machines

ADVANTAGES

Small amount of data to train your model

Training may be very fast on liner model

Can be used for complex algorithms

DISADVANTAGES

Training for nonlinear approaches can be slow

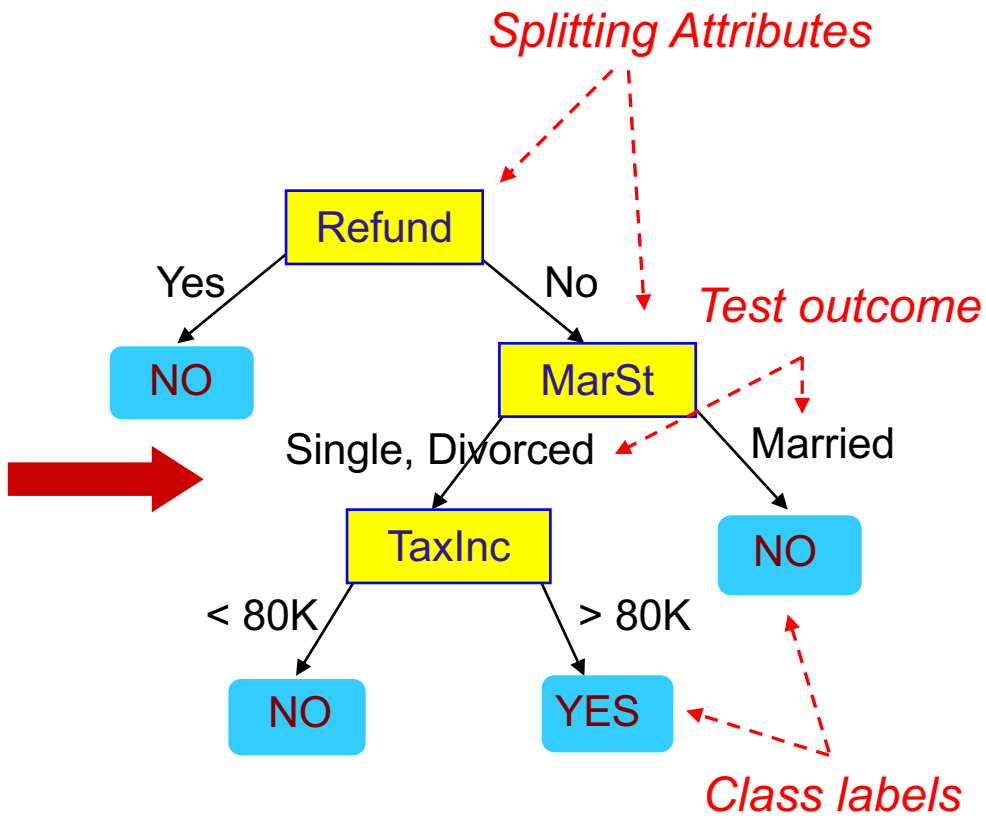
The data should be normalized

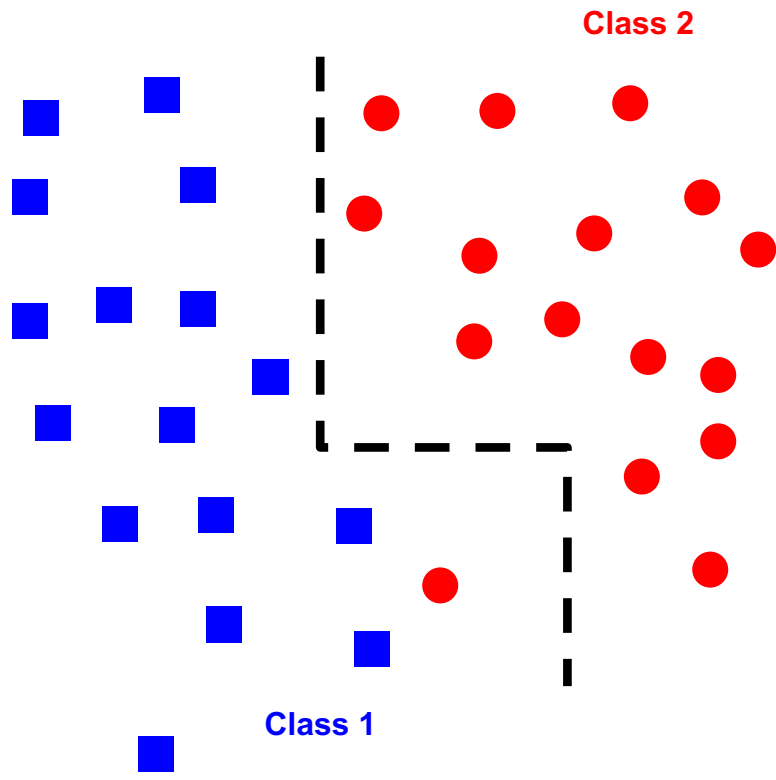
Decision trees

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

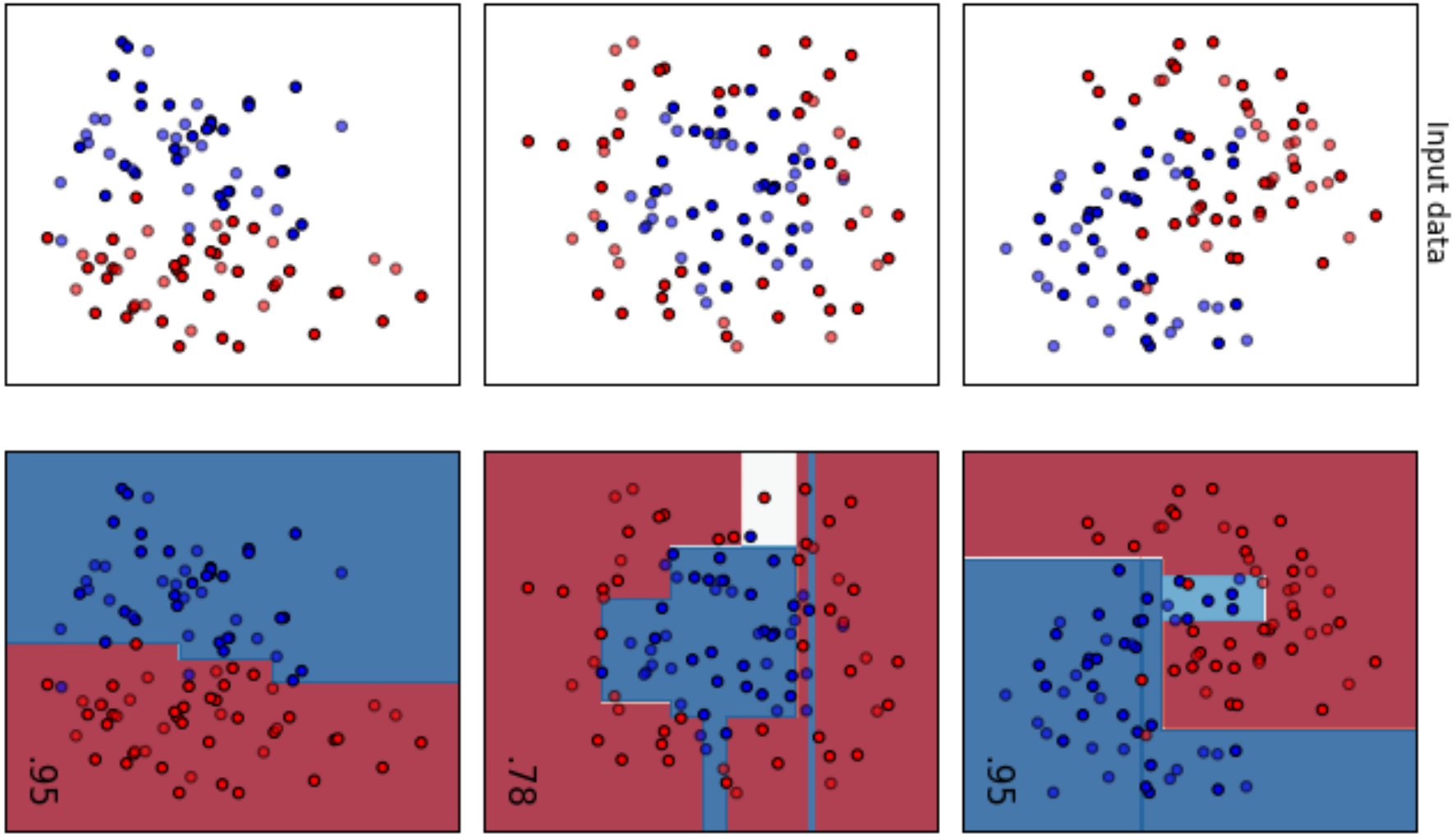
categorical categorical continuous class

Training Data





Decision trees



Decision trees

PARAMETERS TO CONSIDER

Criterion - the function to measure the quality of a split (Gini vs Entropy)

Max depth of the tree:
- high values – the model will overfit
- small values – the model will underfit

Decision trees

ADVANTAGES

Small amount of data to train your model

Training is very fast

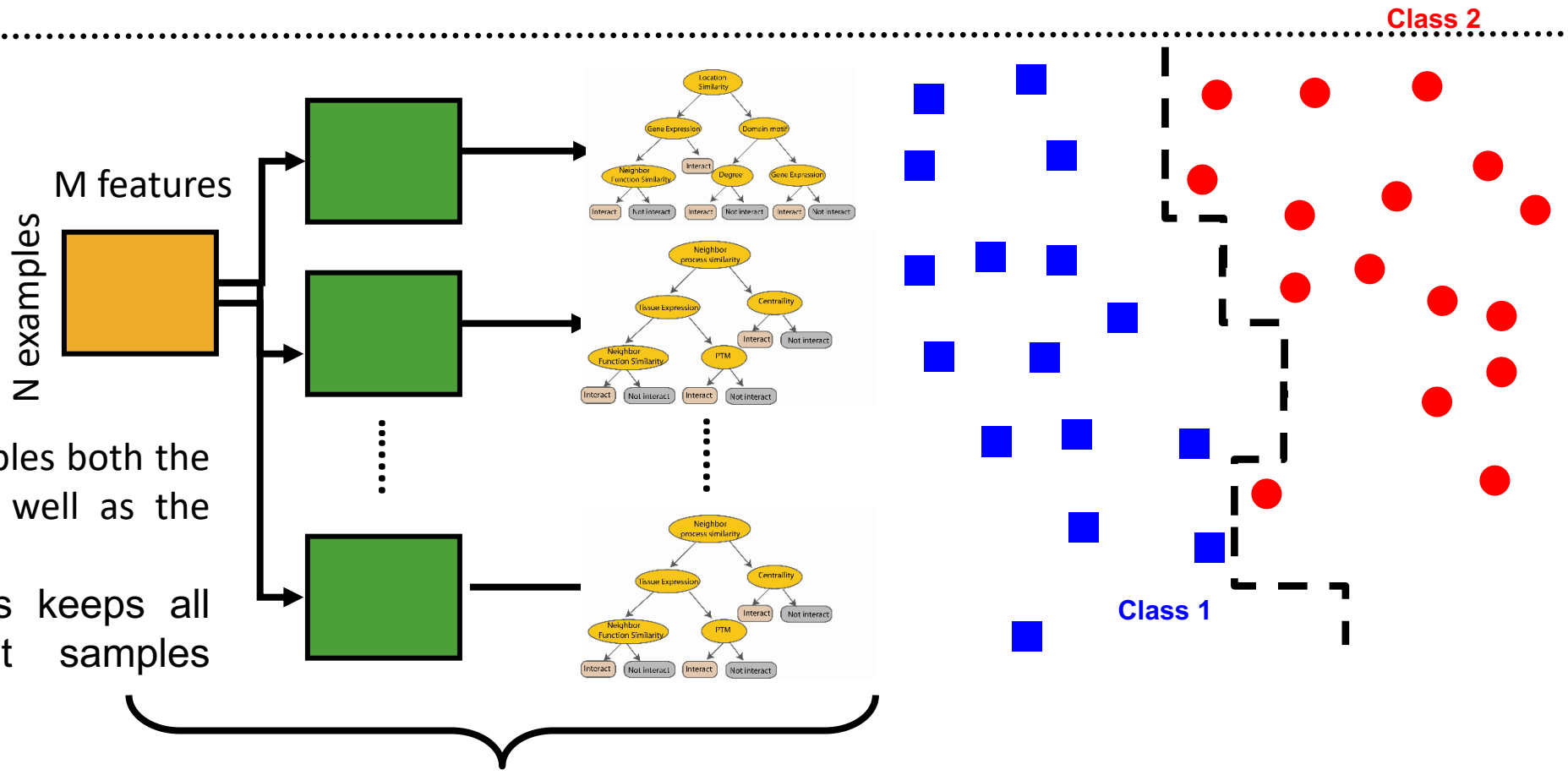
Provide a dendogram that is able to show the importance of each feature

The data do not need to be normalized

DISADVANTAGES

It is very easy to overfit

Bagging algorithms - Random forests

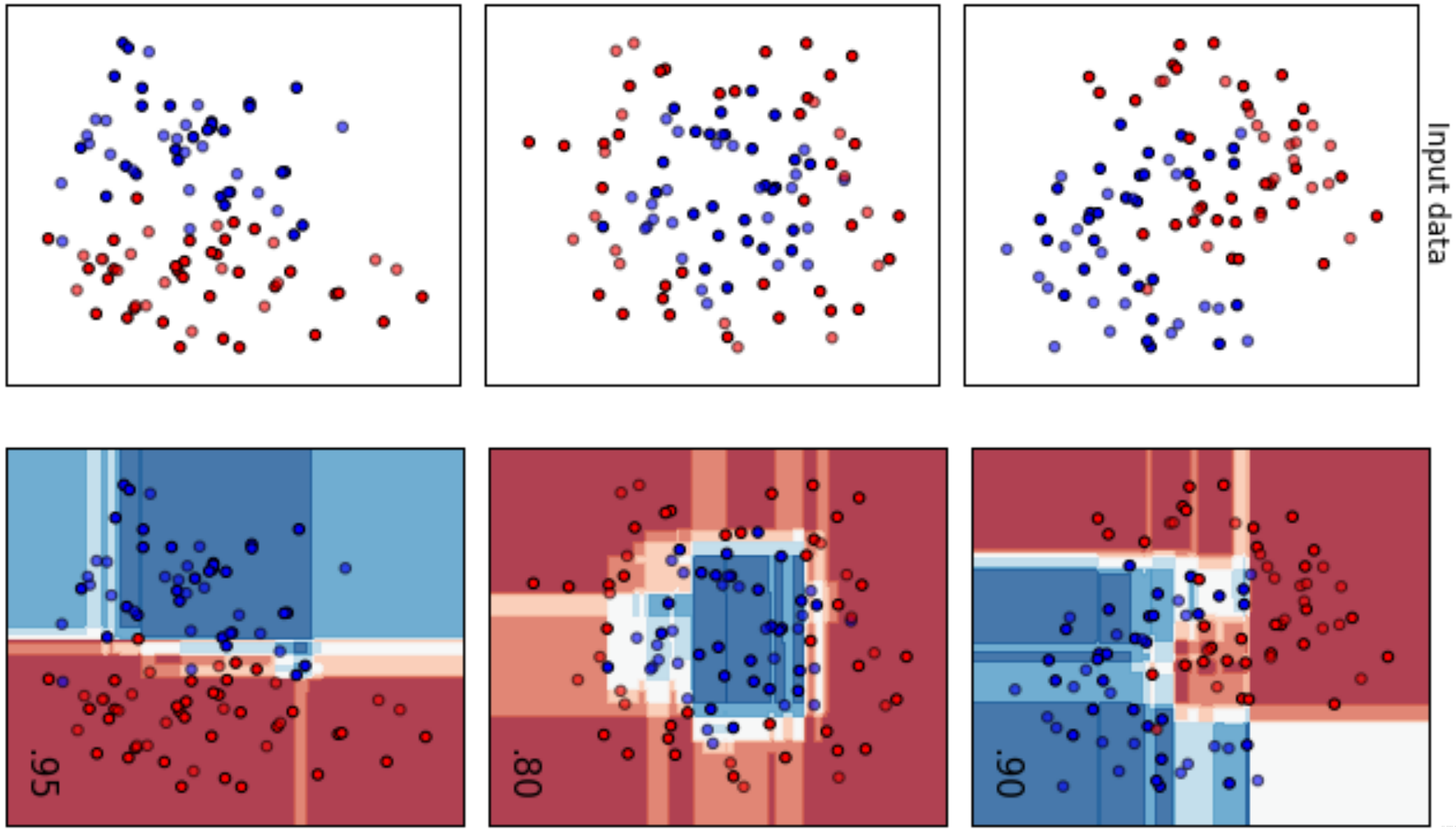


Random Patches samples both the training Instances as well as the Features.

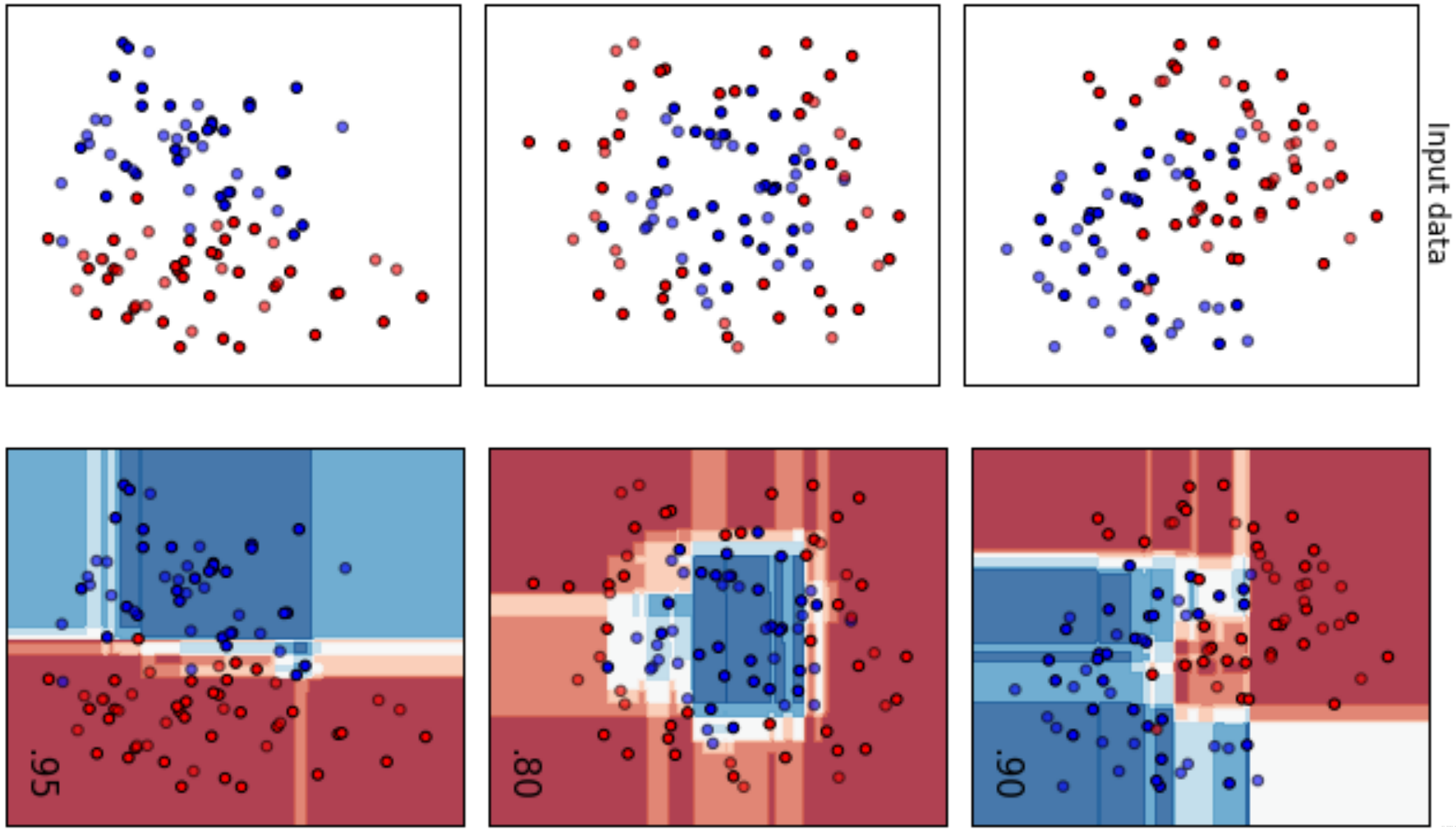
Random Subspaces keeps all the instances but samples features.

Bagging method – the trees are running in parallel (there is no interaction between them)

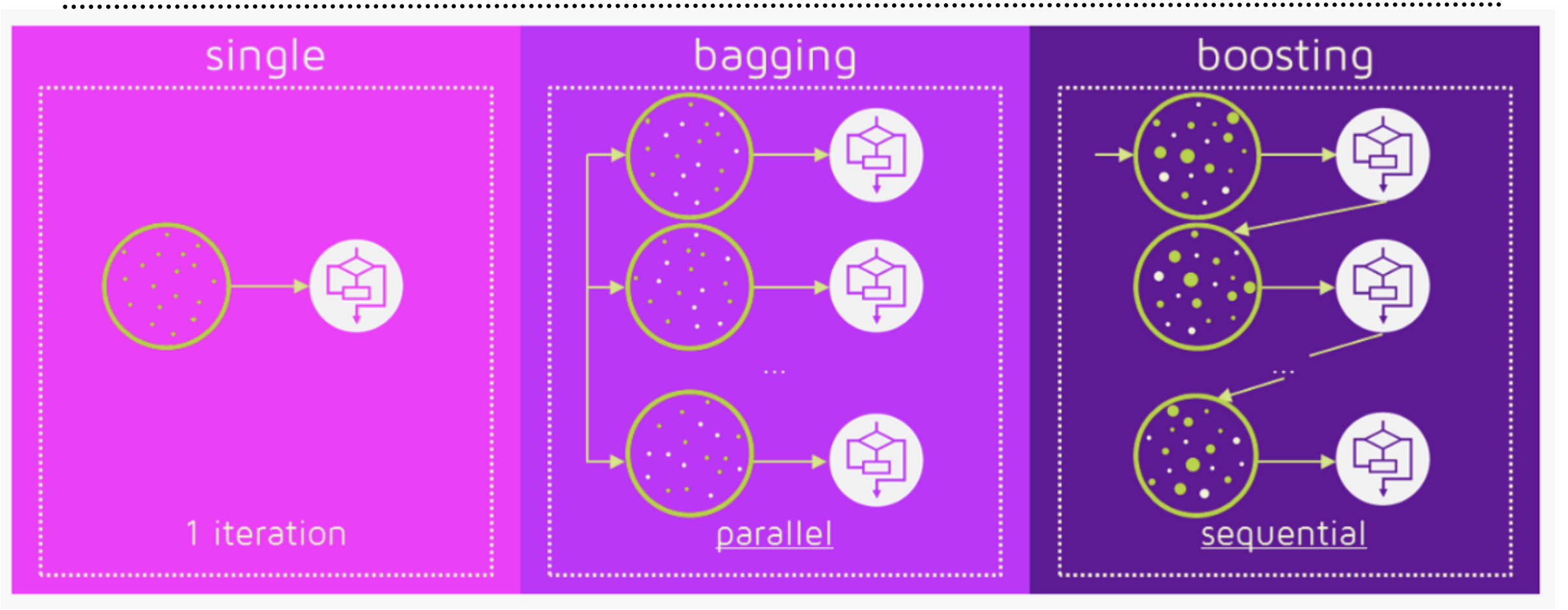
Bagging algorithms - Random forests



Extremely Random forests

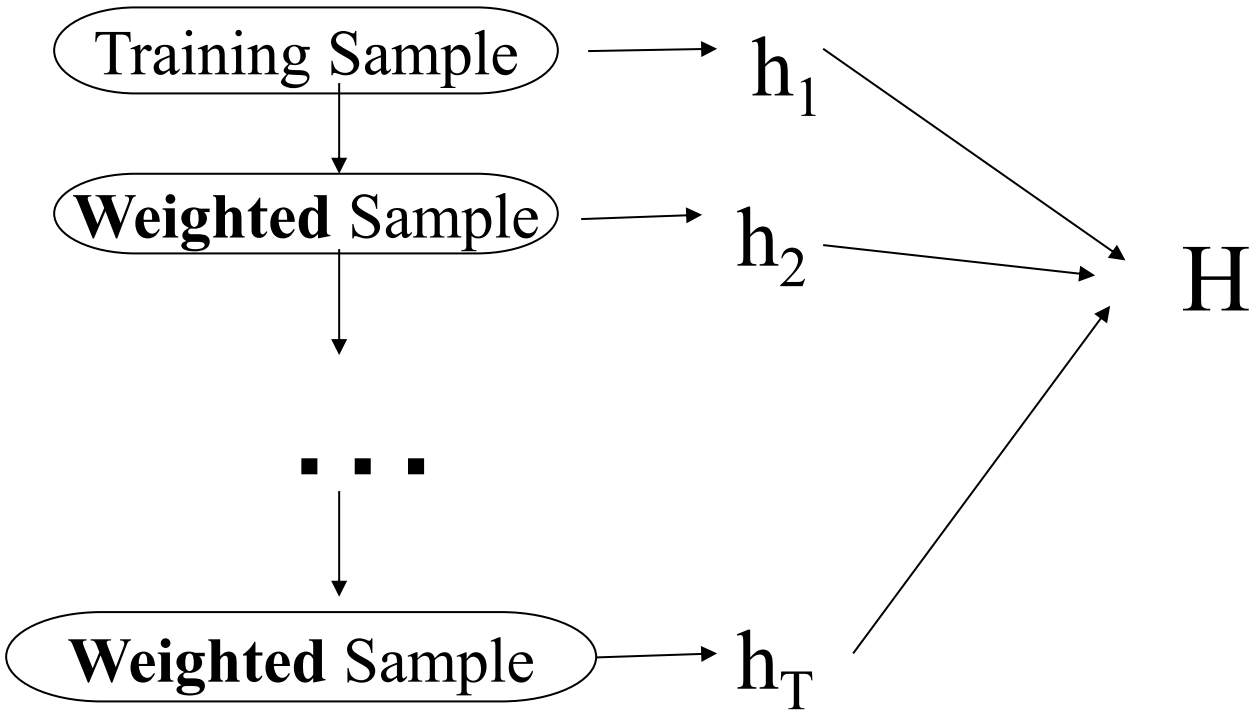


Boosting algorithms - Gradient boosted trees



In boosting – every tree is learning from other which in turn boosts the learning. Trees are learned sequentially (slow).

Adaboost



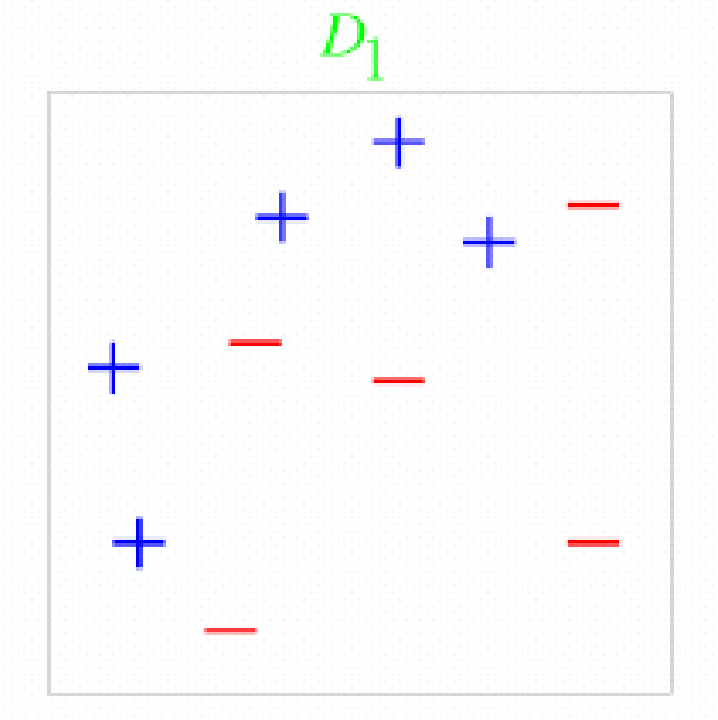
Train a set of weak hypotheses: $h_1, \ldots, h_T$.

The combined hypothesis H is a **weighted** majority vote of the T weak hypotheses.

Each hypothesis h_t has a weight α_t.

$$H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x))$$

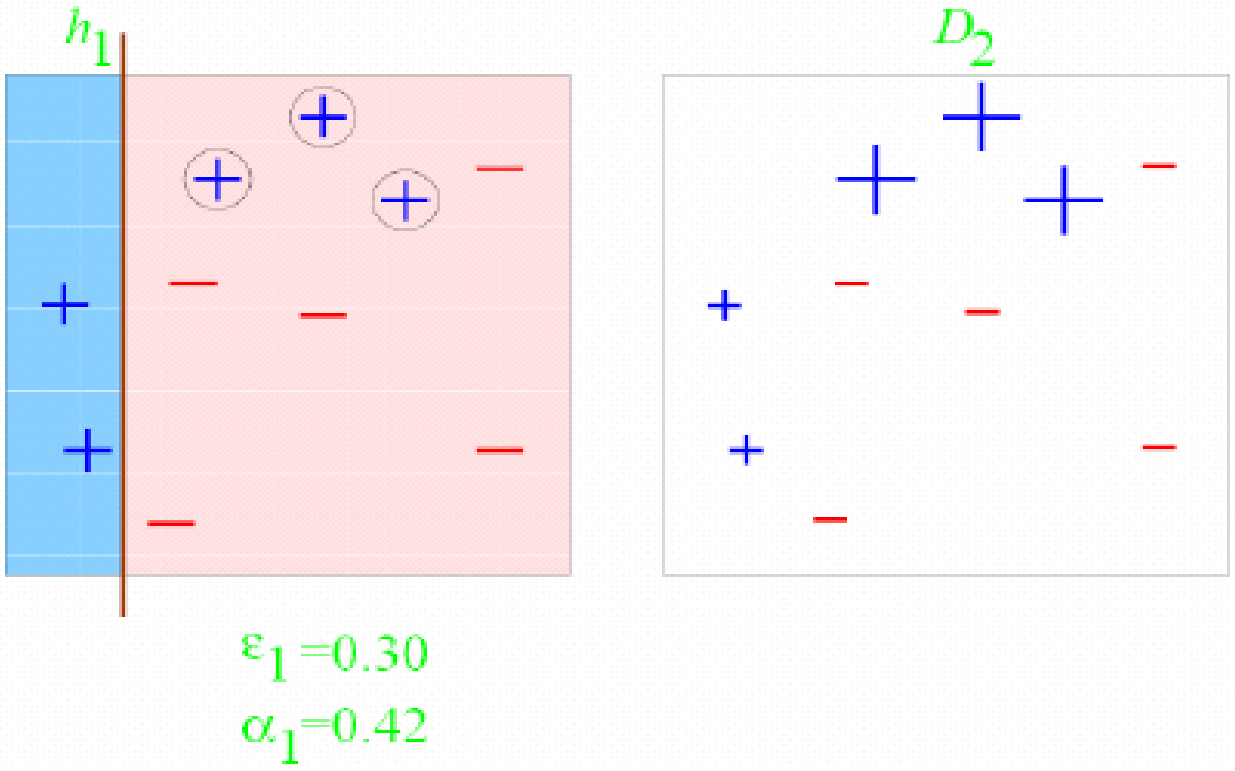
Adaboost



Training set: 10 points (represented by plus or minus)

Original Status: Equal Weights for all training samples

Adaboost

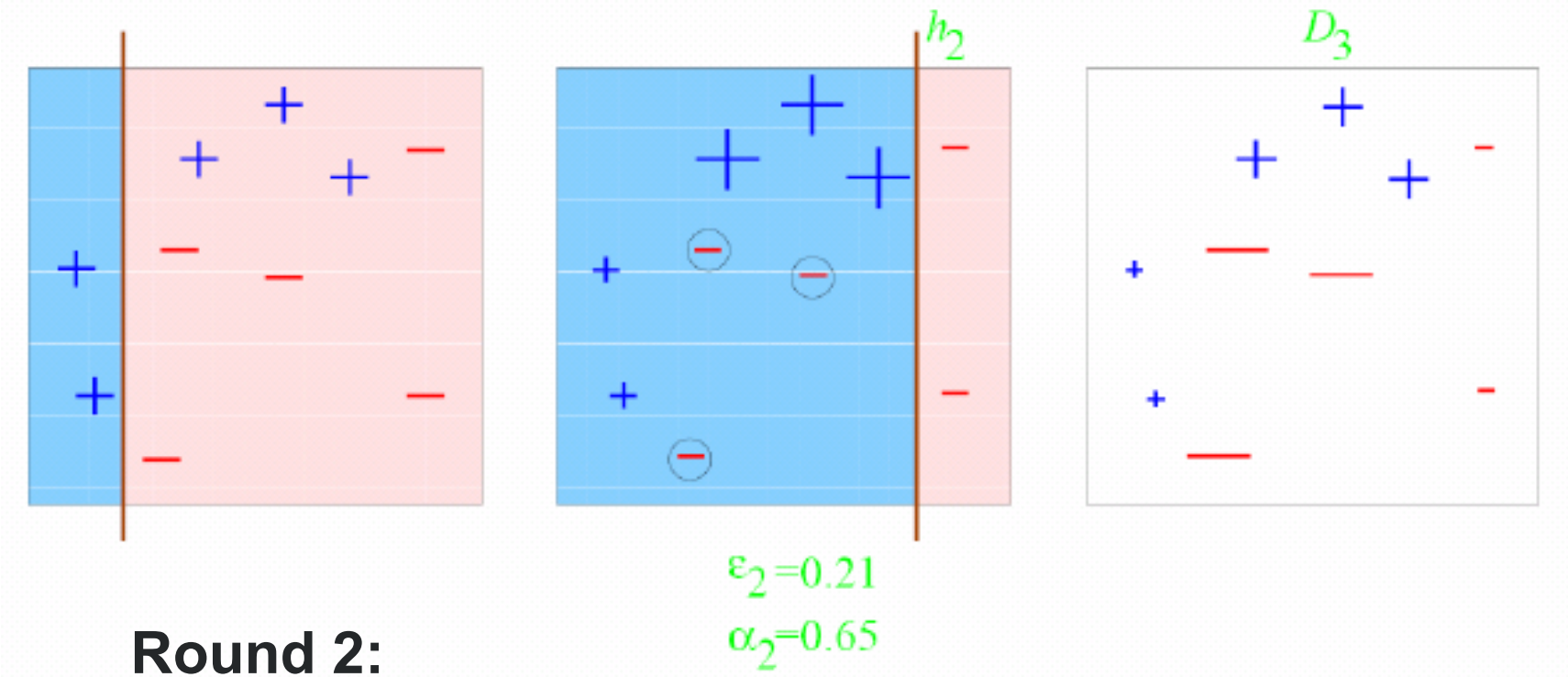


Round 1:

Three “plus” points are not correctly classified;

They are given higher weights.

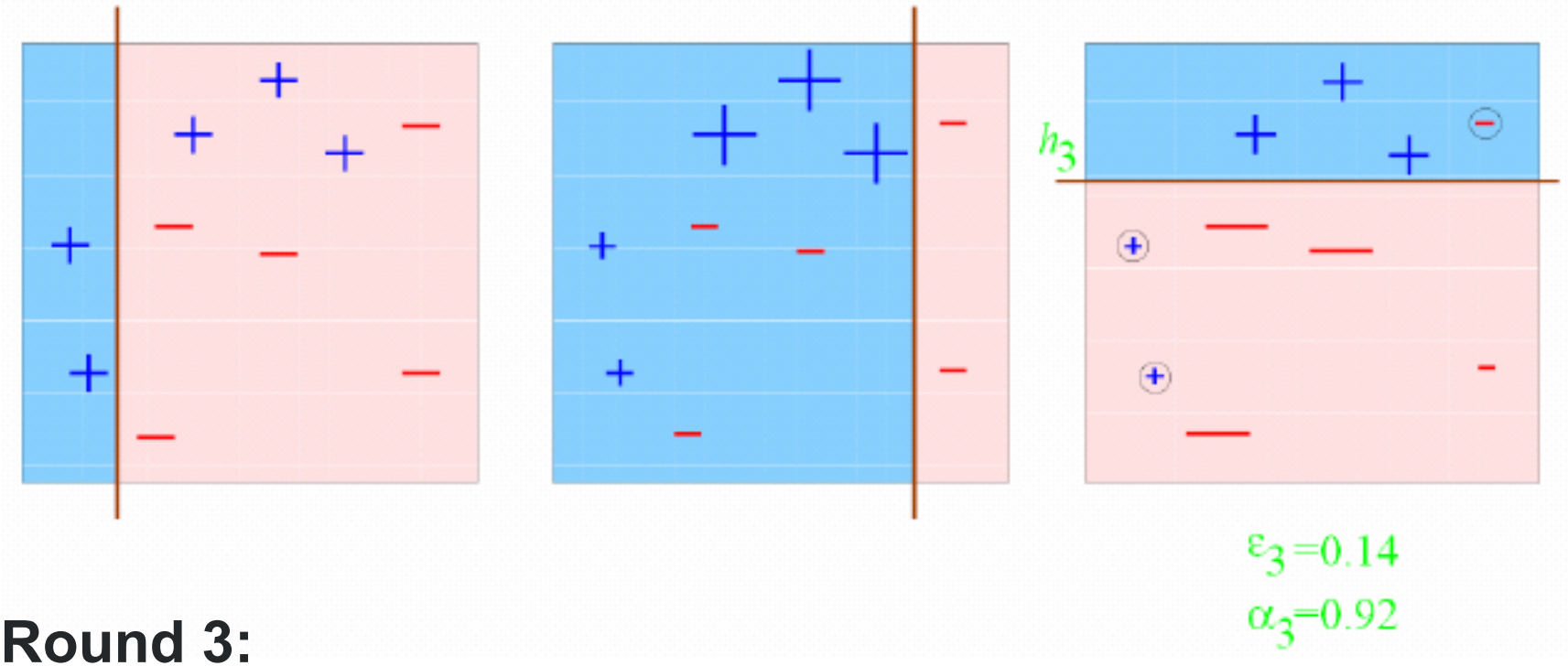
Adaboost



Round 2:

Three “minuses” points are not correctly classified;
They are given higher weights.

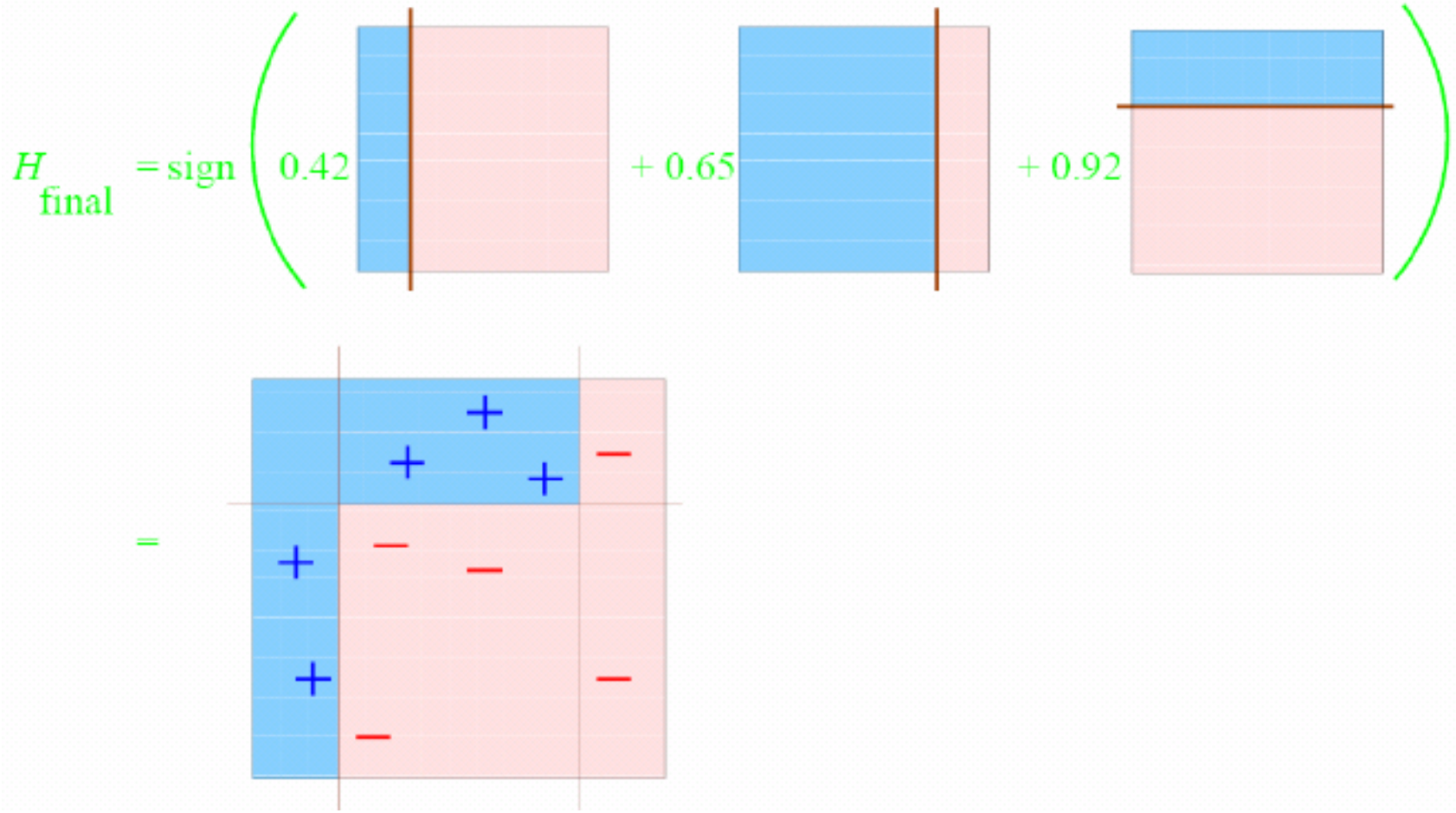
Adaboost



Round 3:

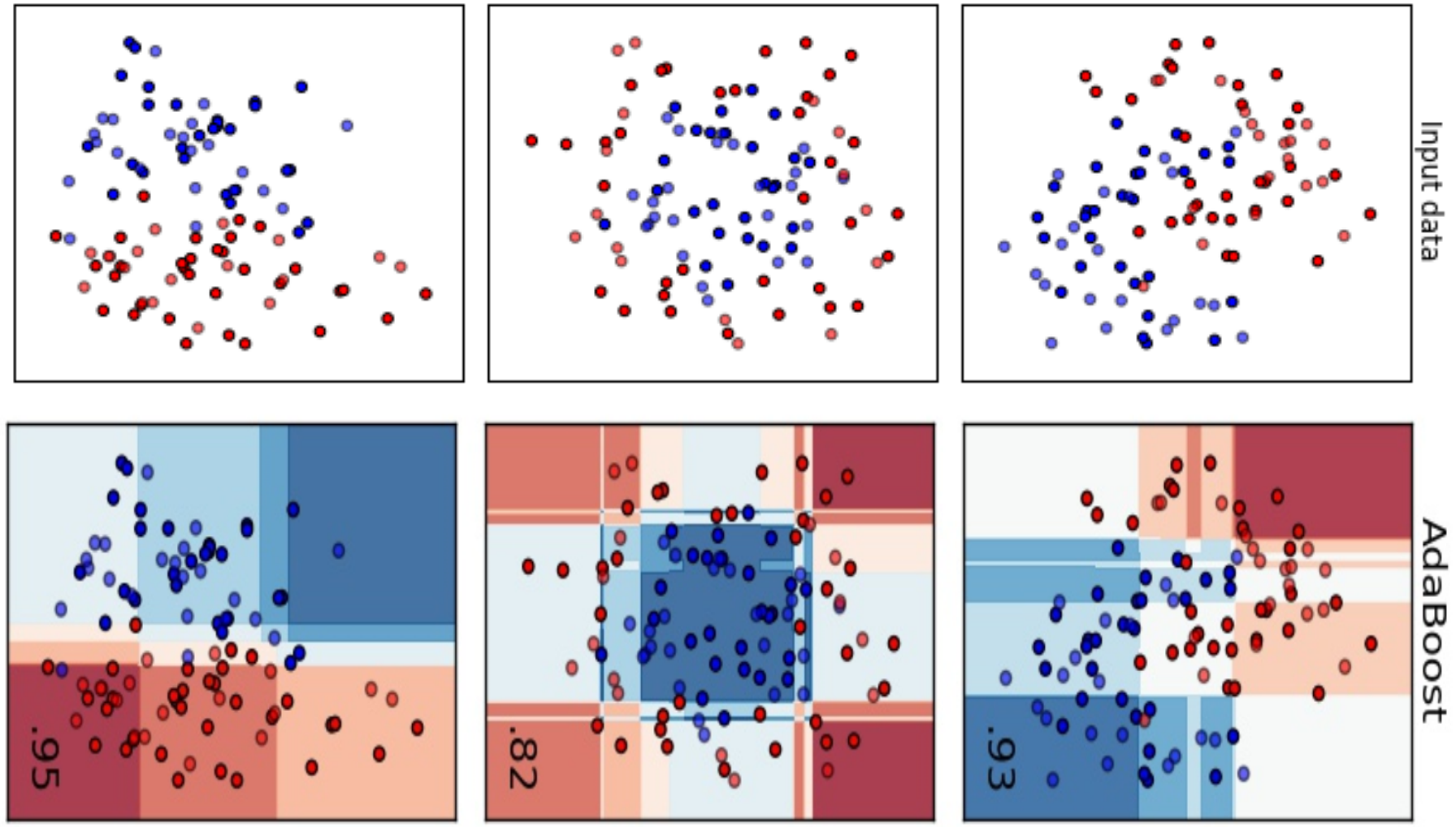
One “minus” and two “plus” points are not correctly classified;
They are given higher weights.

Adaboost

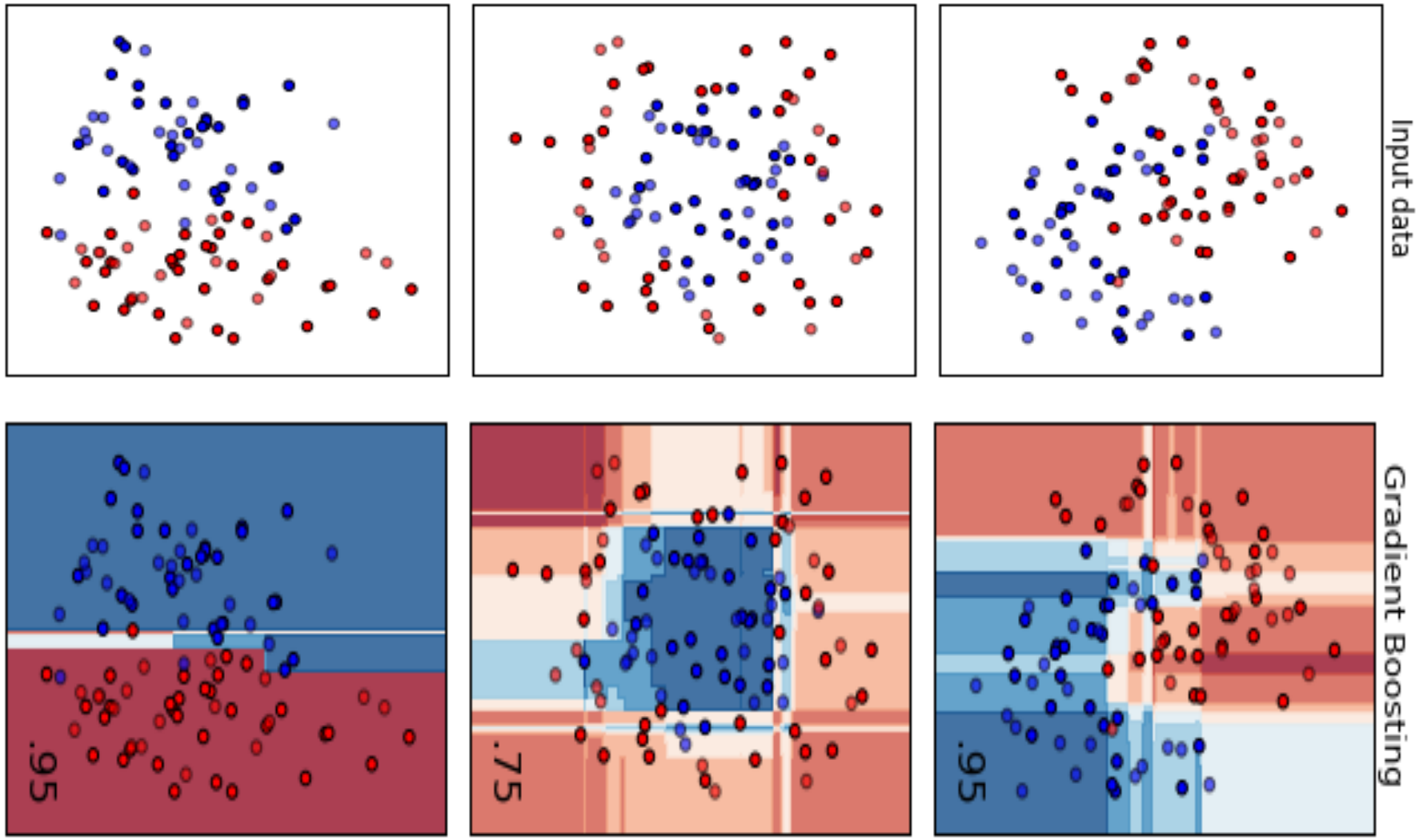


Final Classifier: integrate the three "weak" classifiers and obtain a final strong classifier.

Adaboost



Gradient boosted trees



Gradient boosted trees vs Random forests

- RF are much easier to tune than GBM
- RF are harder to overfit than GBM
- XGBoost is an optimized version of GBM (reduce the overfitting)

XGBoost

XGBoost has gained a lot of popularity recently and has been used in the most winning Kaggle competition models. It is powerful tool to have in your Data Science Toolbox.

XGBoost can work with Trees as well as Linear Models. I recommend reading the XGBoost documentation for further parameter tuning options.

XGBoost is a recent, most preferred and powerful gradient boosting method. Instead of making hard Yes and No Decision at the Leaf Nodes, XGBoost assigns positive and negative values to every decision made. All Trees are weak learners and provide a decisions slightly better than a random guess. But collectively averaged out, XGBoost performs really well.

Gradient boosted trees and Random forests

PARAMETERS TO CONSIDER

Number of trees – high number of trees – the model will have better results / the training classification will last longer

Criterion - the function to measure the quality of a split (Gini vs Entropy)

Max depth of the tree:
- high values – the model will overfit
- small values – the model will underfit

Learning rate (on Gradient Boosting algorithms):
Learning rate parameter shrinks the contribution of each tree. Lowering the value of learning rate increases the number of trees in the ensemble. Be vary that increasing the number of estimators to a large value may overfit the model.

Naïve Bayes

Naive Bayes classifiers are a family of simple probabilistic classifiers based on applying Bayes' theorem with strong (naive) independence assumptions between the features.

Likelihood

Prior

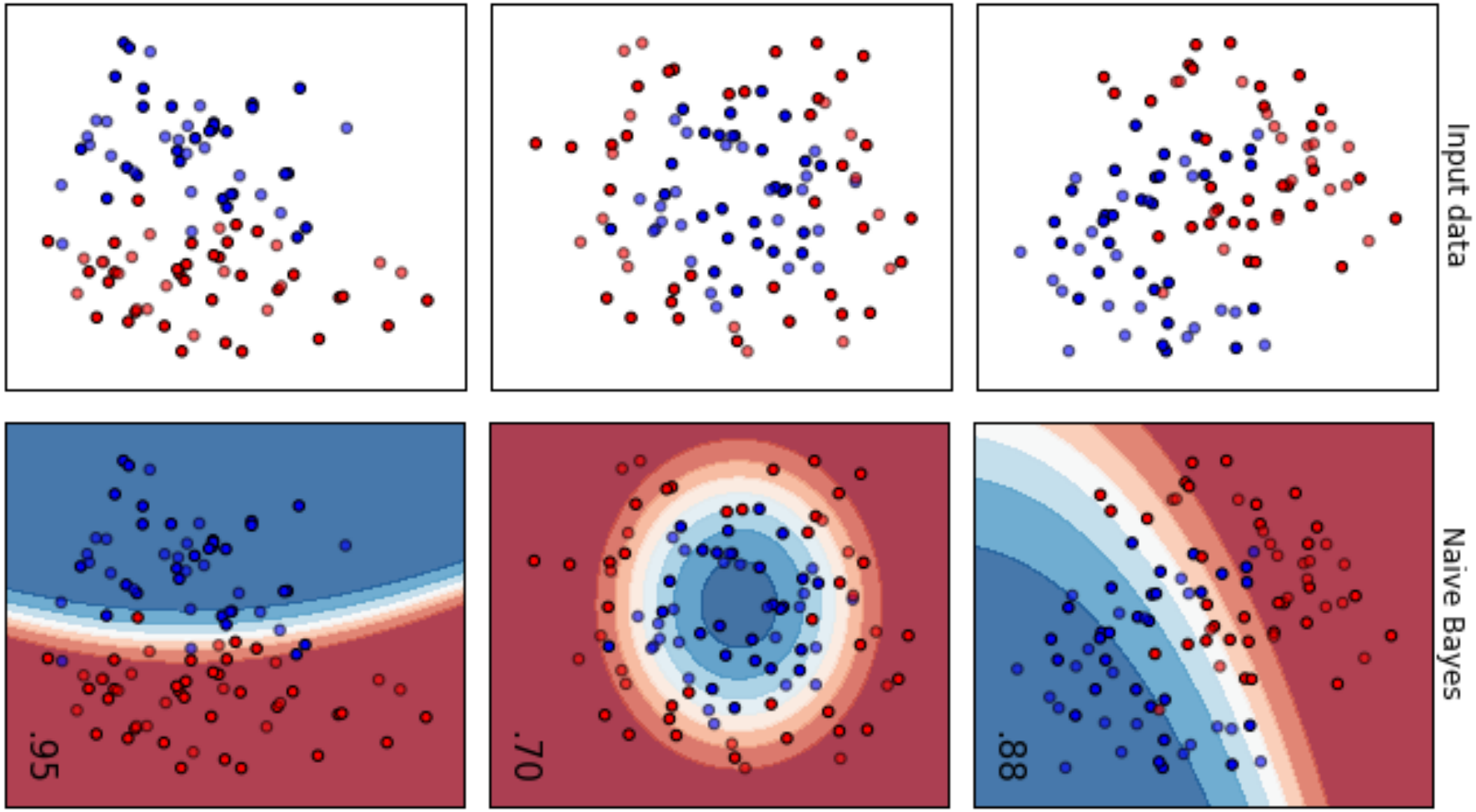
$$P(Y|X_1,\ldots,X_n) = \frac{P(X_1,\ldots,X_n|Y)P(Y)}{P(X_1,\ldots,X_n)}$$

Normalization Constant

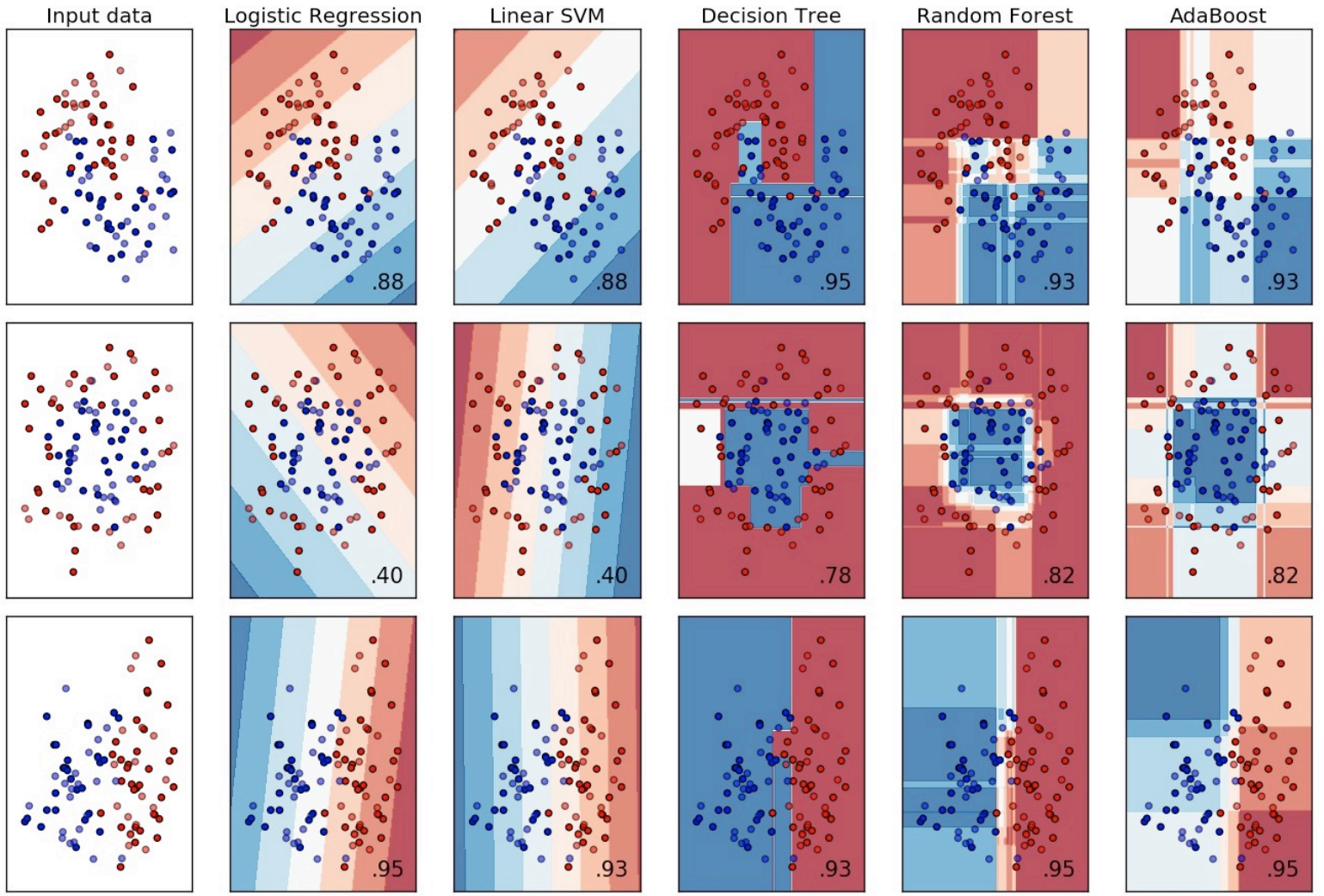
Detect the sales of ice-cream by using the weather forecasting

[Image from www.criminisi.com]

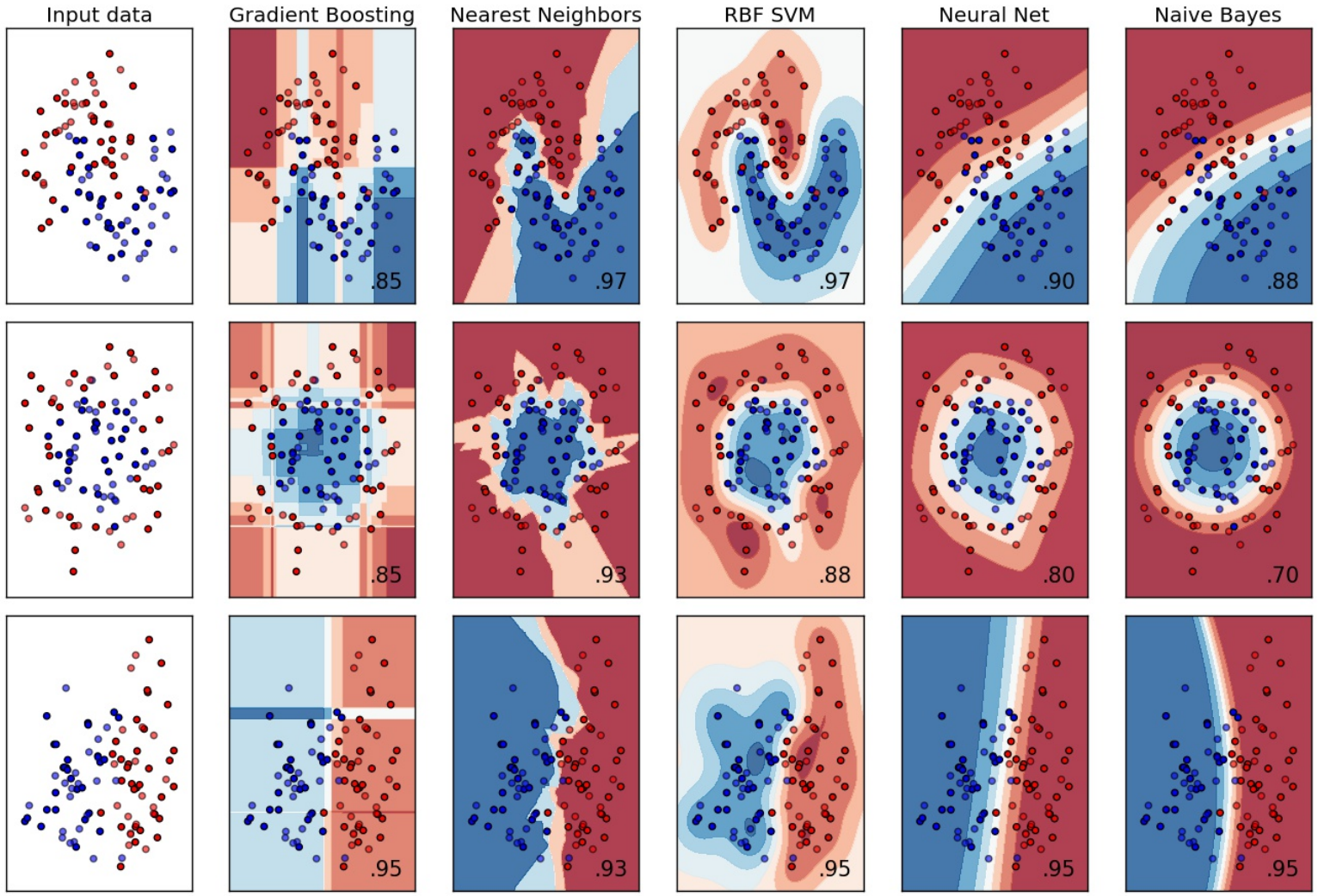
Naïve Bayes



Comparison between different methods



Comparison between different methods



Deep Learning vs classical Machine Learning

DO WE NEED HUNDREDS OF CLASSIFIERS TO SOLVE REAL WORLD CLASSIFICATION PROBLEMS?

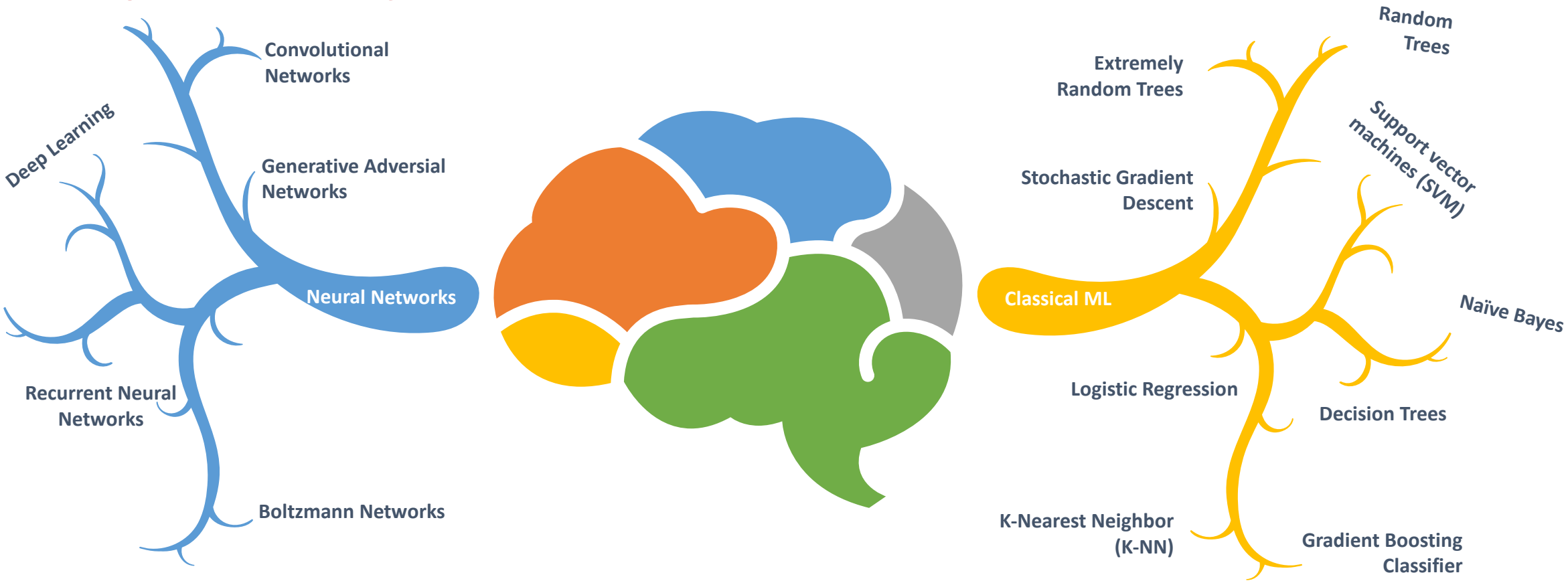
- Performed experiments with 179 classifiers arising from 17 families on 121 datasets (from UCI repository);
- They concluded that the classifiers most likely to be the bests are the random forest and SVM

[Manuel Fernandez-Delgado et al., 2015]

Deep Learning

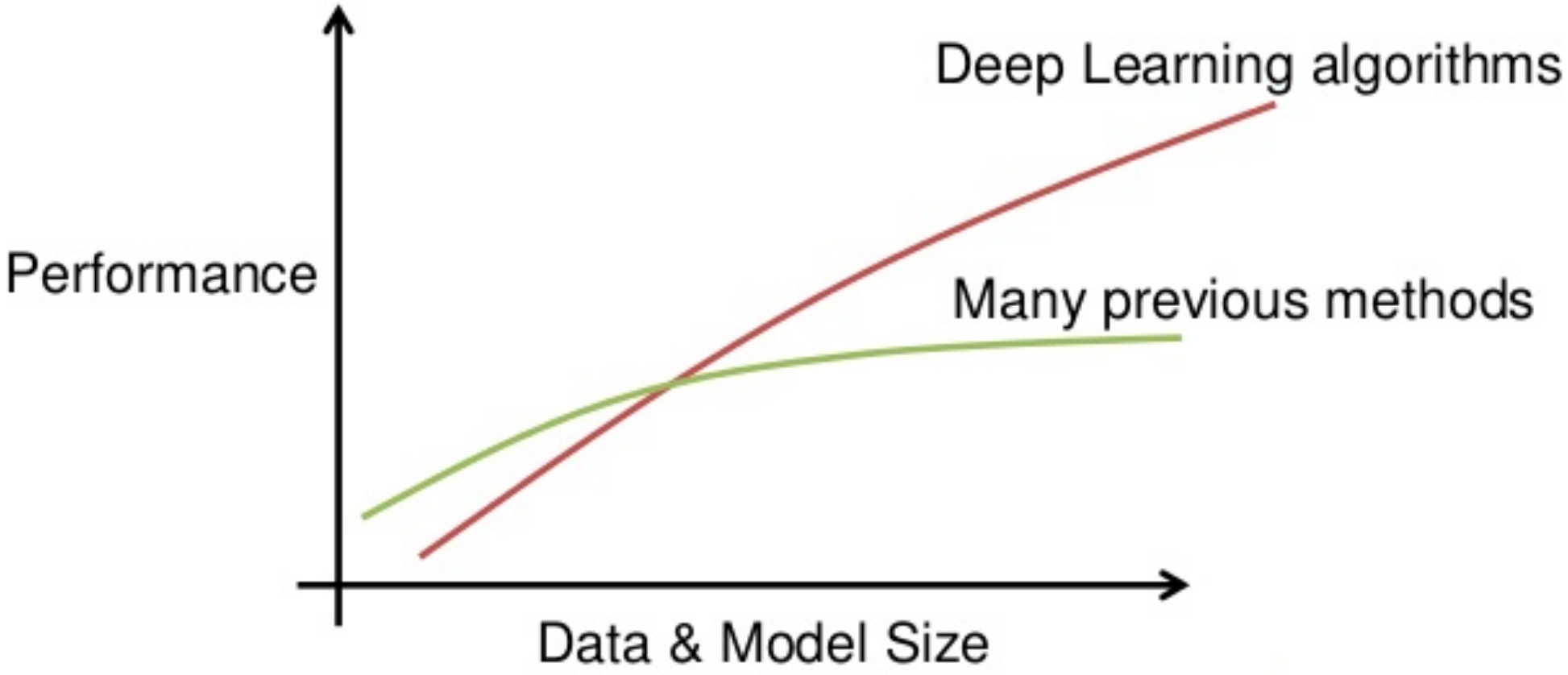
Deep Learning vs classical Machine Learning

SUPERVISED LEARNING

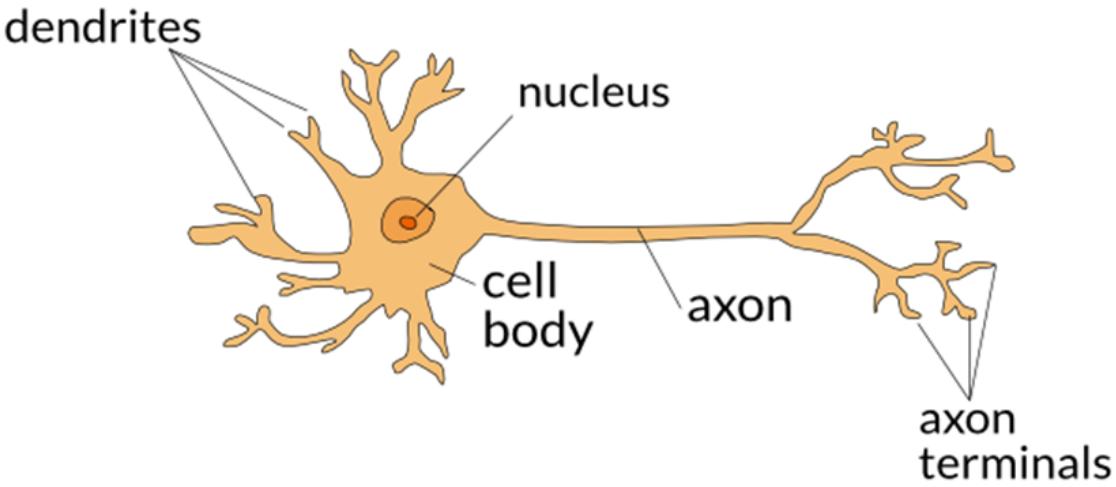


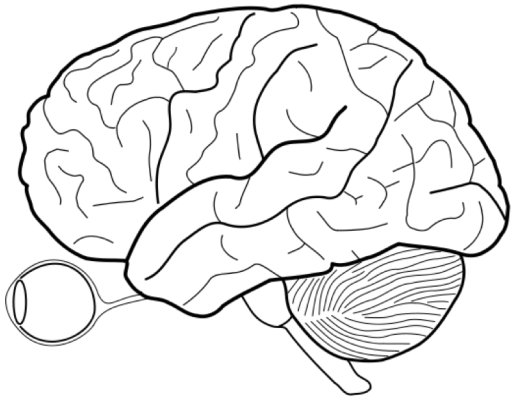
When Deep Learning is better than classical Machine Learning?

Deep Learning vs classical Machine Learning

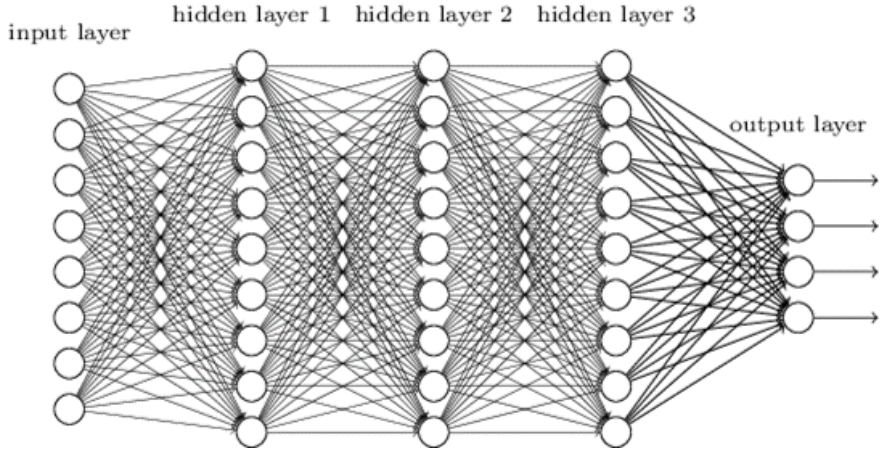


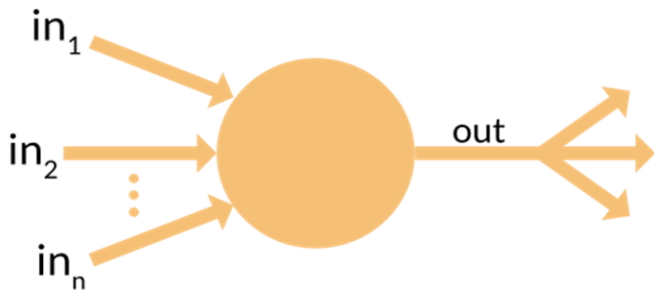
Neural networks





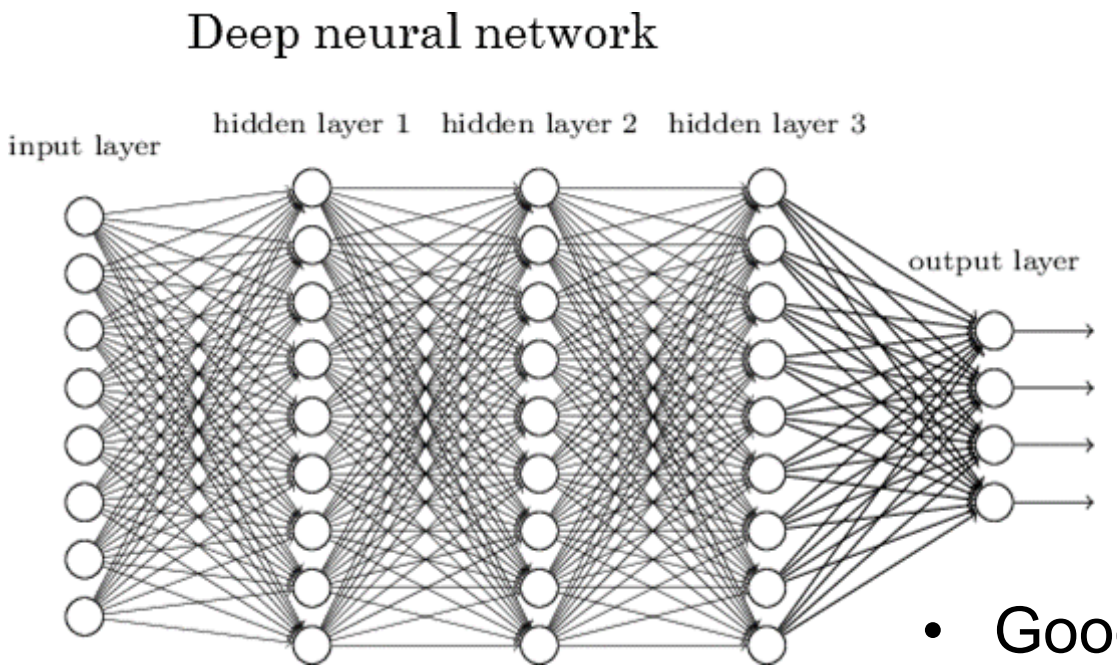
Deep neural network



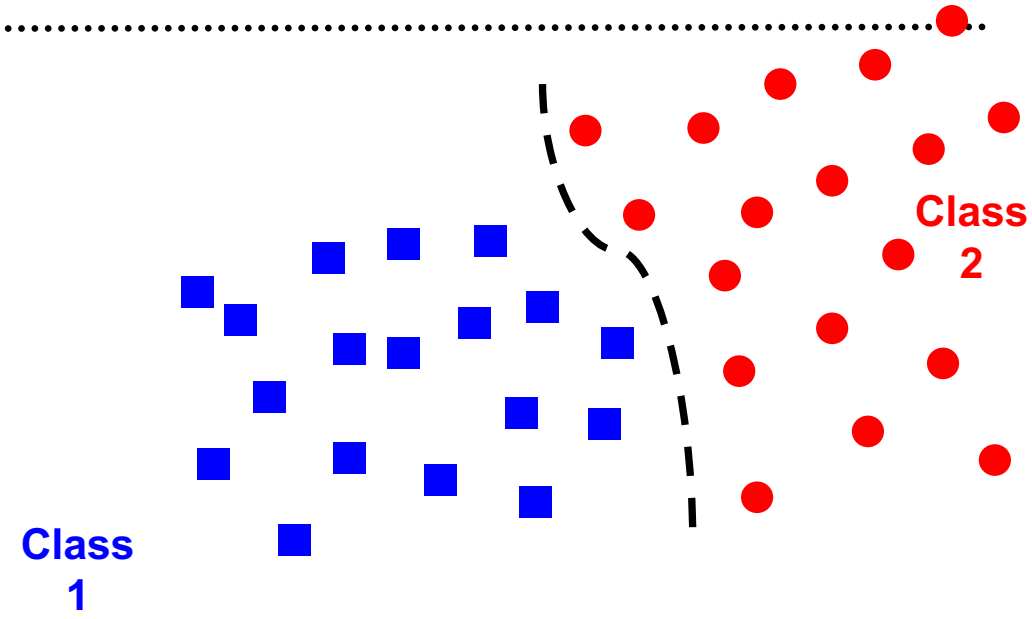


[Images from: https://appliedgo.net/perceptron, wiki.org and quora.com]

Neural networks

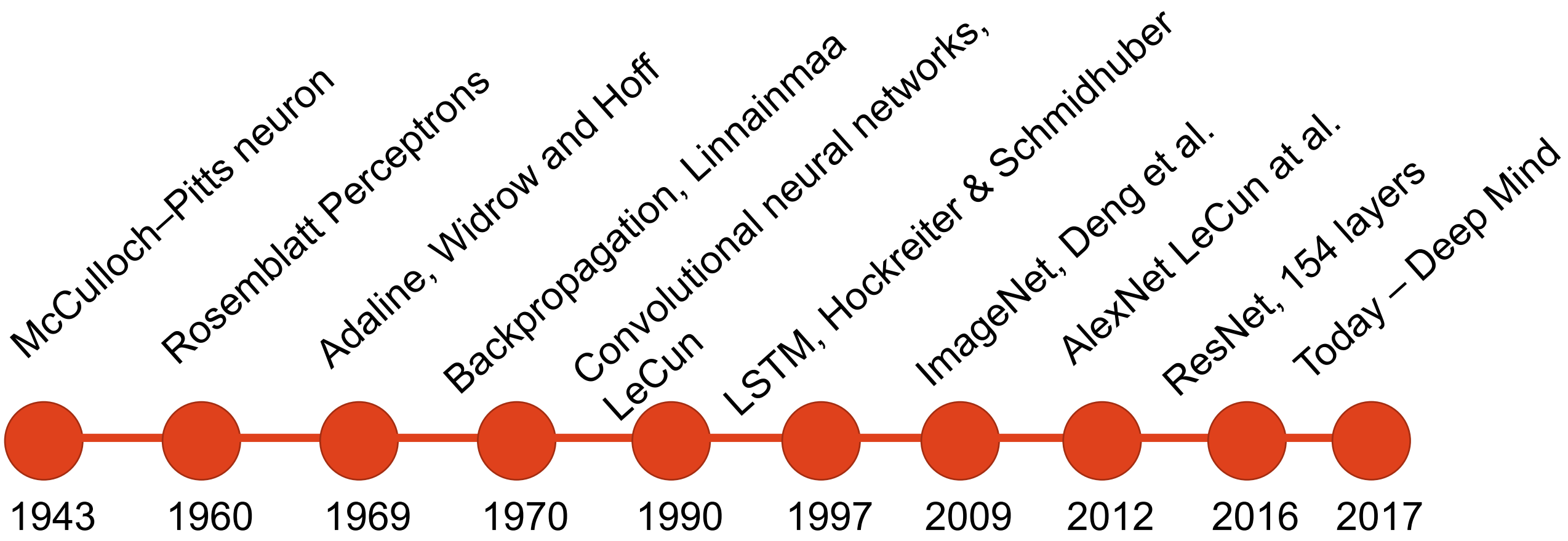


[Image from: http://quora.com]

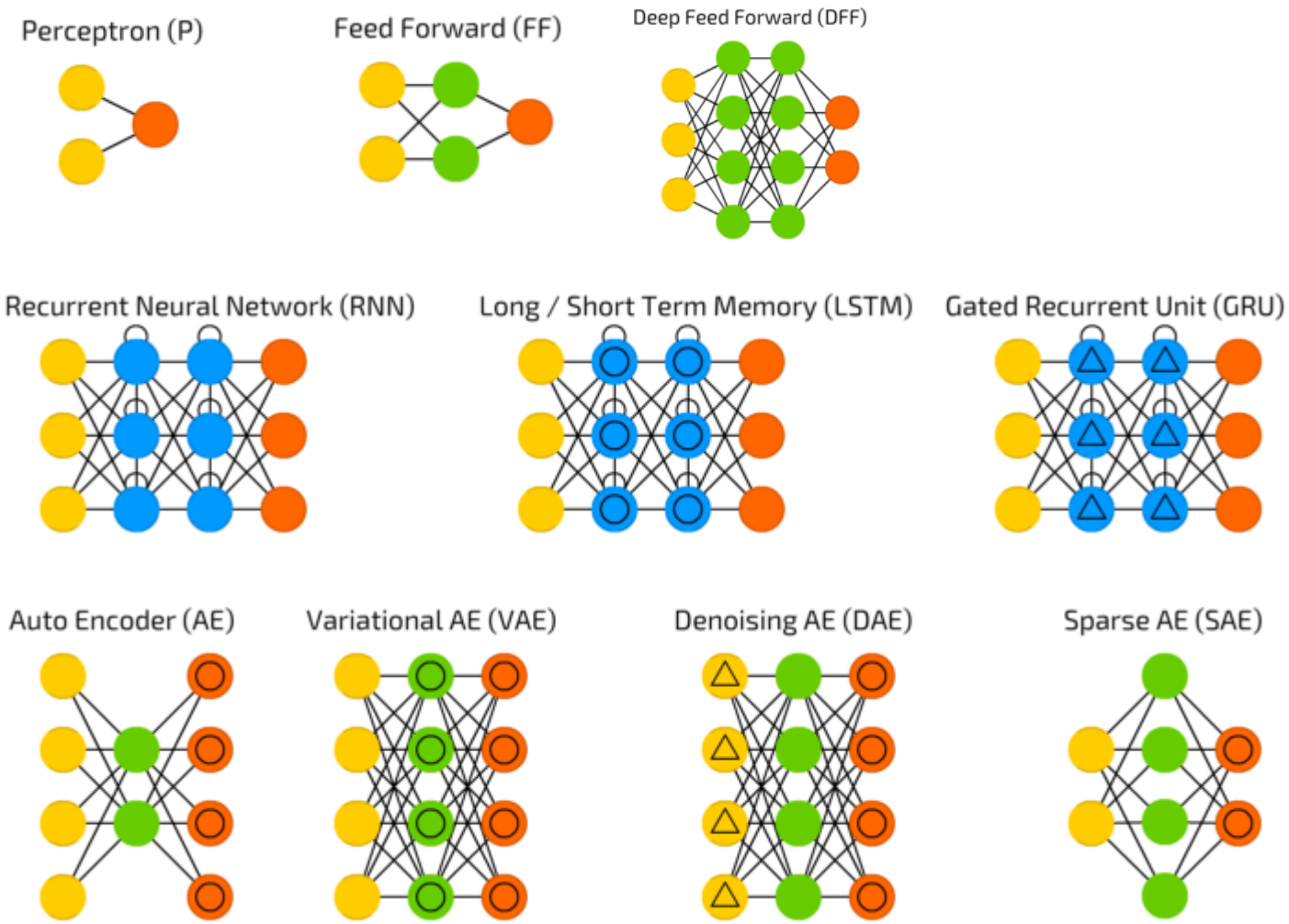


- Good to find the best separating plane between two classes.
- Complicated structure, with many parameters and several hyper-parameters, non – trivial to tune.
- Prone to overfitting.

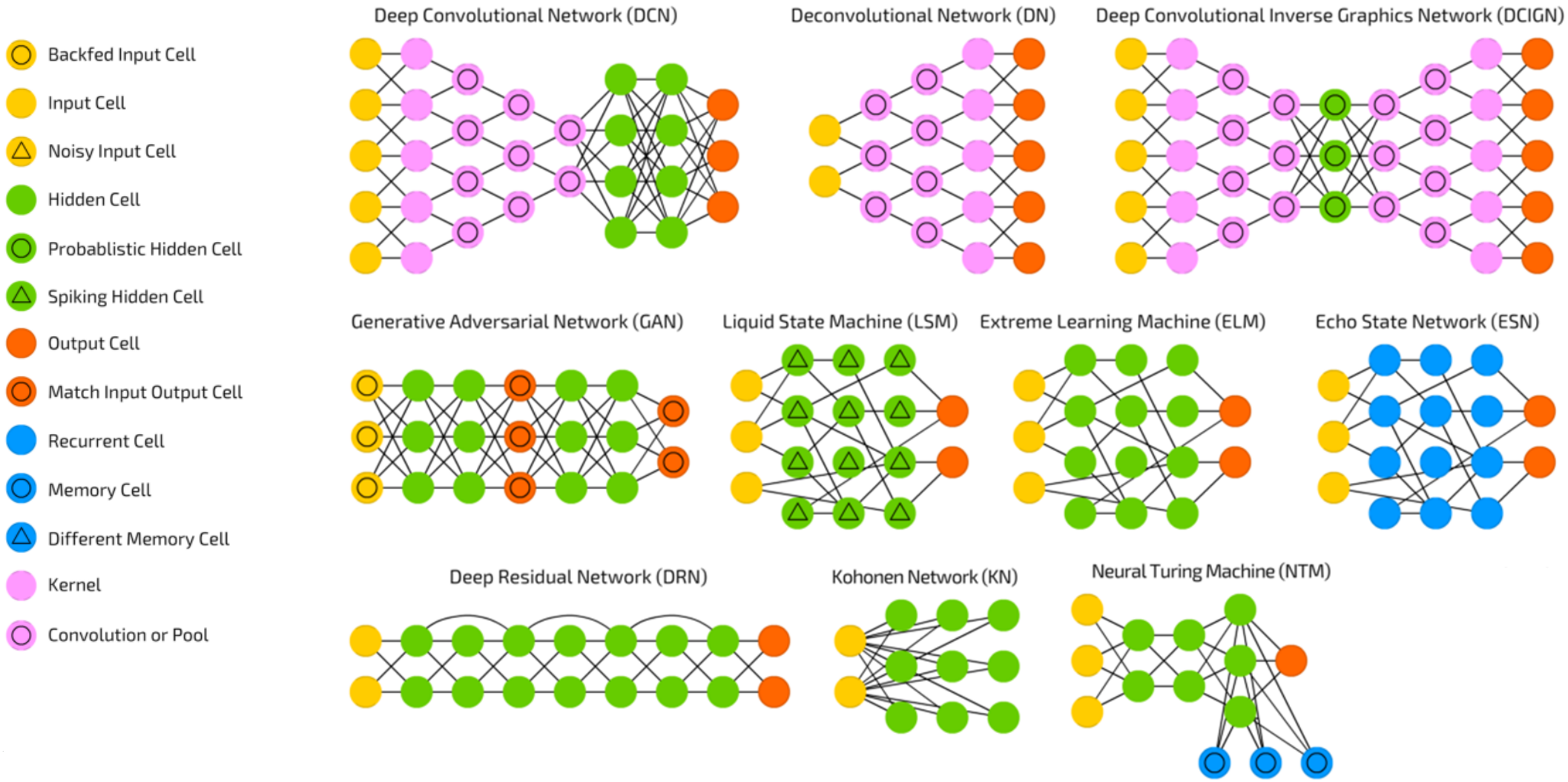
Neural networks history



Neural networks



Neural networks



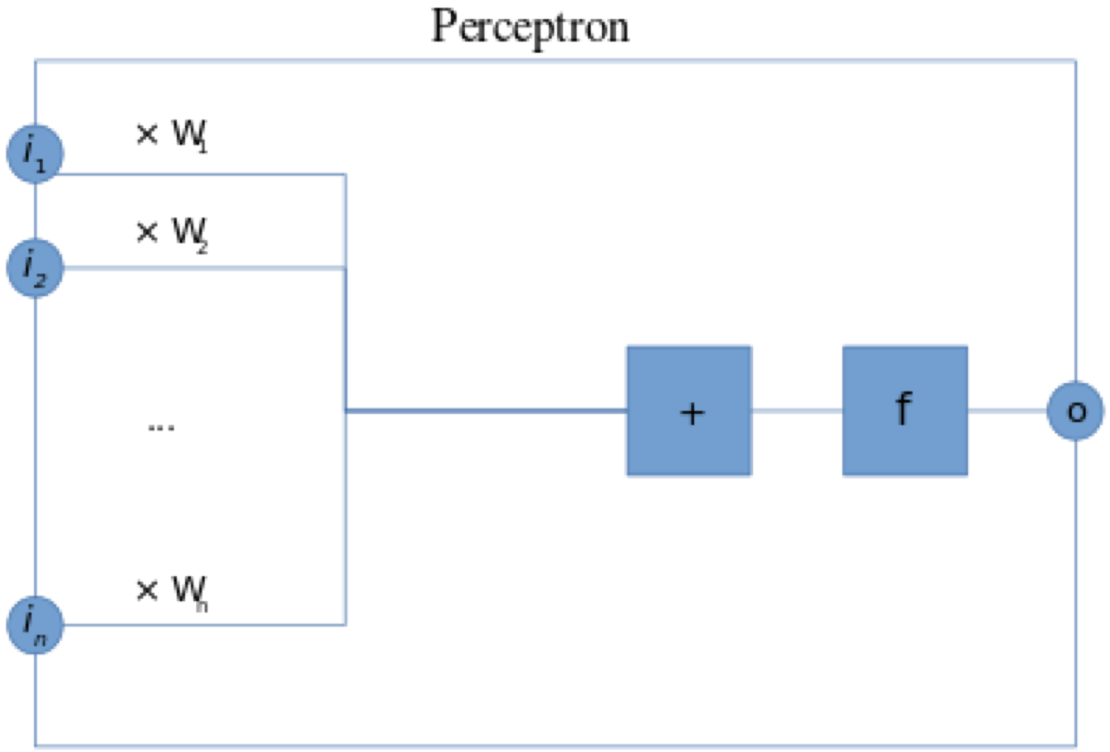
Neural networks

ONE NEURON

Status of existing checking account

Credit history

Purpose of the credit



$$o = f\left(\sum_{k=1}^{n} i_k \cdot W_k\right)$$

Neural networks

TWO LAYERS OF NEURONS

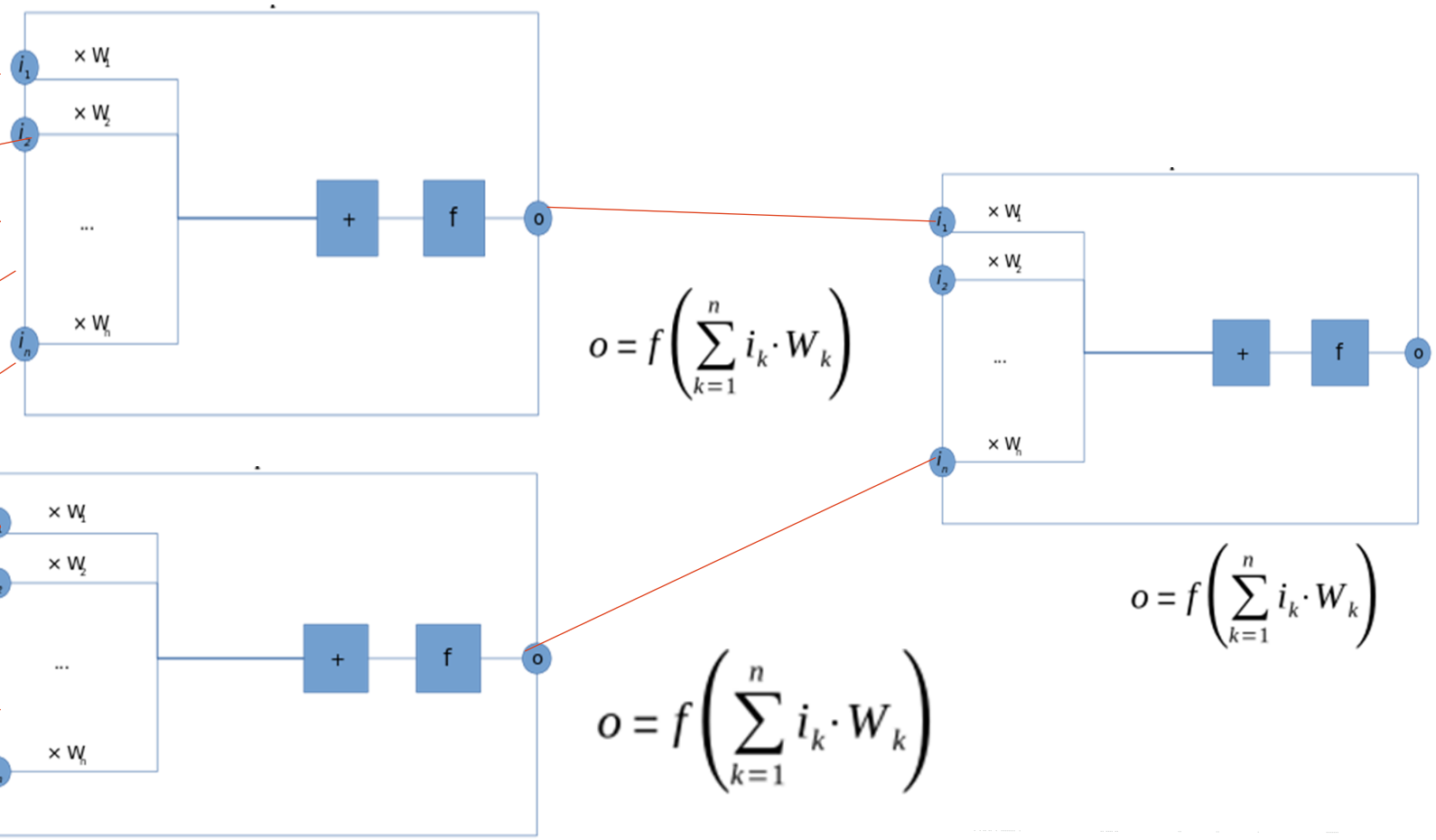
Status of existing checking account

Credit history

Present employment since

Purpose of the credit

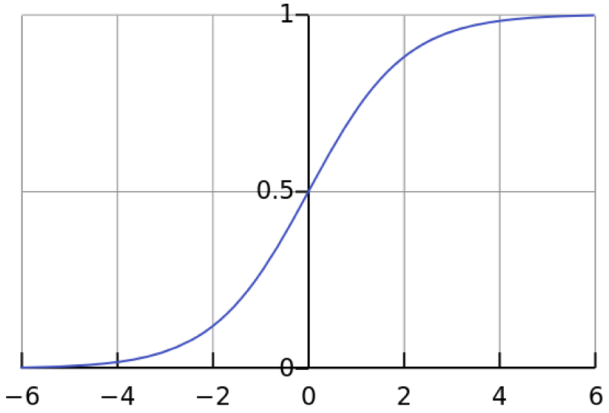
Credit amount



Neural networks

ACTIVATION FUNCTIONS

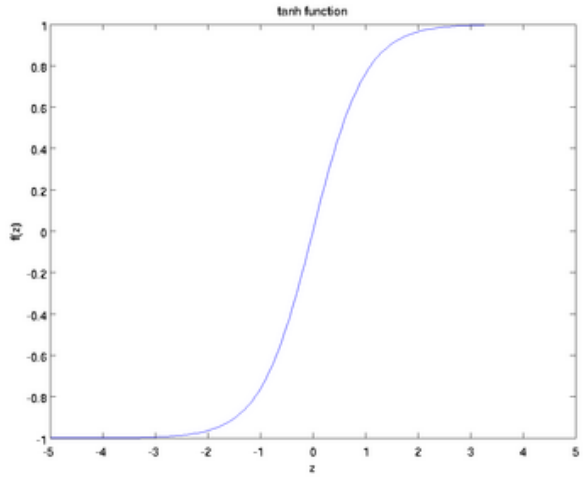
Sigmoid Function



$$A = \frac{1}{1+e^{-x}}$$

- Sigmoid outputs are not zero-centered
- Sigmoids saturate and kill gradients.

Tanh Function

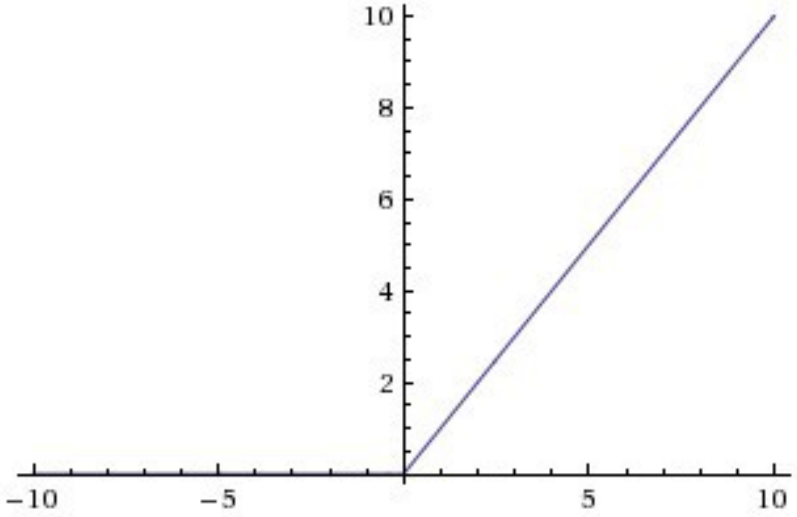


$$f(x) = tanh(x) = \frac{2}{1+e^{-2x}} - 1$$

Neural networks

ACTIVATION FUNCTIONS

RELU Function



$$f(x) = x^+ = \max(0, x),$$

- has accelerated convergence

- it does not saturate

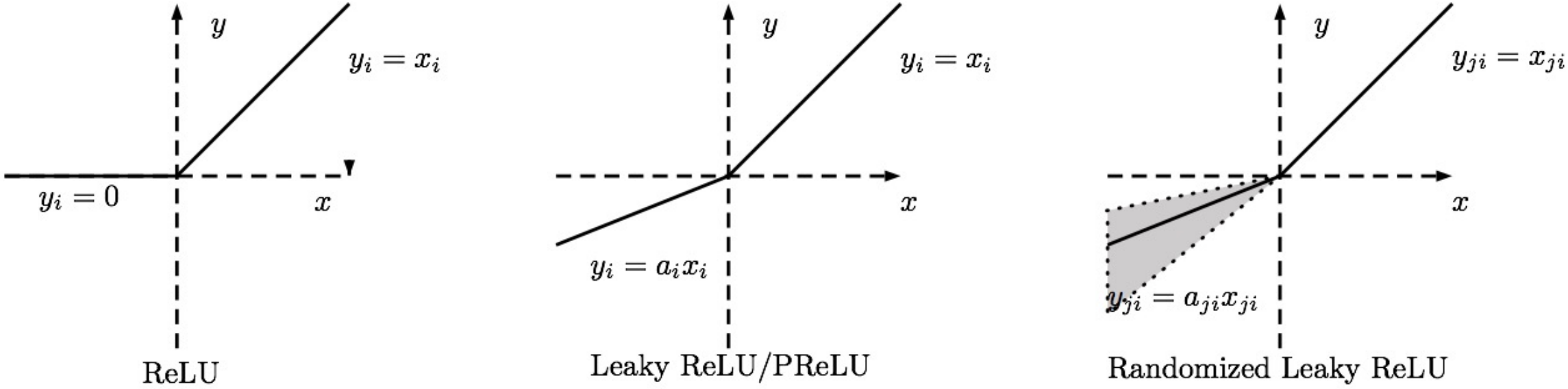
- involve inexpensive operations (a simple threshold)

- ReLU units can die during the training (the neuron will never activate)
  - more sensible to learning rate parameter

Neural networks

ACTIVATION FUNCTIONS

Leaky RELU

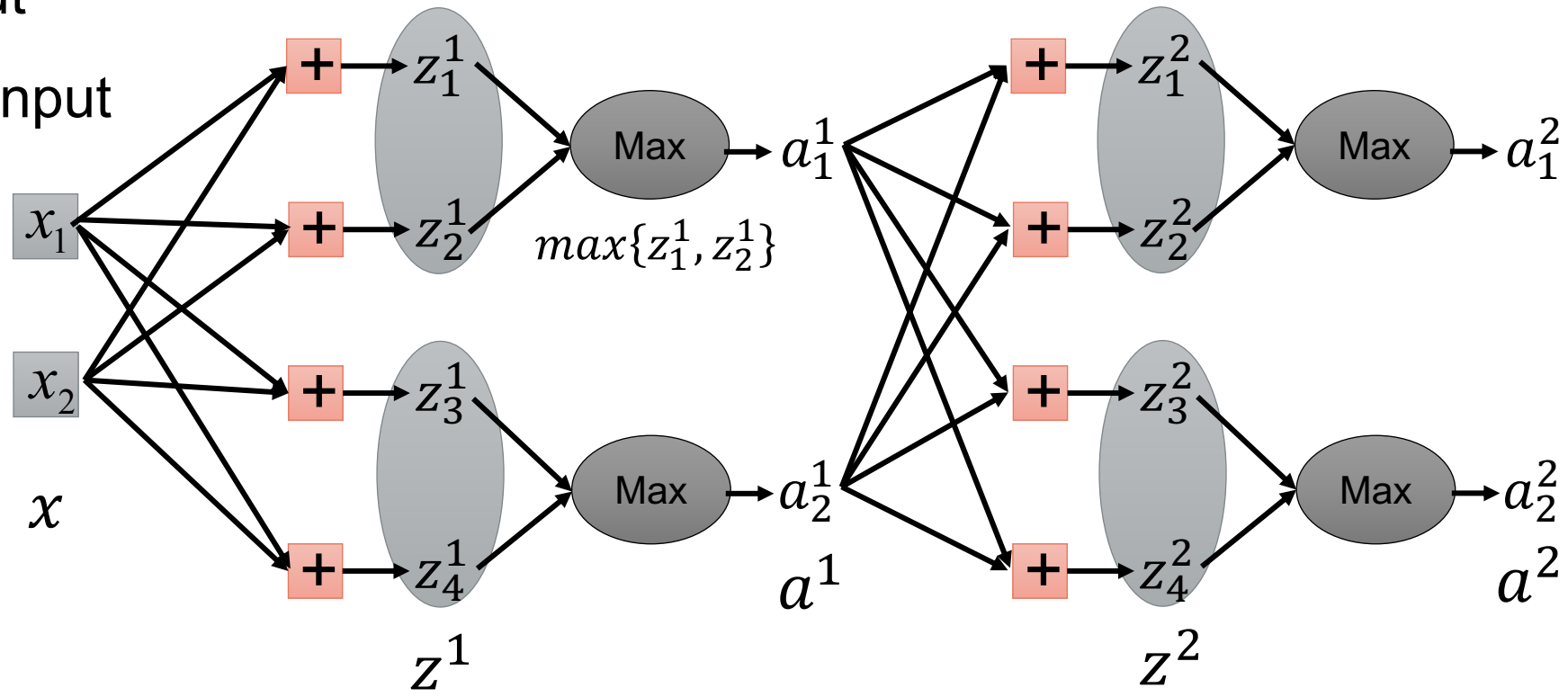


Some people report success with this form of activation function, but the results are not always consistent.

Neural networks

ACTIVATION FUNCTIONS

Maxout



$$z^1 = W^1 x$$

$$z^2 = W^2 a^1$$

[Goodfellow et al.]

Neural networks

ACTIVATION FUNCTIONS

Softmax

The idea of softmax is to define a new type of output layer for our neural networks

$$a_j^L = \frac{e^{z_j^L}}{\sum_k e^{z_k^L}},$$

Neural networks

SWISS KNIFE "*What neuron type should I use?*"

- Use the ReLU non-linearity, be careful with your learning rates and possibly monitor the fraction of "dead" units in a network.
- If this concerns you, give Leaky ReLU or Maxout a try. Never use sigmoid. Try tanh, but expect it to work worse than ReLU/Maxout.
- Use Softmax for the final layer

Regularization techniques for overfitting decrease

L1 / L2 NORMALIZATION

- Penalize large weights, and tend to make the network to prefer small weights.

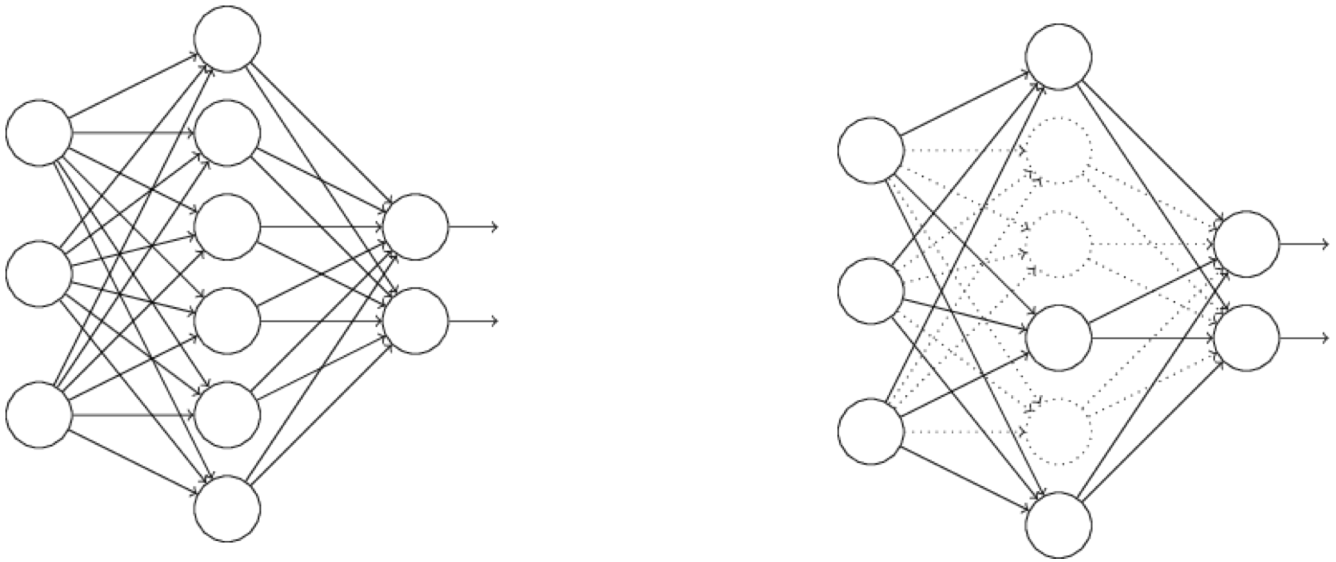
L1 normalization $$S = \sum_{i=1}^{n} |y_i - f(x_i)|.$$

L2 normalization $$S = \sum_{i=1}^{n} (y_i - f(x_i))^2$$

Regularization techniques for overfitting decrease

DROPOUT

- Penalize the neurons.
- Unlike L1 and L2 regularization, dropout doesn't rely on modifying the cost function. Instead, in dropout we modify the network itself.



Regularization techniques for overfitting decrease

DROPOUT

- When teams up, if everyone expect the partner will do the work, nothing will be done finally.
- However, if you know your partner will dropout, you will do better.

- When testing, no one dropout actually, so obtaining good results eventually.

- You will need more neurons to train your model.

Regularization techniques for overfitting decrease

BATCH NORMALIZATION

- Normalize distribution of each input feature in each layer across each minibatch to N(0, 1) learn the scale and shift

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β
Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}} \qquad \text{// normalize}$$

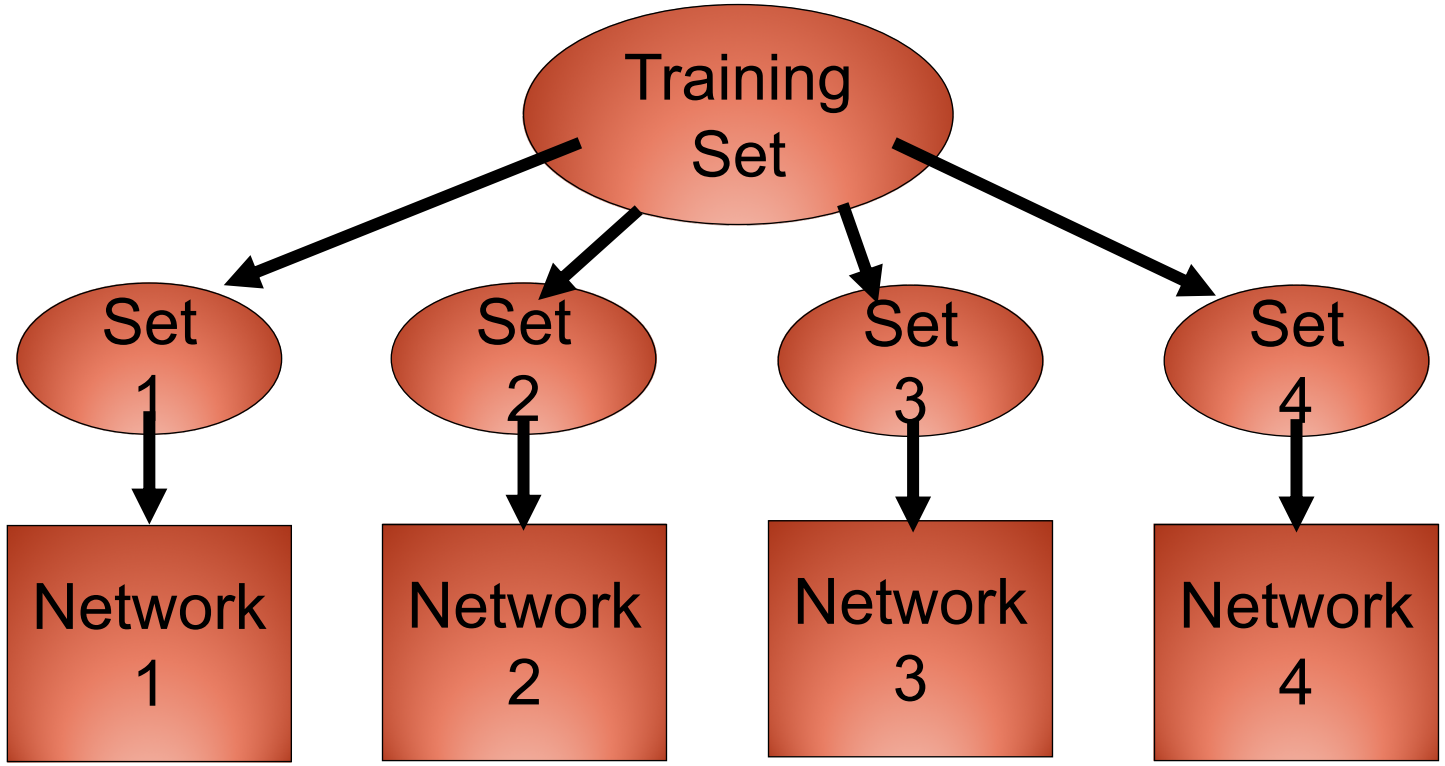
$$y_i \leftarrow \gamma\widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

Algorithm 1: Batch Normalizing Transform, applied to activation x over a mini-batch.

[Ioffe et al.]

Regularization techniques for overfitting decrease

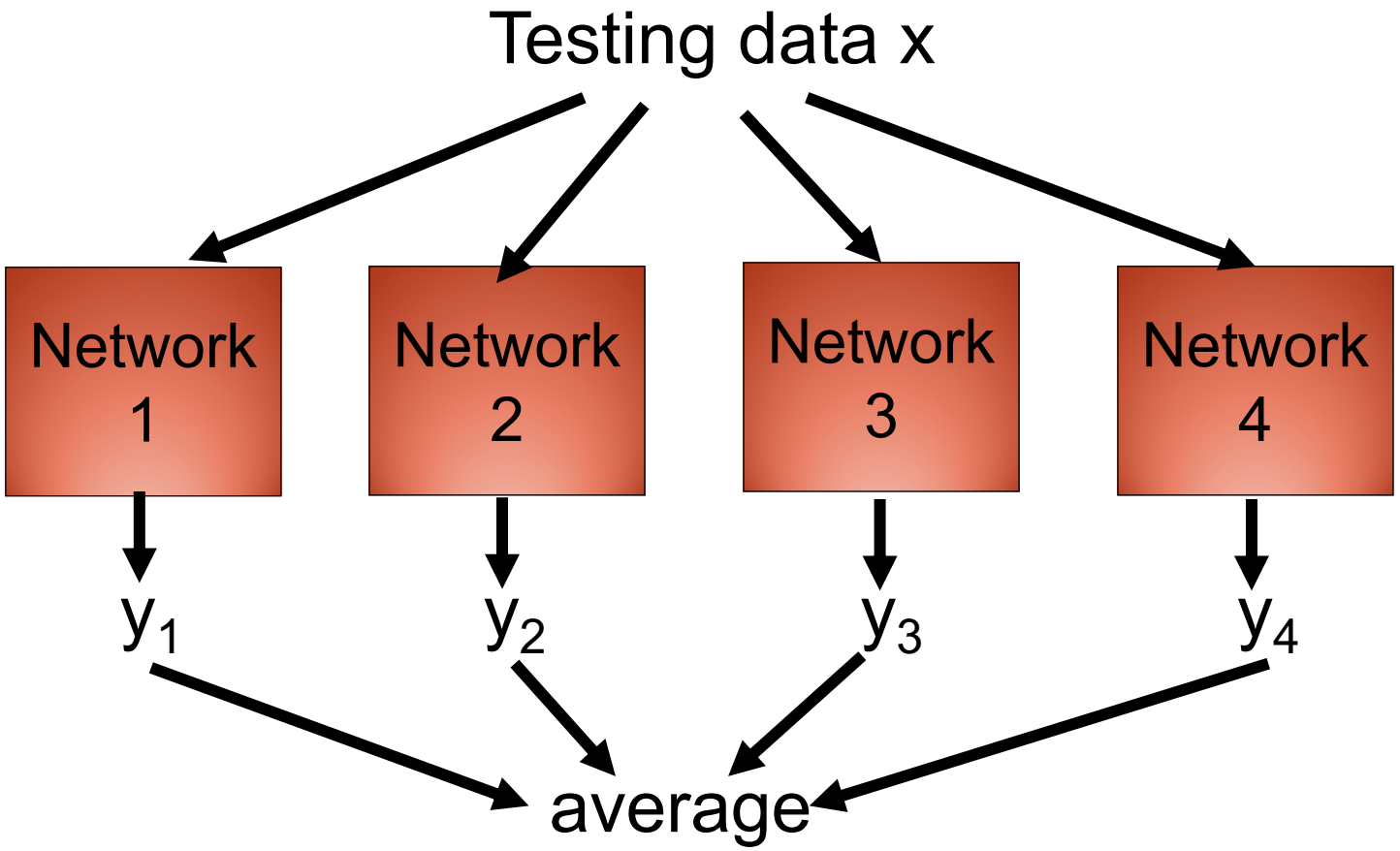
DROPOUT ENSEMBLE



Train a bunch of networks with different structures

Regularization techniques for overfitting decrease

DROPOUT ENSEMBLE



Regularization techniques for overfitting decrease

ARTIFICIALLY EXPANDING THE TRAINING DATA

[image source: http://wiki.com]

Regularization techniques for overfitting decrease

ARTIFICIALLY EXPANDING THE TRAINING DATA

Regularization techniques for overfitting decrease

ARTIFICIALLY EXPANDING THE TRAINING DATA

[Goodfellow et al.]

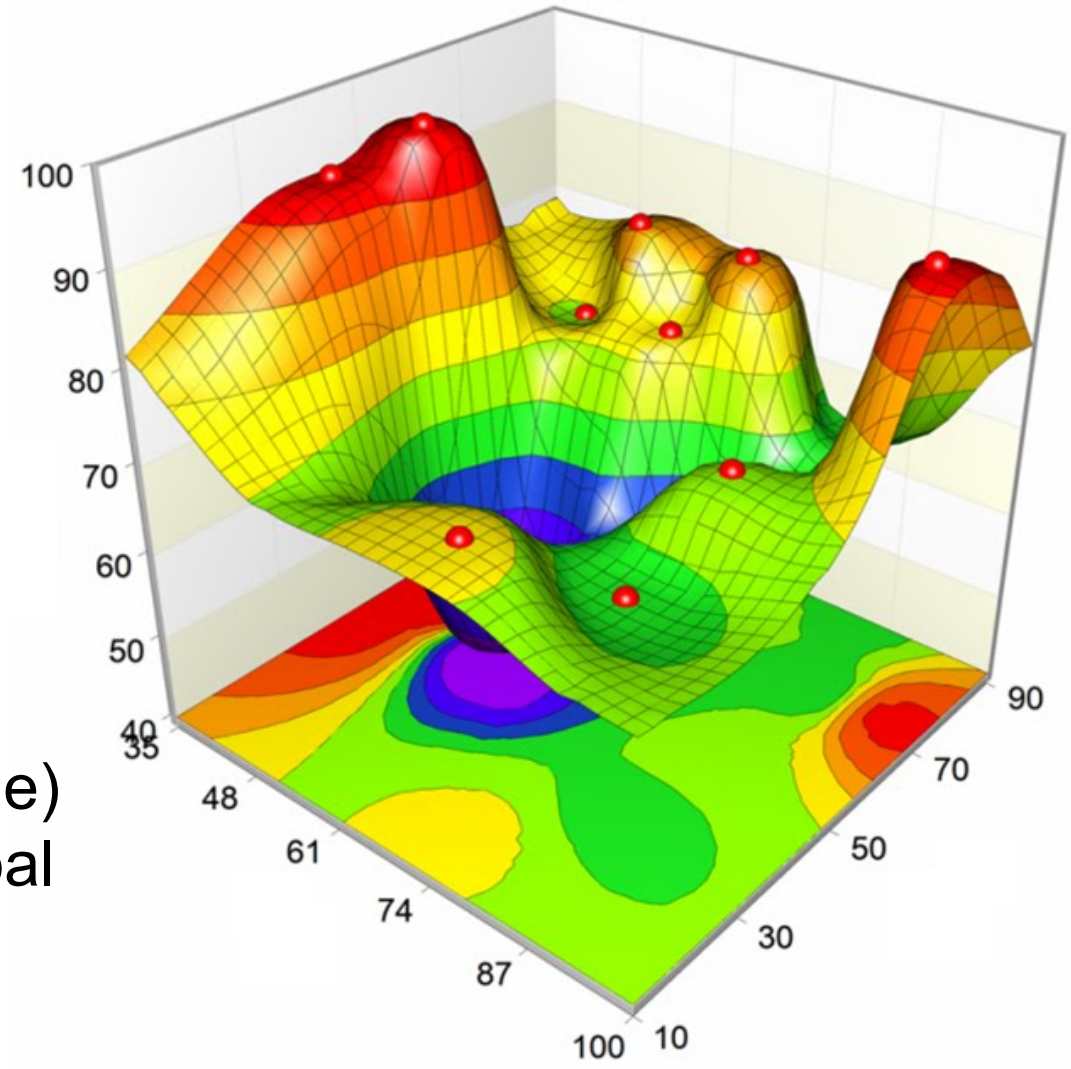
Training Neural Networks

Gradient Descent

To find argminx f(x):

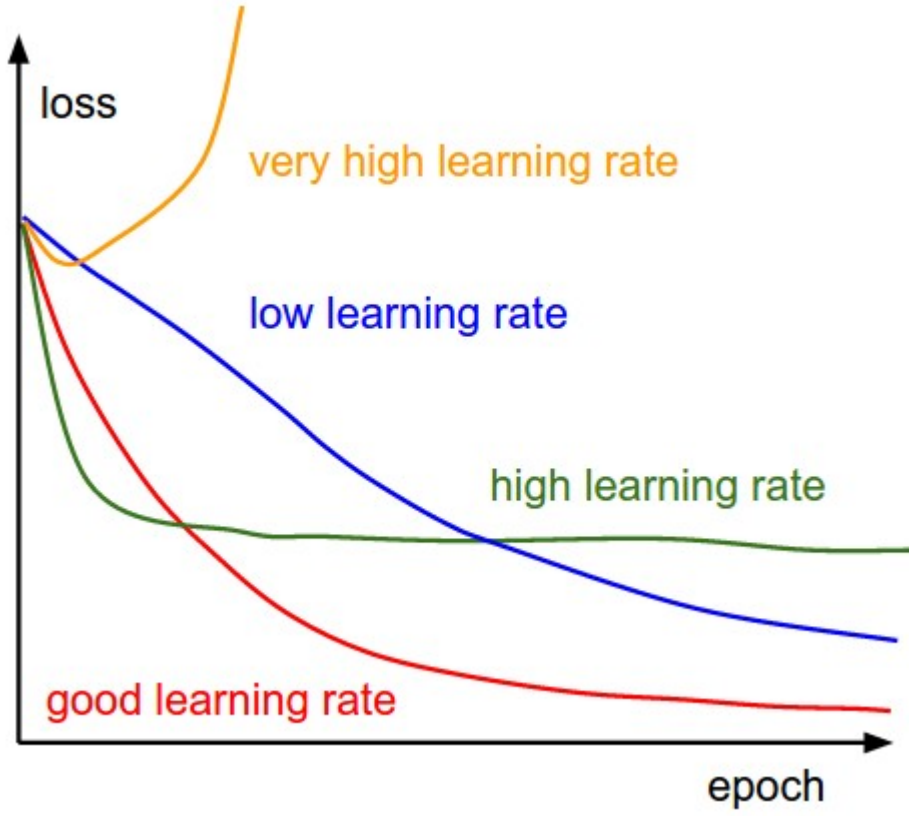
- *Start with x0*
- *For t=1….*
  - *xt+1 = xt + alpha f'(x t)*
    where alpha is smaller than 1

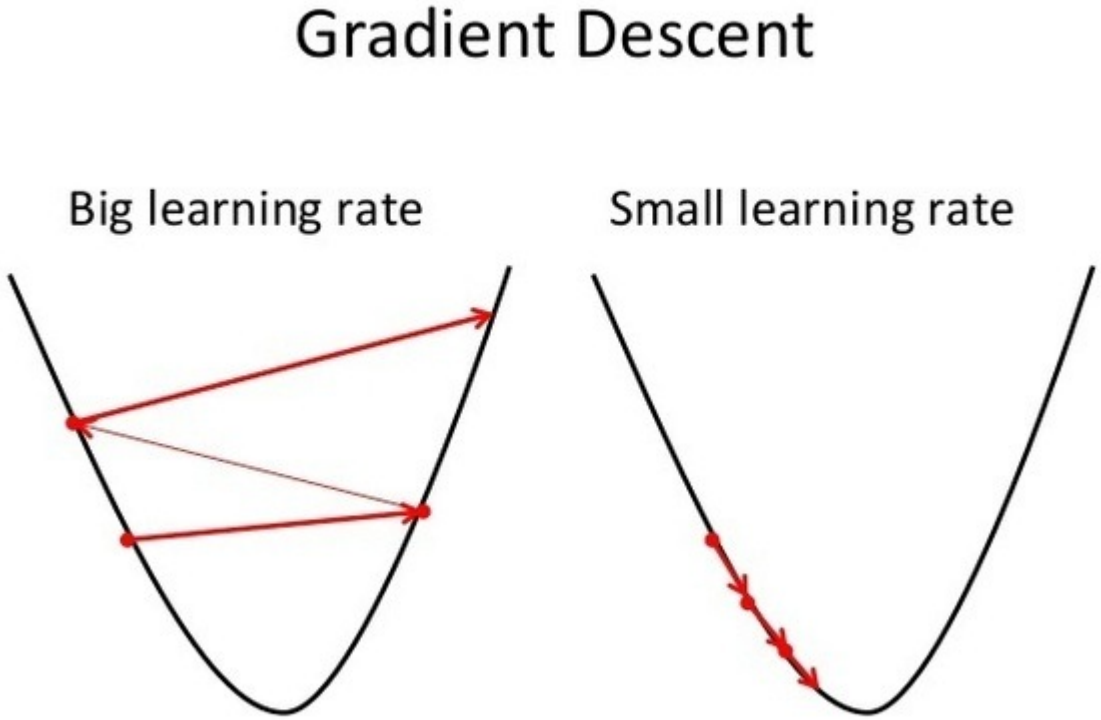
- Simple and often quite effective on ML tasks
- Often very scalable
- Only applies to smooth functions (differentiable)
- Might find a local minimum, rather than a global one



Training Neural Networks

LEARNING RATE

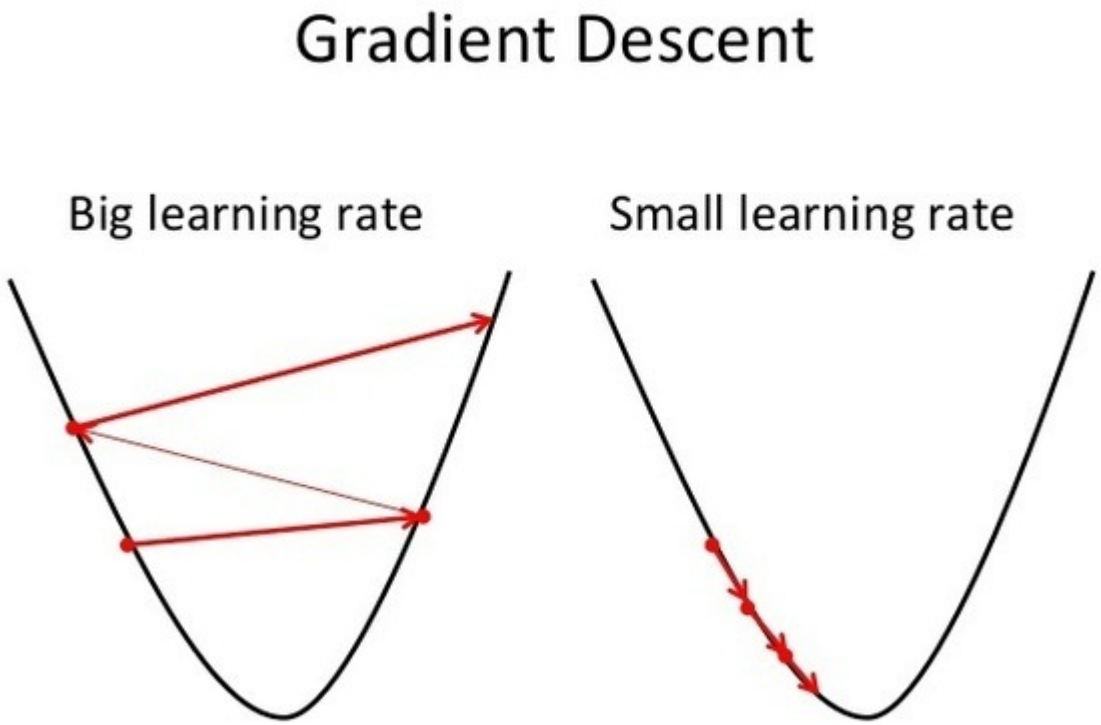


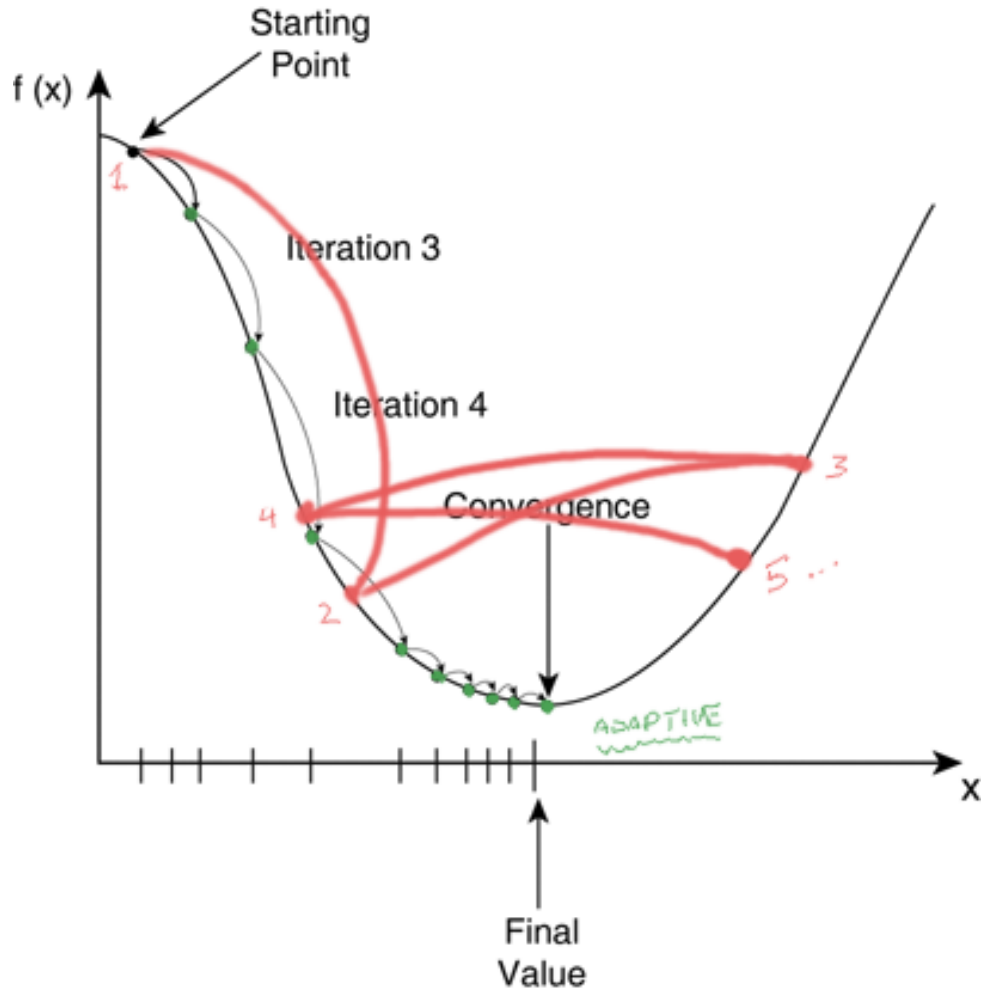


How to solve this problem?

Training Neural Networks

LEARNING RATE





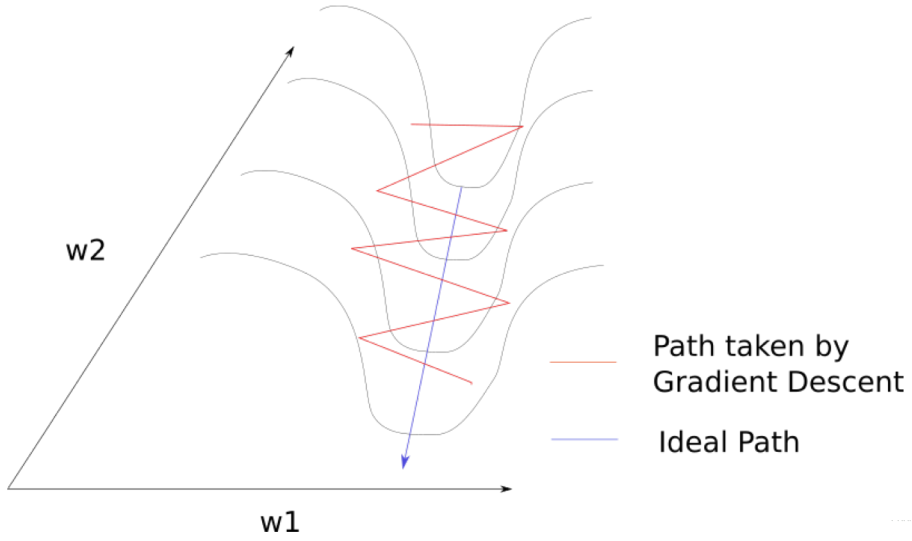
Training Neural Networks

Gradient Descent

Stochastic Gradient Descent

Batch Gradient Descent

Momentum / RMSProp / Adam etc



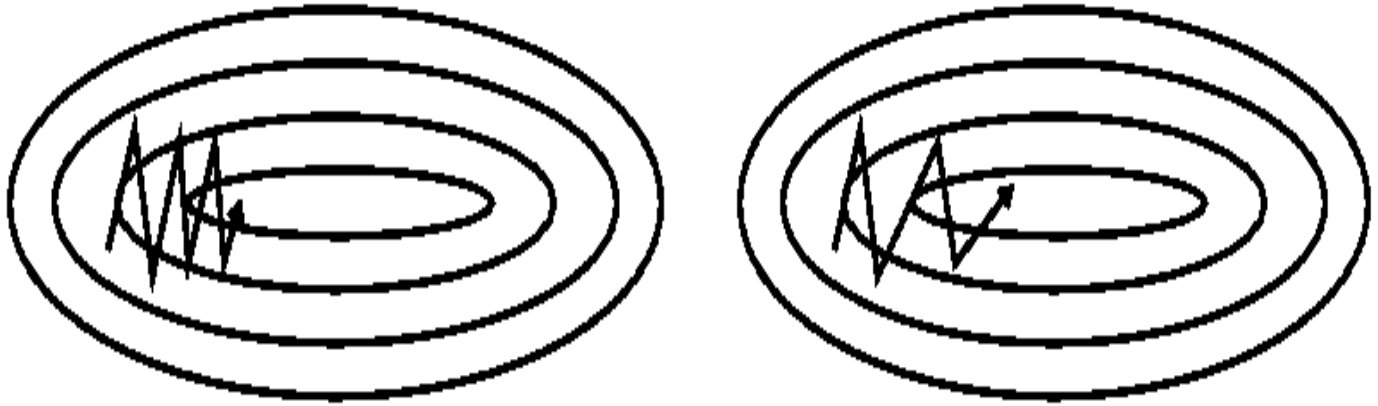
Training Neural Networks

Momentum

Instead of using only the gradient of the current step to guide the search, momentum also accumulates the gradient of the past steps to determine the direction to go:

$$v_{dw} = \beta \cdot v_{dw} + (1 - \beta) \cdot dw$$

$$W = W - \alpha \cdot v_{dw}$$



Training Neural Networks

RMSProp

RMSprop is developed by Prof. Geoffrey Hinton in his neural nets class.

Instead of letting all of the gradients accumulate for momentum, it only accumulates gradients in a fixed window.

$$v_{dw} = \beta \cdot v_{dw} + (1 - \beta) \cdot dw^2$$

$$W = W - \alpha \cdot \frac{dw}{\sqrt{v_{dw}} + \epsilon}$$

Training Neural Networks

Adam

Adam combines RMSProp with Momentum.

$$v_{dw}{}_{M} = \beta \cdot v_{dw} + (1 - \beta) \cdot dw$$

$$v_{dw}{}_{R} = \beta \cdot v_{dw} + (1 - \beta) \cdot dw^2$$

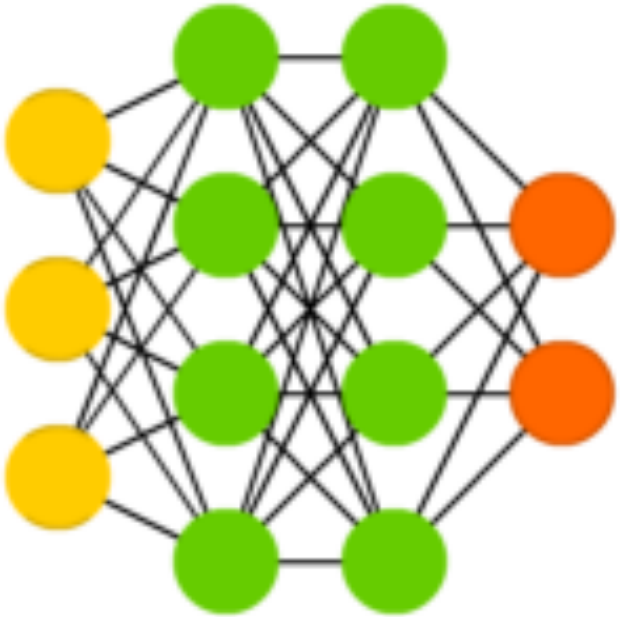
$$W = W - \alpha \cdot \frac{dw}{\sqrt{v_{dw}}_{R} + \epsilon} v_{dw}{}_{M}$$

Training Neural Architectures

FEED FORWARD

Feed Forward (FF)

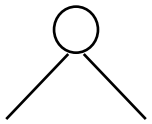
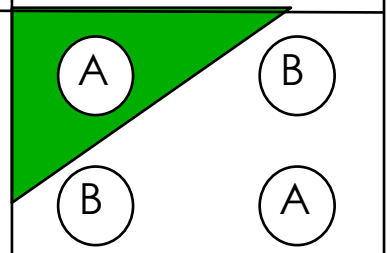
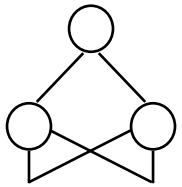
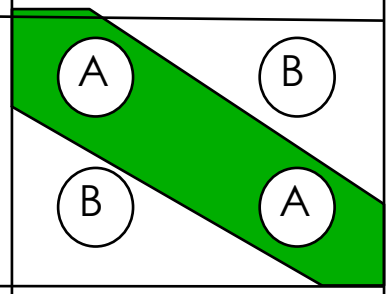
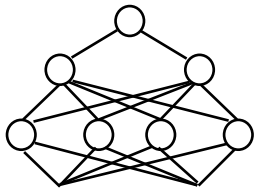
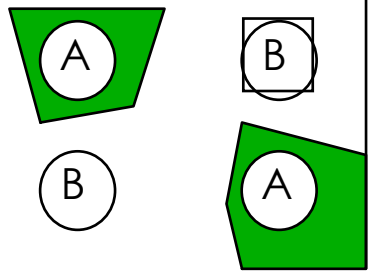
Deep Feed Forward (DFF)



Why we are going to deeper architectures?

Training Neural Architectures

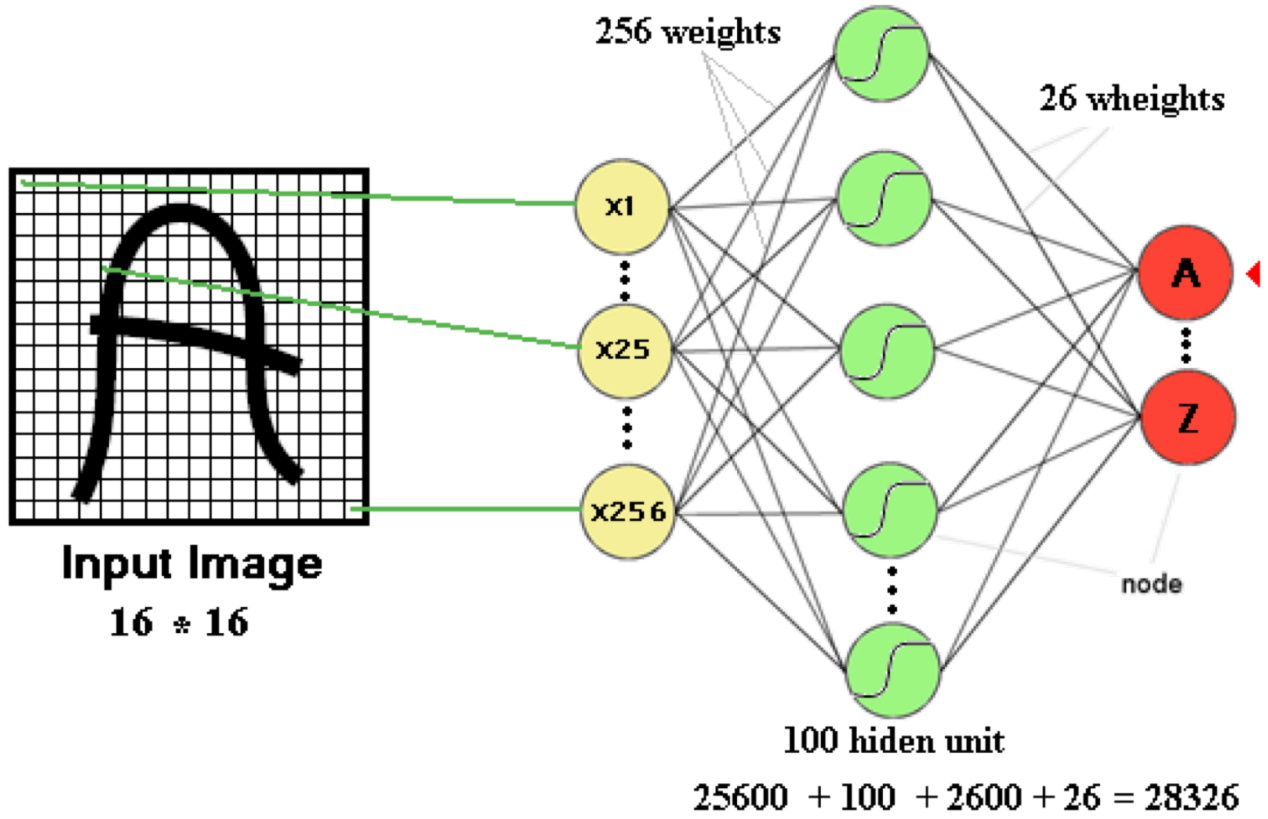
FEED FORWARD

Structure	*Types of Decision Regions*	*Exclusive-OR Problem*
Single-Layer 	*Half Plane Bounded By Hyper plane*	
Two-Layer 	*Convex Open Or Closed Regions*	
Three-Layer 	Arbitrary (Complexity Limited by No. of Nodes)	

Training Neural Architectures

FEED FORWARD

Ex: applying deep learning to image recognition

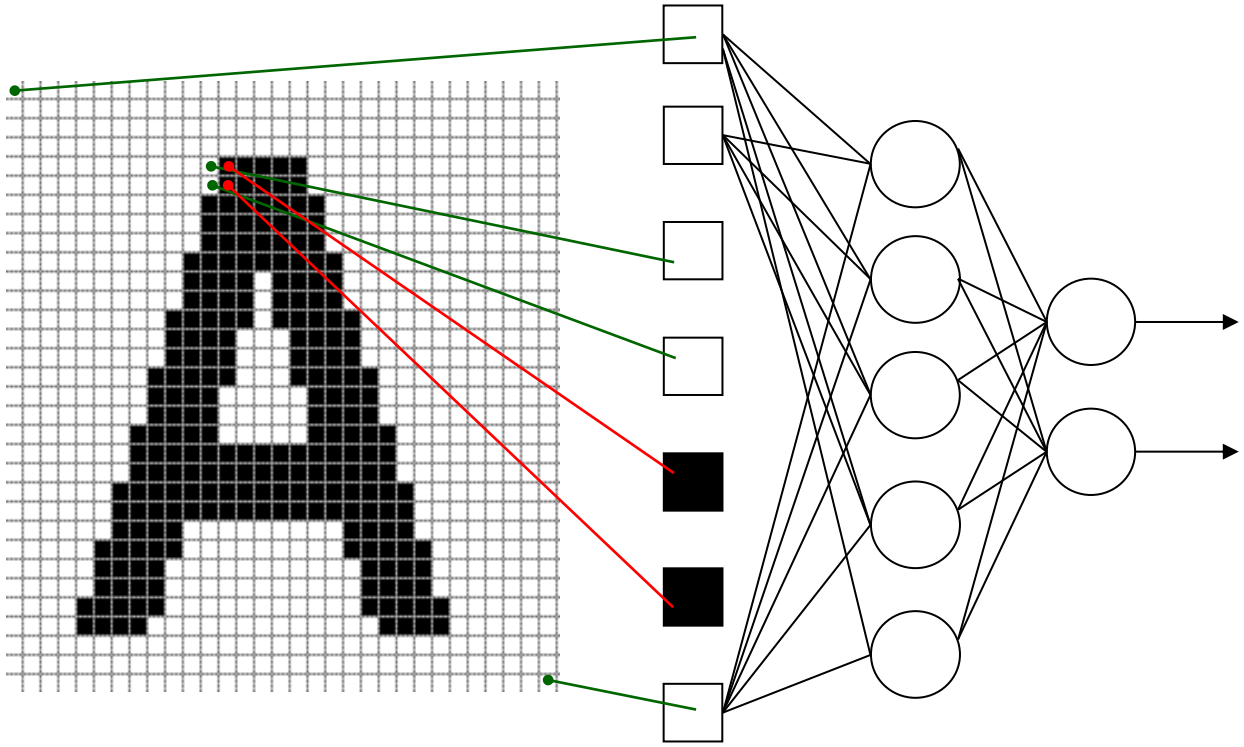


[Abin – Roozgard]

Training Neural Architectures

FEED FORWARD

Little or no invariance to shifting, scaling, and other forms of distortion

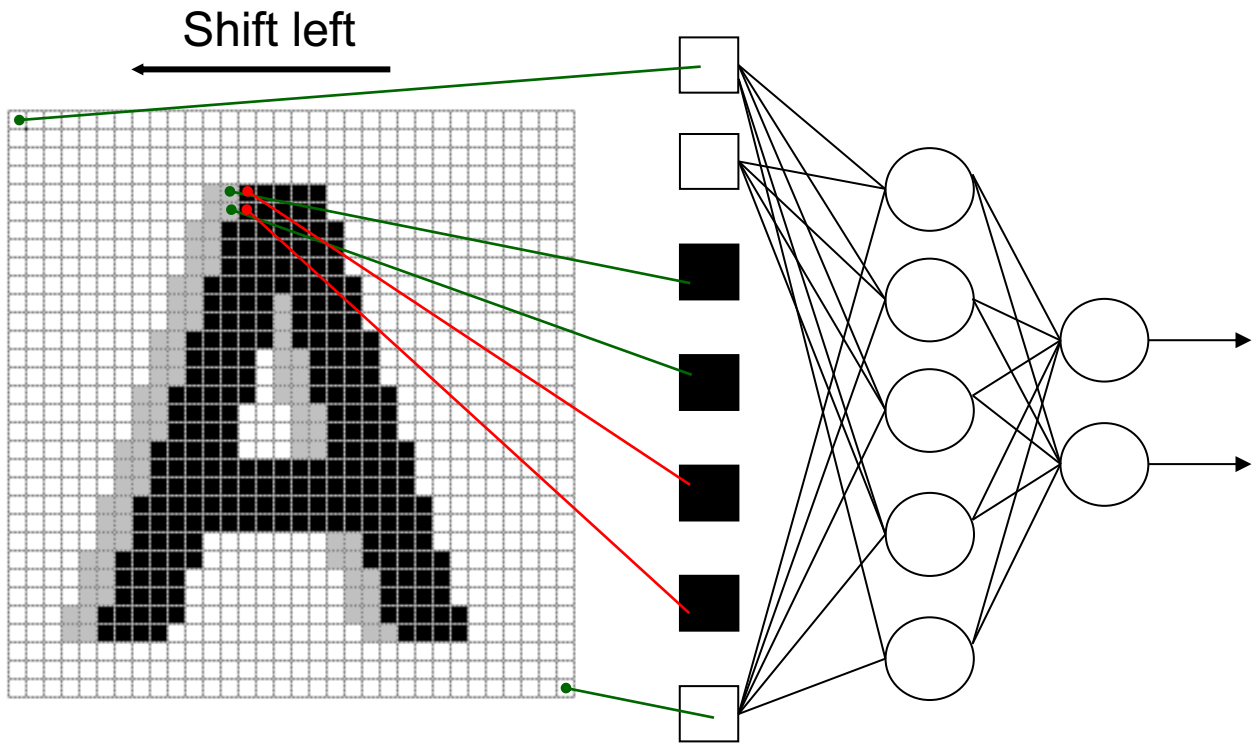


[Abin – Roozgard]

Training Neural Architectures

FEED FORWARD

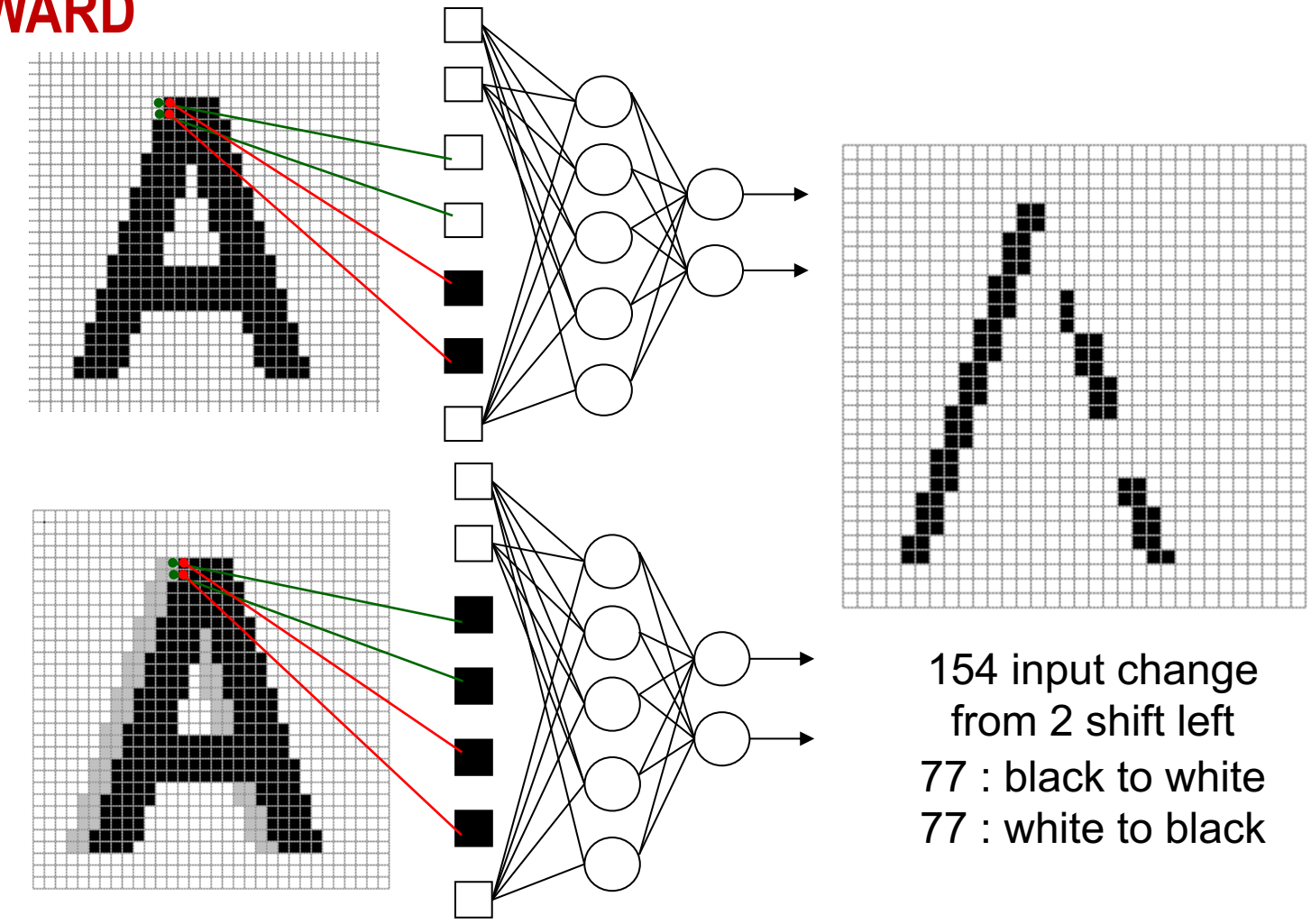
Little or no invariance to shifting, scaling, and other forms of distortion



[Abin – Roozgard]

Training Neural architectures

FEED FORWARD



154 input change
from 2 shift left
77 : black to white
77 : white to black

[Abin – Roozgard]

Training Neural Architectures

FEED FORWARD

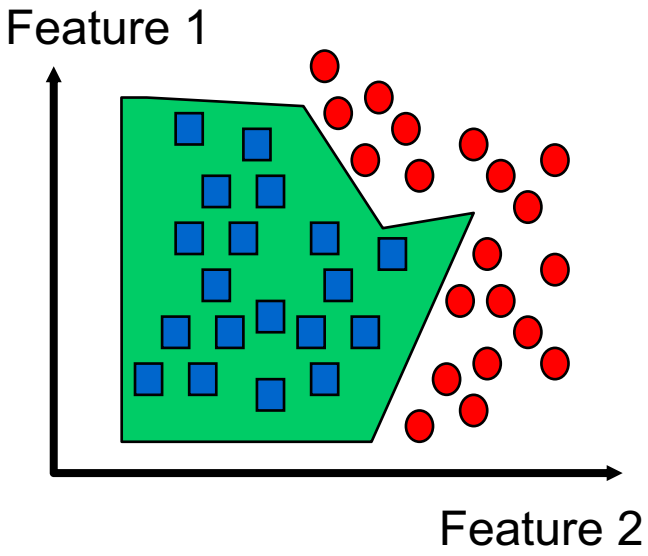
- scaling, and other forms of distortion

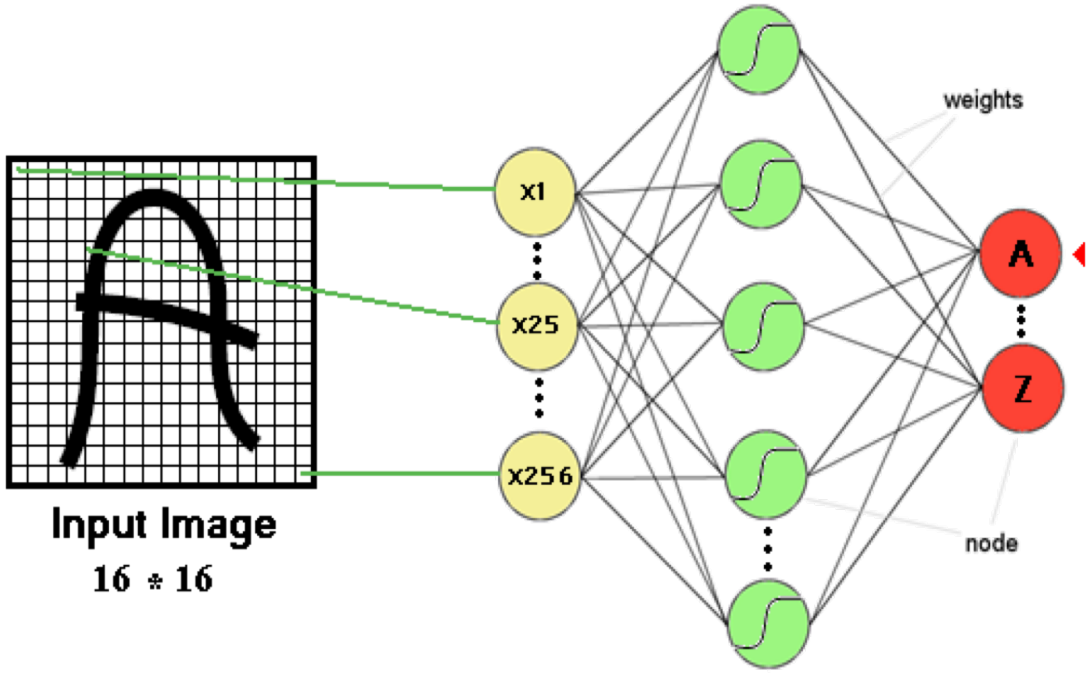
[Abin – Roozgard]

Training Neural Architectures

FEED FORWARD

- the topology of the input data is completely ignored
- work with raw data.





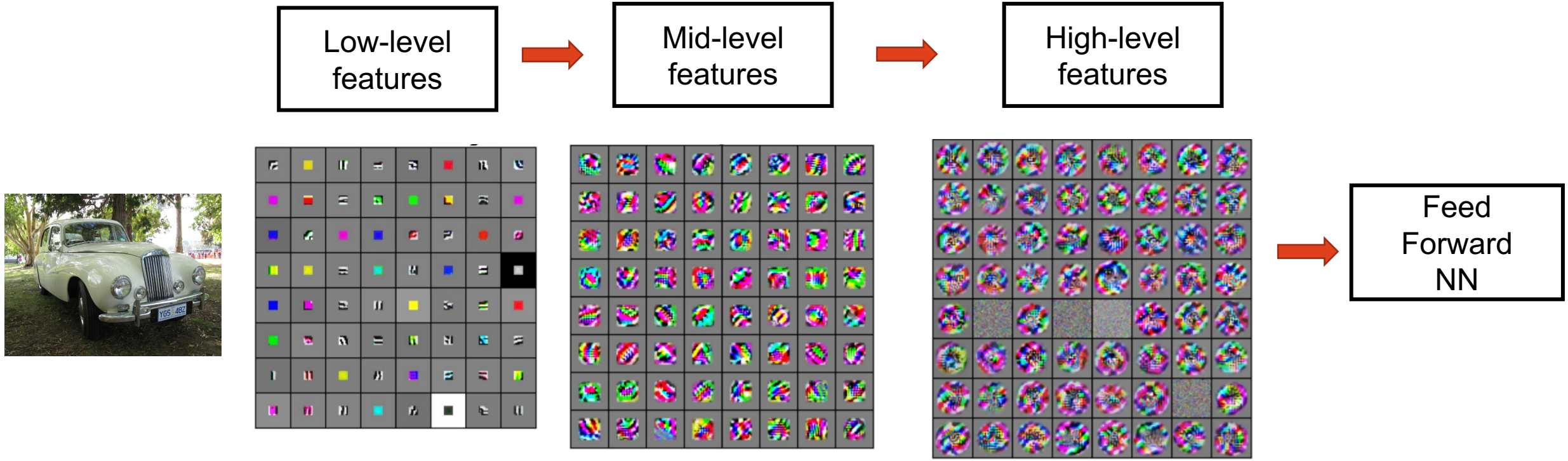
[Abin – Roozgard]

Convolutional Networks

GENERAL SCHEMA

Convolutional Networks

GENERAL SCHEMA

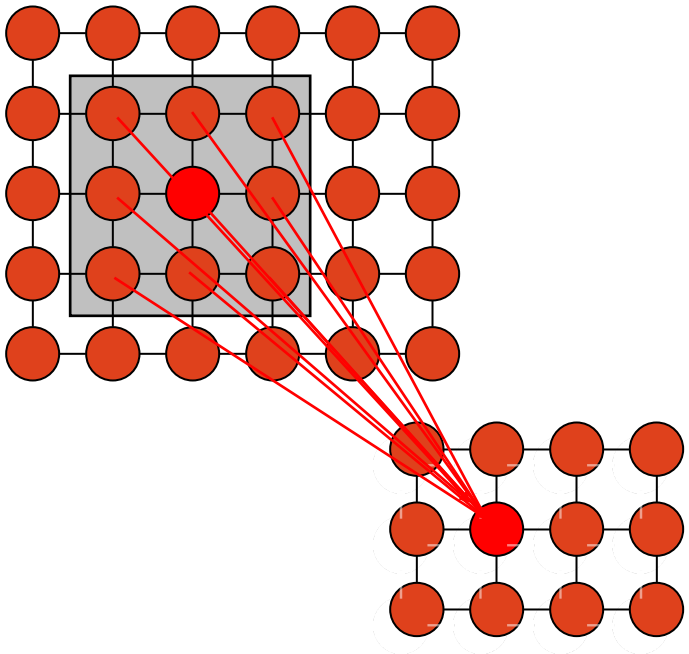


[image source: http://wiki.com]

[Zeiler and Fergus 2013]

Convolutional Networks

CONVOLUTION



10	5	3
4	6	1
1	1	8

Local image neighborhood

0	0	0
0	0.5	0
0	1	0.5

mask

	8	

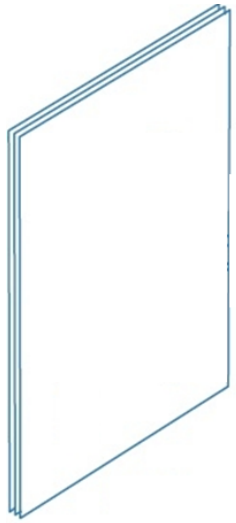
Modified image data

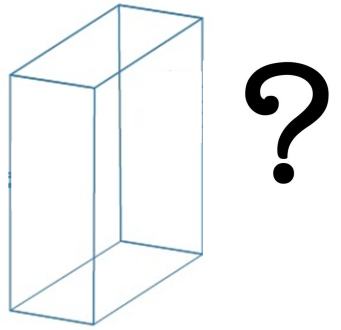
New Value = 3 + 1 + 4 = 8

Other parameters: Stride & Padding

Convolutional Networks

EXAMPLE

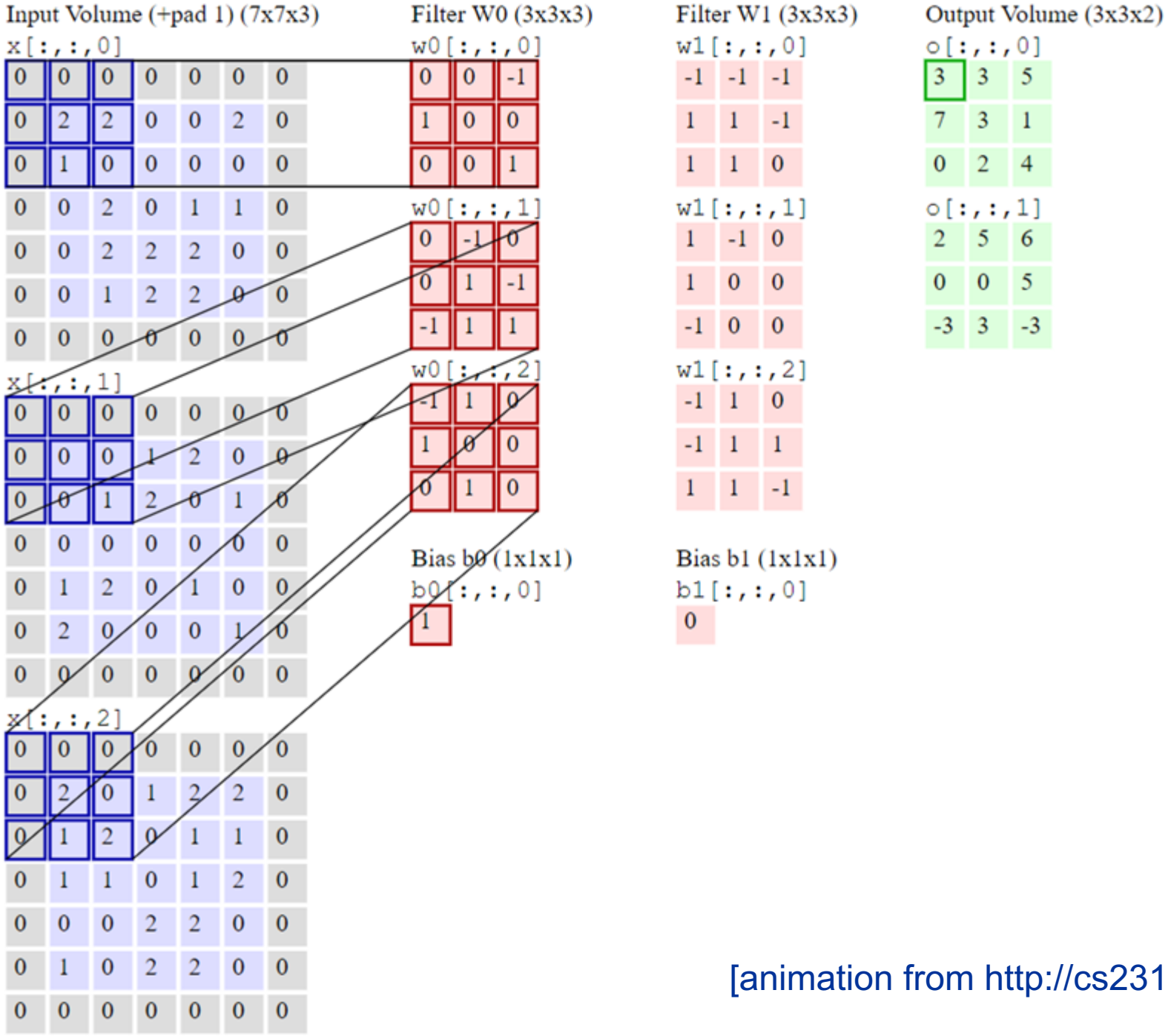




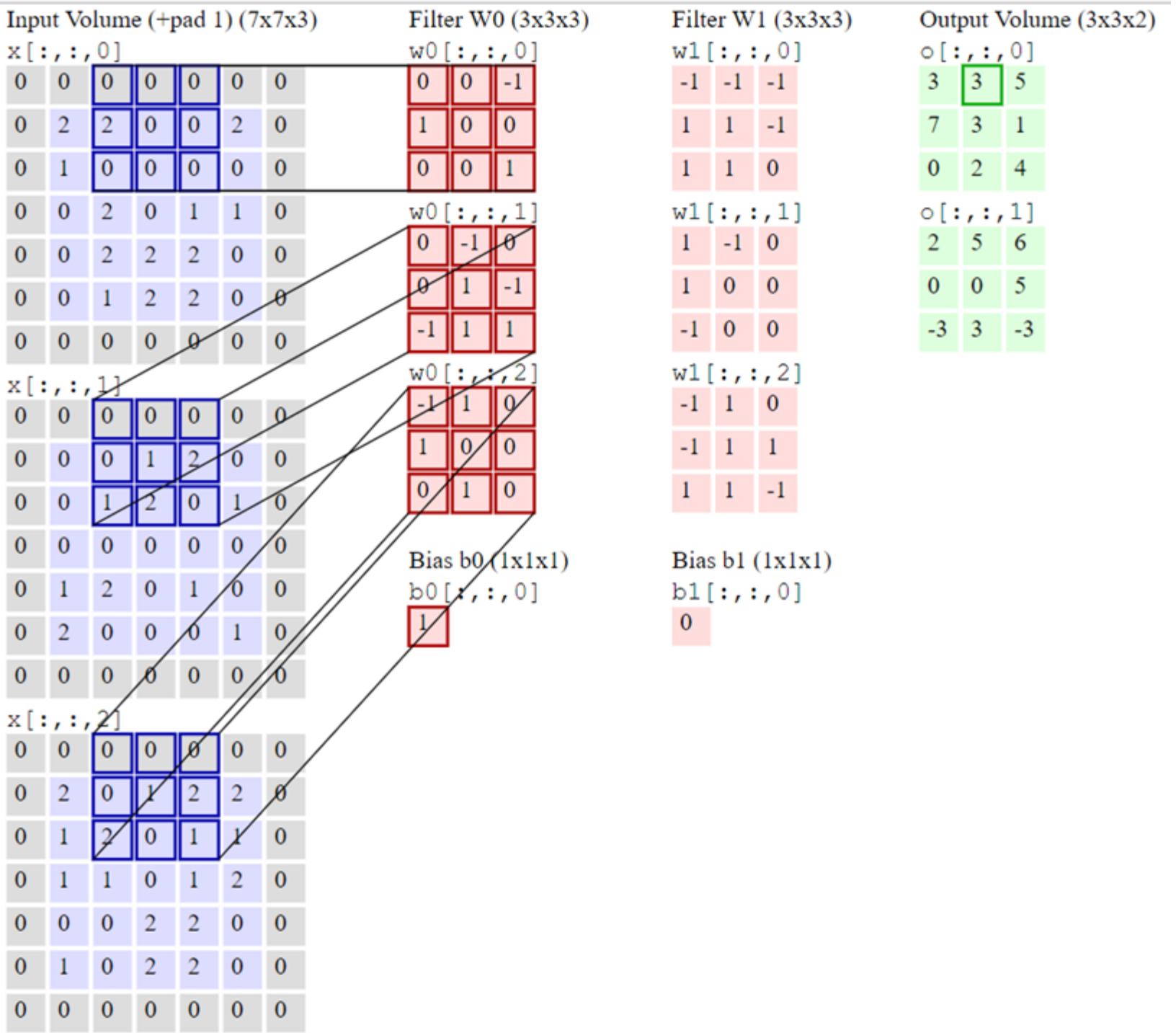
?

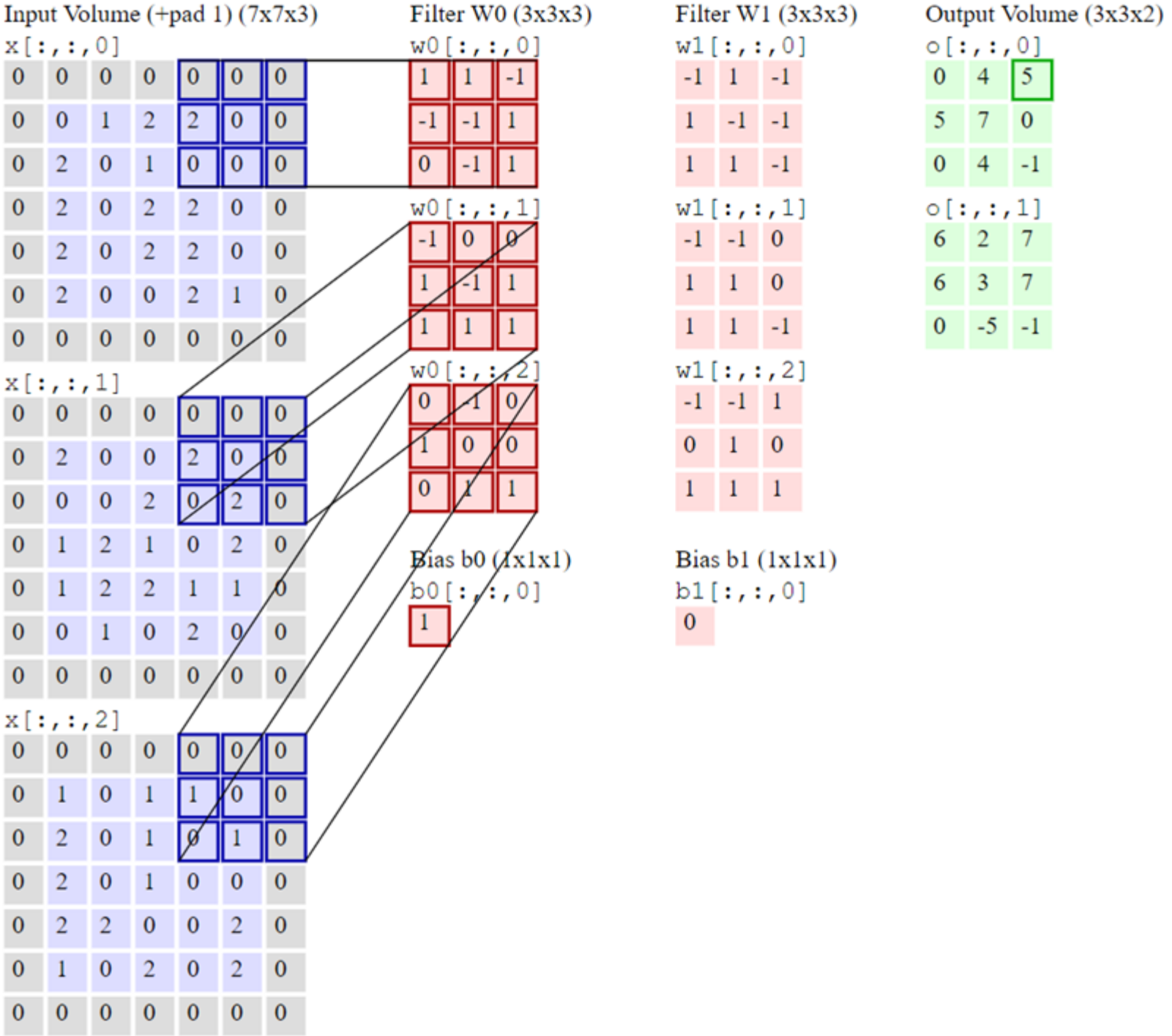
Image: **5x5x3**
Padding: 1
Stride: 2
Number of convolutional kernels: 2
Size of convolutional kernels (3x3)

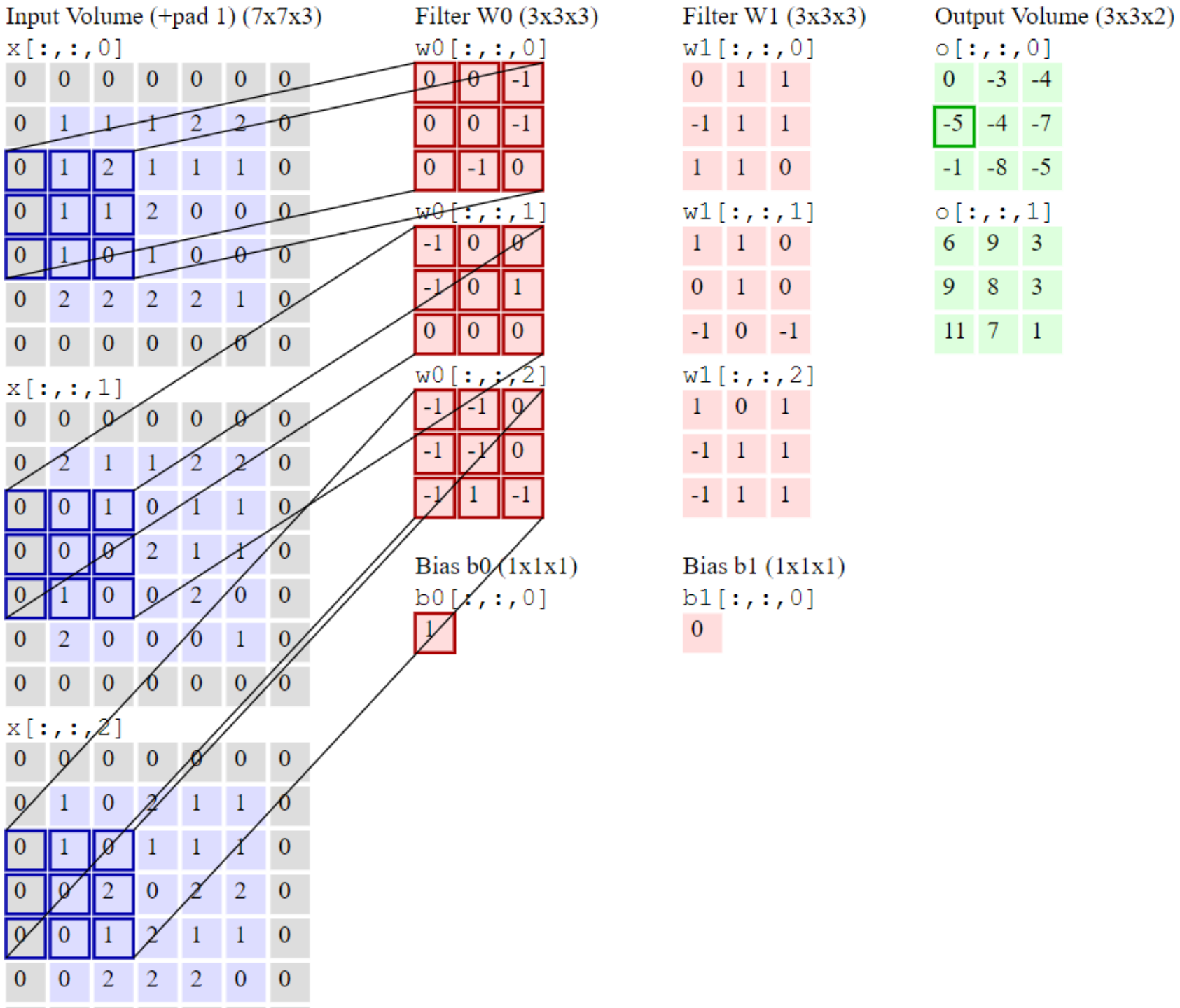
Image: **3x3x6**

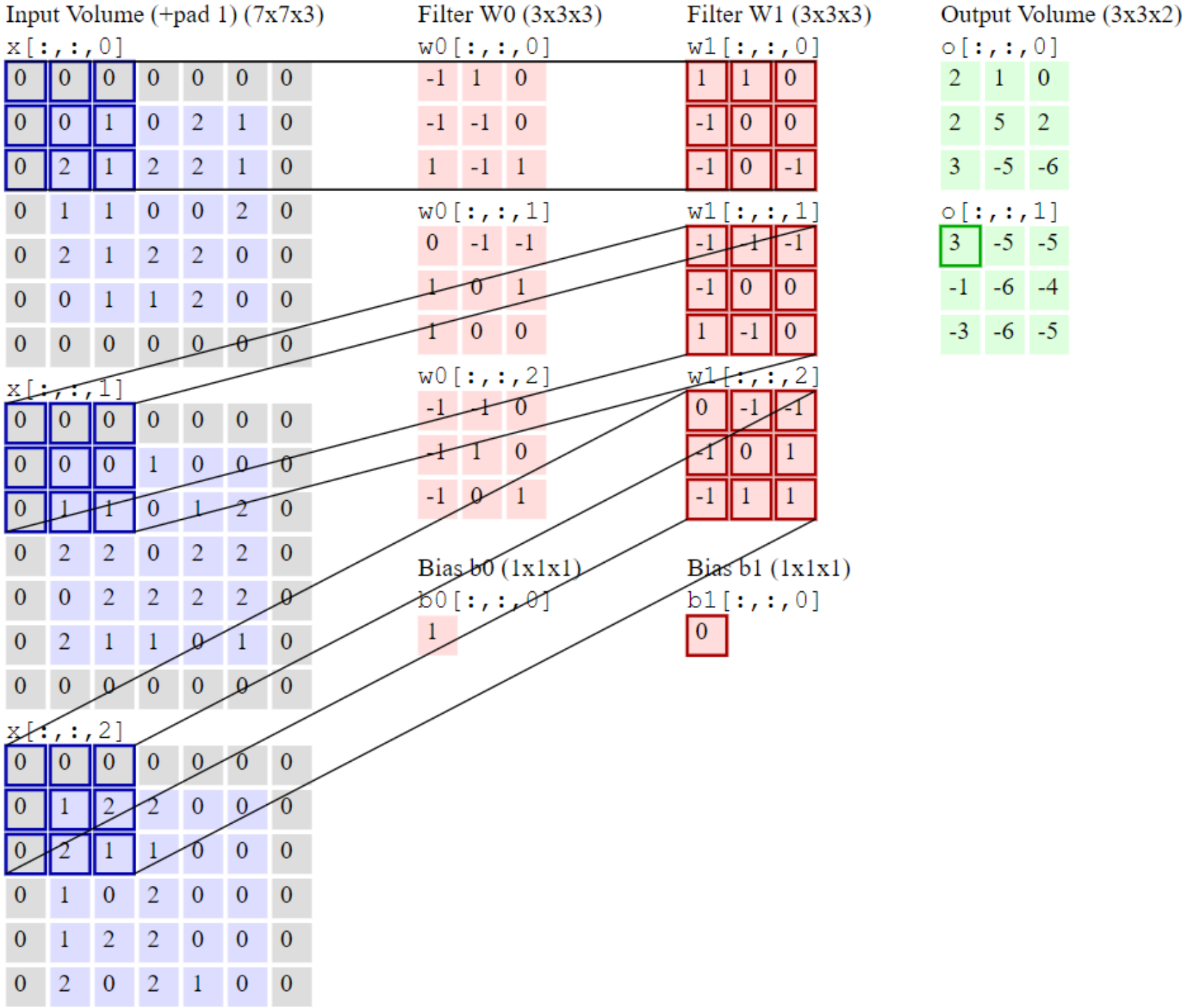


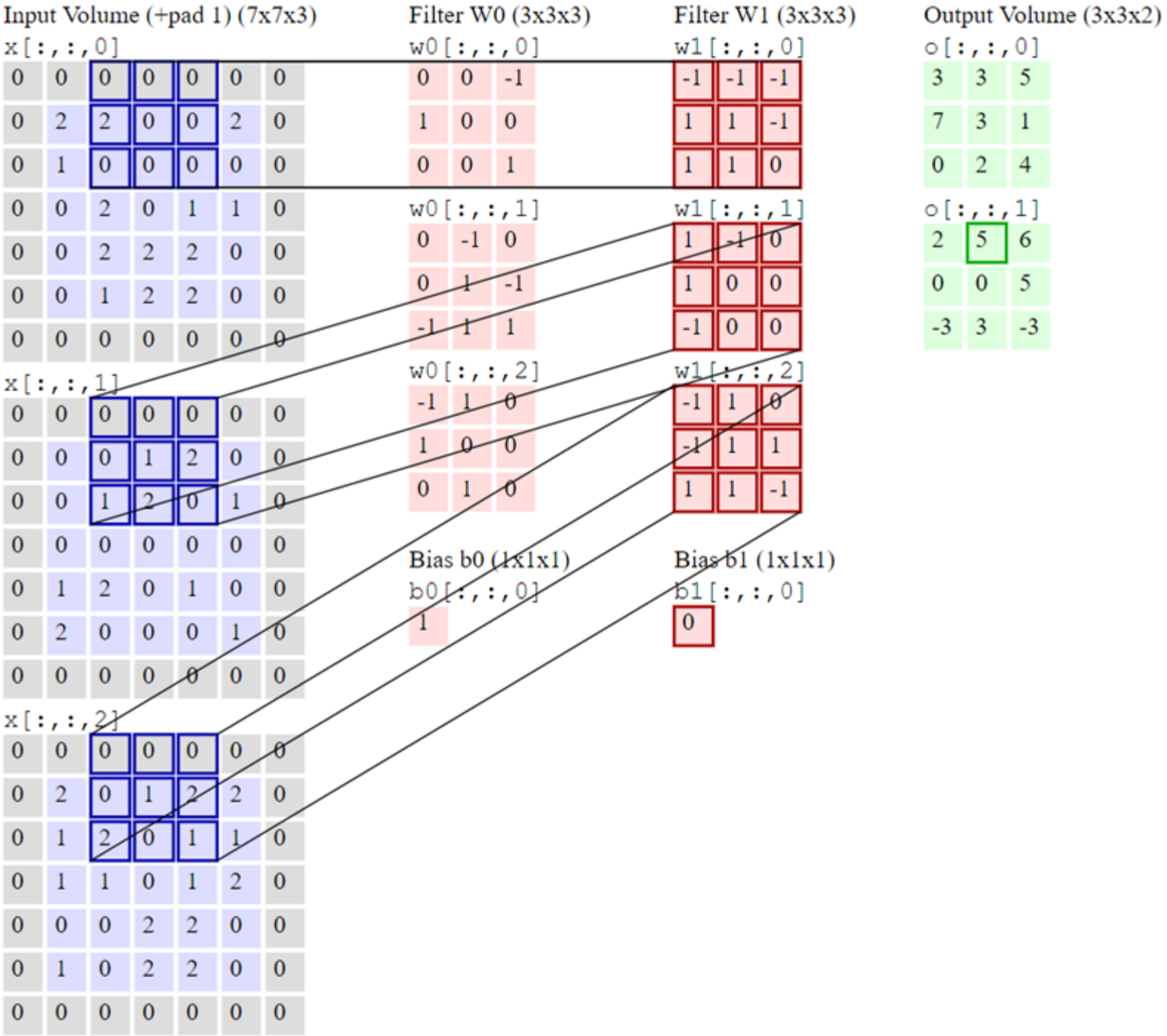
[animation from http://cs231n.github.io/convolutional-networks]

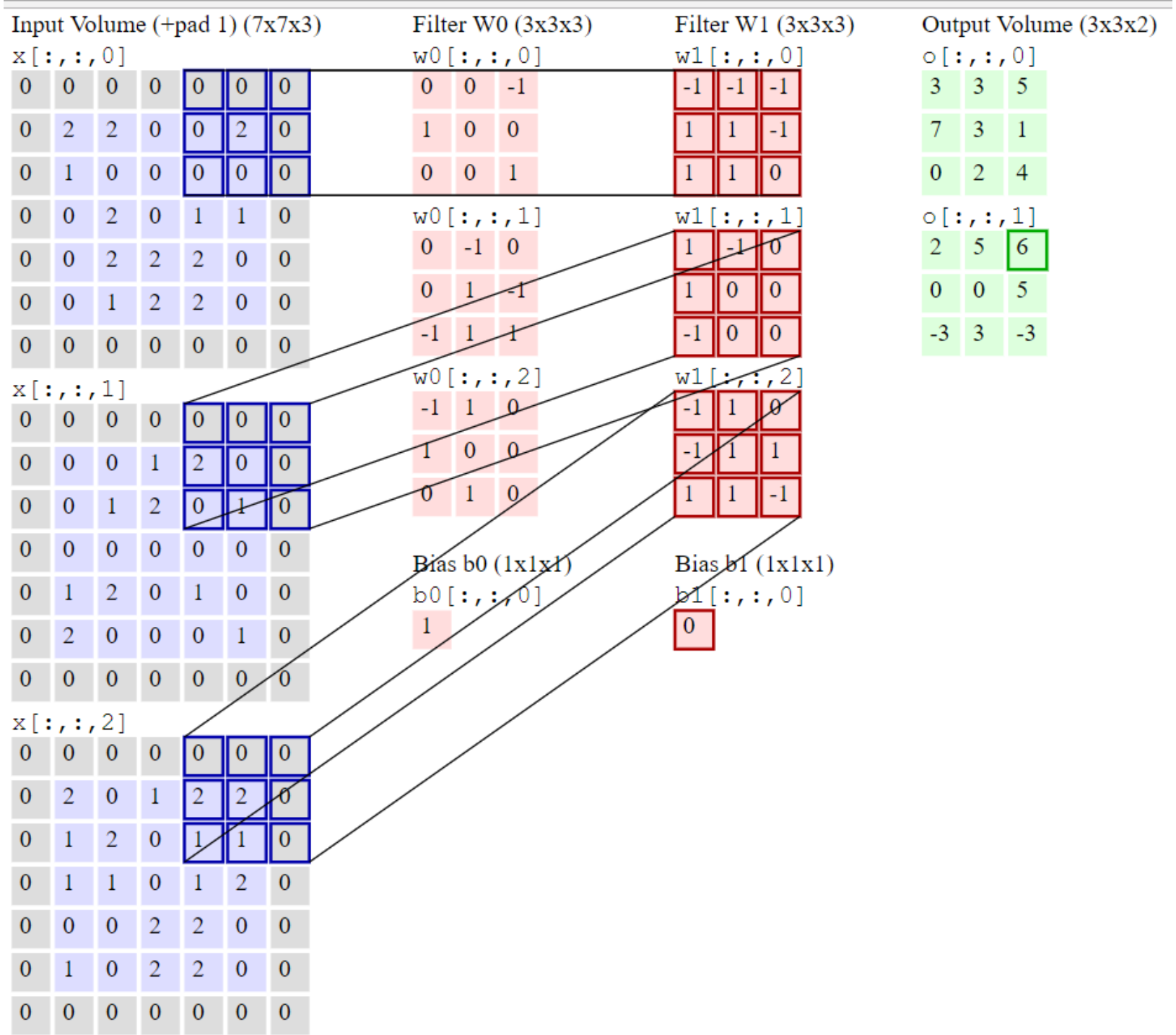

Input Volume (+pad 1) (7x7x3)
x[:,:,0]
0 0 0 0 0 0 0
0 2 2 0 0 2 0
0 1 0 0 0 0 0
0 0 2 0 1 1 0
0 0 2 2 2 0 0
0 0 1 2 2 0 0
0 0 0 0 0 0 0
x[:,:,1]
0 0 0 0 0 0 0
0 0 0 1 2 0 0
0 0 1 2 0 1 0
0 0 0 0 0 0 0
0 1 2 0 1 0 0
0 2 0 0 0 1 0
0 0 0 0 0 0 0
x[:,:,2]
0 0 0 0 0 0 0
0 2 0 1 2 2 0
0 1 2 0 1 1 0
0 1 1 0 1 2 0
0 0 0 2 2 0 0
0 1 0 2 2 0 0
0 0 0 0 0 0 0
Filter W0 (3x3x3)
w0[:,:,0]
0 0 -1
1 0 0
0 0 1
w0[:,:,1]
0 -1 0
0 1 -1
-1 1 1
w0[:,:,2]
-1 1 0
1 0 0
0 1 0
Bias b0 (1x1x1)
b0[:,:,0]
1
Filter W1 (3x3x3)
w1[:,:,0]
-1 -1 -1
1 1 -1
1 1 0
w1[:,:,1]
1 -1 0
1 0 0
-1 0 0
w1[:,:,2]
-1 1 0
-1 1 1
1 1 -1
Bias b1 (1x1x1)
b1[:,:,0]
0
Output Volume (3x3x2)
o[:,:,0]
3 3 5
7 3 1
0 2 4
o[:,:,1]
2 5 6
0 0 5
-3 3 -3

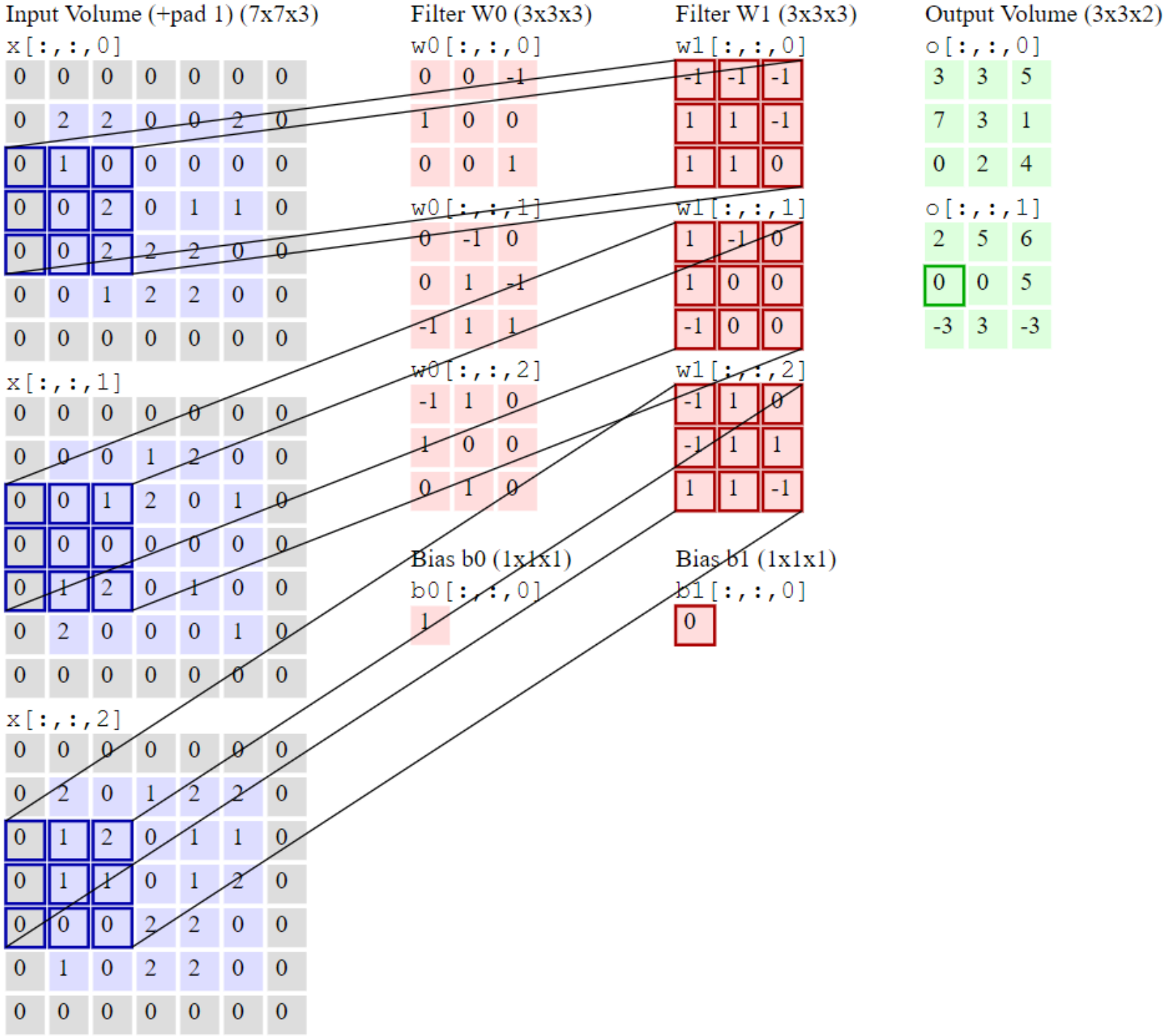

Input Volume (+pad 1) (7x7x3)
x[:,:,0]
0 0 0 0 0 0 0
0 0 1 2 2 0 0
0 2 0 1 0 0 0
0 2 0 2 2 0 0
0 2 0 2 2 0 0
0 2 0 0 2 1 0
0 0 0 0 0 0 0
x[:,:,1]
0 0 0 0 0 0 0
0 2 0 0 2 0 0
0 0 0 2 0 2 0
0 1 2 1 0 2 0
0 1 2 2 1 1 0
0 0 1 0 2 0 0
0 0 0 0 0 0 0
x[:,:,2]
0 0 0 0 0 0 0
0 1 0 1 1 0 0
0 2 0 1 0 1 0
0 2 0 1 0 0 0
0 2 2 0 0 2 0
0 1 0 2 0 2 0
0 0 0 0 0 0 0
Filter W0 (3x3x3)
w0[:,:,0]
1 1 -1
-1 -1 1
0 -1 1
w0[:,:,1]
-1 0 0
1 -1 1
1 1 1
w0[:,:,2]
0 -1 0
1 0 0
0 1 1
Bias b0 (1x1x1)
b0[:,:,0]
1
Filter W1 (3x3x3)
w1[:,:,0]
-1 1 -1
1 -1 -1
1 1 -1
w1[:,:,1]
-1 -1 0
1 1 0
1 1 -1
w1[:,:,2]
-1 -1 1
0 1 0
1 1 1
Bias b1 (1x1x1)
b1[:,:,0]
0
Output Volume (3x3x2)
o[:,:,0]
0 4 5
5 7 0
0 4 -1
o[:,:,1]
6 2 7
6 3 7
0 -5 -1

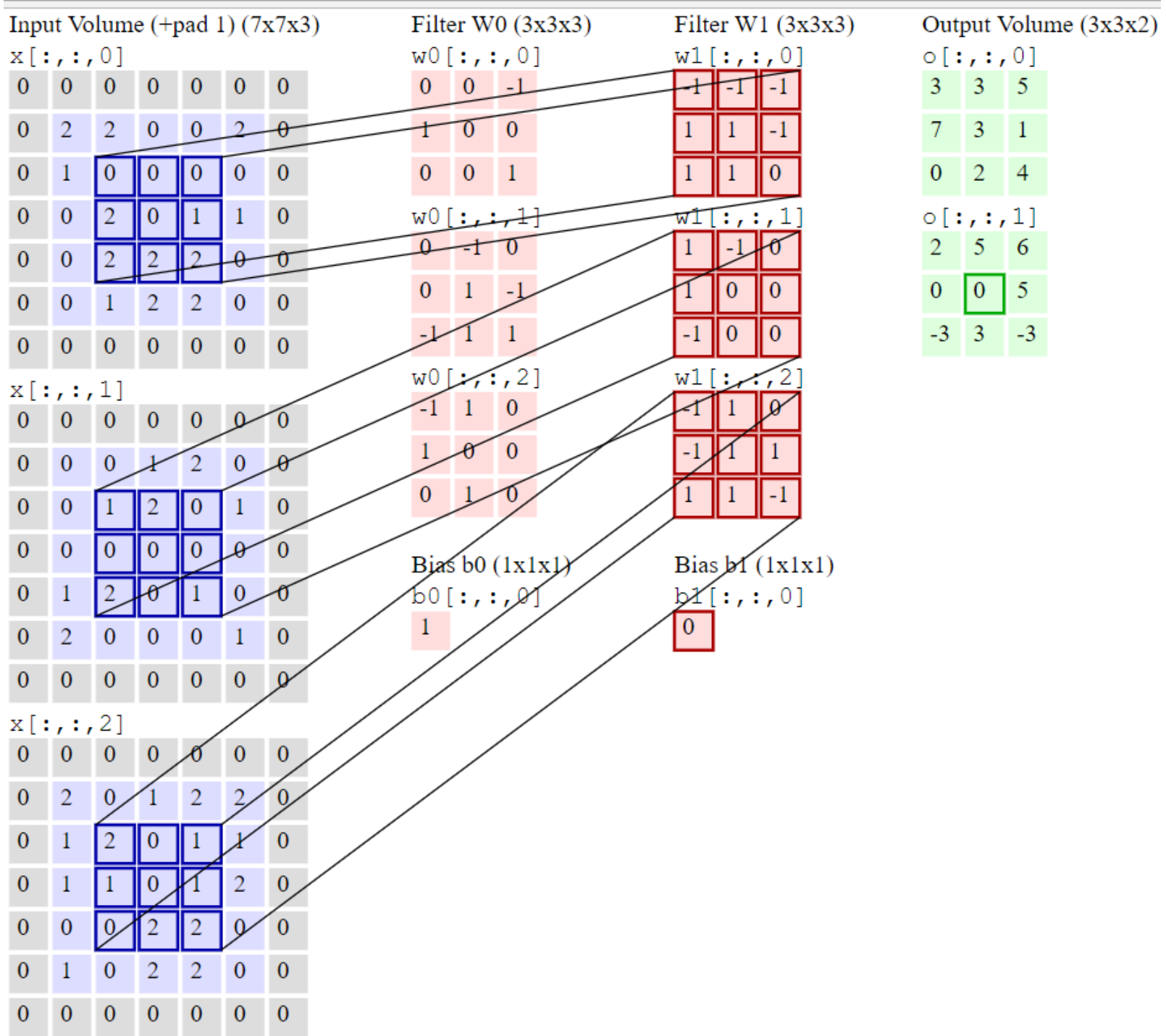

Input Volume (+pad 1) (7x7x3)
x[:,:,0]
0 0 0 0 0 0 0
0 1 1 1 2 2 0
0 1 2 1 1 1 0
0 1 1 2 0 0 0
0 1 0 1 0 0 0
0 2 2 2 2 1 0
0 0 0 0 0 0 0
x[:,:,1]
0 0 0 0 0 0 0
0 2 1 1 2 2 0
0 0 1 0 1 1 0
0 0 0 2 1 1 0
0 1 0 0 2 0 0
0 2 0 0 0 1 0
0 0 0 0 0 0 0
x[:,:,2]
0 0 0 0 0 0 0
0 1 0 2 1 1 0
0 1 0 1 1 1 0
0 0 2 0 2 2 0
0 0 1 2 1 1 0
0 0 2 2 2 0 0
Filter W0 (3x3x3)
w0[:,:,0]
0 0 -1
0 0 -1
0 -1 0
w0[:,:,1]
-1 0 0
-1 0 1
0 0 0
w0[:,:,2]
-1 -1 0
-1 -1 0
-1 1 -1
Bias b0 (1x1x1)
b0[:,:,0]
1
Filter W1 (3x3x3)
w1[:,:,0]
0 1 1
-1 1 1
1 1 0
w1[:,:,1]
1 1 0
0 1 0
-1 0 -1
w1[:,:,2]
1 0 1
-1 1 1
-1 1 1
Bias b1 (1x1x1)
b1[:,:,0]
0
Output Volume (3x3x2)
o[:,:,0]
0 -3 -4
-5 -4 -7
-1 -8 -5
o[:,:,1]
6 9 3
9 8 3
11 7 1


Input Volume (+pad 1) (7x7x3)
x[:,:,0]
0 0 0 0 0 0 0
0 0 1 0 2 1 0
0 2 1 2 2 1 0
0 1 1 0 0 2 0
0 2 1 2 2 0 0
0 0 1 1 2 0 0
0 0 0 0 0 0 0
x[:,:,1]
0 0 0 0 0 0 0
0 0 0 1 0 0 0
0 1 1 0 1 2 0
0 2 2 0 2 2 0
0 0 2 2 2 2 0
0 2 1 1 0 1 0
0 0 0 0 0 0 0
x[:,:,2]
0 0 0 0 0 0 0
0 1 2 2 0 0 0
0 2 1 1 0 0 0
0 1 0 2 0 0 0
0 1 2 2 0 0 0
0 2 0 2 1 0 0
Filter W0 (3x3x3)
w0[:,:,0]
-1 1 0
-1 -1 0
1 -1 1
w0[:,:,1]
0 -1 -1
1 0 1
1 0 0
w0[:,:,2]
-1 -1 0
-1 1 0
-1 0 1
Filter W1 (3x3x3)
w1[:,:,0]
1 1 0
-1 0 0
-1 0 -1
w1[:,:,1]
-1 1 -1
-1 0 0
1 -1 0
w1[:,:,2]
0 -1 -1
-1 0 1
-1 1 1
Bias b0 (1x1x1)
b0[:,:,0]
1
Bias b1 (1x1x1)
b1[:,:,0]
0
Output Volume (3x3x2)
o[:,:,0]
2 1 0
2 5 2
3 -5 -6
o[:,:,1]
3 -5 -5
-1 -6 -4
-3 -6 -5


Input Volume (+pad 1) (7x7x3)
x[:,:,0]
0 0 0 0 0 0 0
0 2 2 0 0 2 0
0 1 0 0 0 0 0
0 0 2 0 1 1 0
0 0 2 2 2 0 0
0 0 1 2 2 0 0
0 0 0 0 0 0 0
x[:,:,1]
0 0 0 0 0 0 0
0 0 0 1 2 0 0
0 0 1 2 0 1 0
0 0 0 0 0 0 0
0 1 2 0 1 0 0
0 2 0 0 0 1 0
0 0 0 0 0 0 0
x[:,:,2]
0 0 0 0 0 0 0
0 2 0 1 2 2 0
0 1 2 0 1 1 0
0 1 1 0 1 2 0
0 0 0 2 2 0 0
0 1 0 2 2 0 0
0 0 0 0 0 0 0
Filter W0 (3x3x3)
w0[:,:,0]
0 0 -1
1 0 0
0 0 1
w0[:,:,1]
0 -1 0
0 1 -1
-1 1 1
w0[:,:,2]
-1 1 0
1 0 0
0 1 0
Bias b0 (1x1x1)
b0[:,:,0]
1
Filter W1 (3x3x3)
w1[:,:,0]
-1 -1 -1
1 1 -1
1 1 0
w1[:,:,1]
1 -1 0
1 0 0
-1 0 0
w1[:,:,2]
-1 1 0
-1 1 1
1 1 -1
Bias b1 (1x1x1)
b1[:,:,0]
0
Output Volume (3x3x2)
o[:,:,0]
3 3 5
7 3 1
0 2 4
o[:,:,1]
2 5 6
0 0 5
-3 3 -3

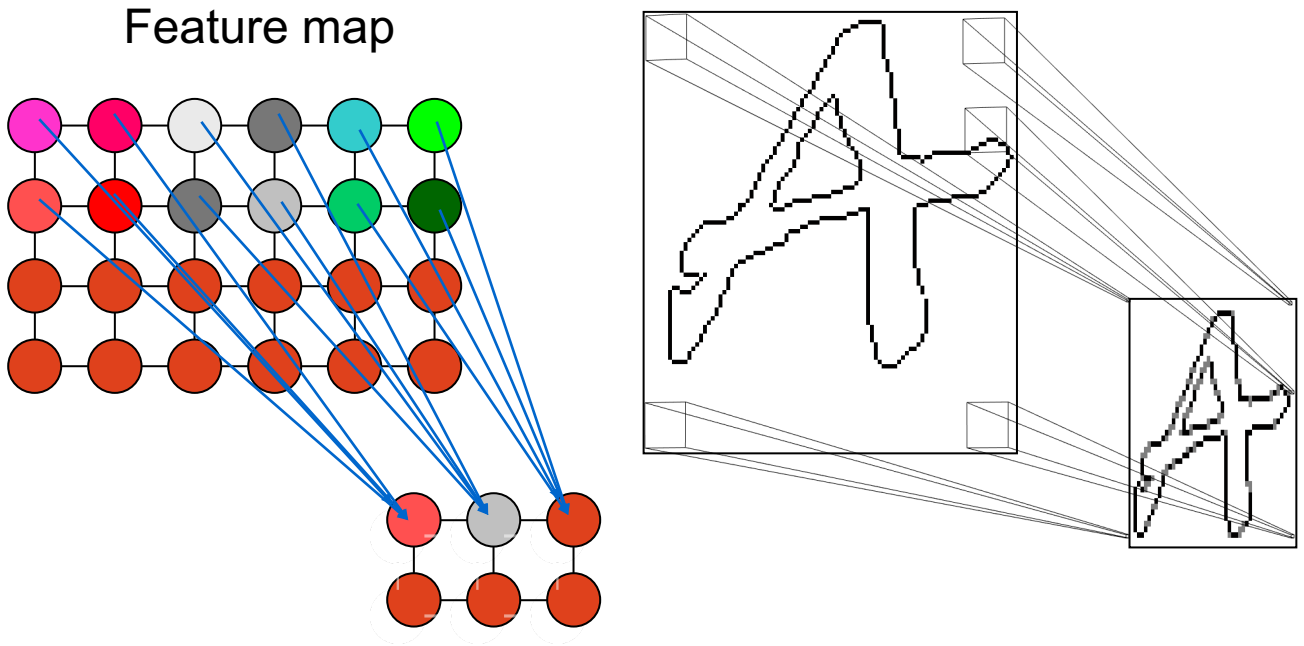

Input Volume (+pad 1) (7x7x3)
x[:,:,0]
0 0 0 0 0 0 0
0 2 2 0 0 2 0
0 1 0 0 0 0 0
0 0 2 0 1 1 0
0 0 2 2 2 0 0
0 0 1 2 2 0 0
0 0 0 0 0 0 0
x[:,:,1]
0 0 0 0 0 0 0
0 0 0 1 2 0 0
0 0 1 2 0 1 0
0 0 0 0 0 0 0
0 1 2 0 1 0 0
0 2 0 0 0 1 0
0 0 0 0 0 0 0
x[:,:,2]
0 0 0 0 0 0 0
0 2 0 1 2 2 0
0 1 2 0 1 1 0
0 1 1 0 1 2 0
0 0 0 2 2 0 0
0 1 0 2 2 0 0
0 0 0 0 0 0 0
Filter W0 (3x3x3)
w0[:,:,0]
0 0 -1
1 0 0
0 0 1
w0[:,:,1]
0 -1 0
0 1 -1
-1 1 1
w0[:,:,2]
-1 1 0
1 0 0
0 1 0
Filter W1 (3x3x3)
w1[:,:,0]
-1 -1 -1
1 1 -1
1 1 0
w1[:,:,1]
1 -1 0
1 0 0
-1 0 0
w1[:,:,2]
-1 1 0
-1 1 1
1 1 -1
Bias b0 (1x1x1)
b0[:,:,0]
1
Bias b1 (1x1x1)
b1[:,:,0]
0
Output Volume (3x3x2)
o[:,:,0]
3 3 5
7 3 1
0 2 4
o[:,:,1]
2 5 6
0 0 5
-3 3 -3

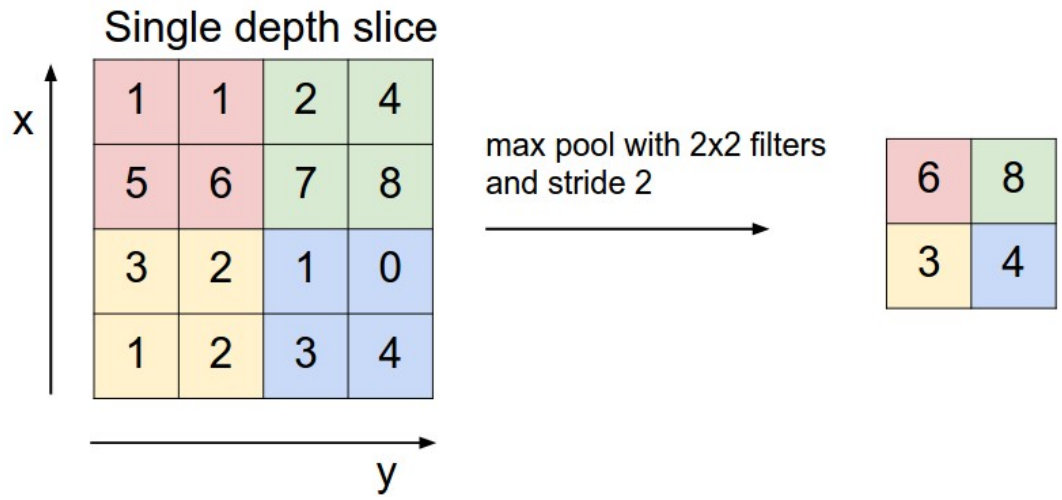

Input Volume (+pad 1) (7x7x3)
x[:,:,0]
0 0 0 0 0 0 0
0 2 2 0 0 2 0
0 1 0 0 0 0 0
0 0 2 0 1 1 0
0 0 2 2 2 0 0
0 0 1 2 2 0 0
0 0 0 0 0 0 0
x[:,:,1]
0 0 0 0 0 0 0
0 0 0 1 2 0 0
0 0 1 2 0 1 0
0 0 0 0 0 0 0
0 1 2 0 1 0 0
0 2 0 0 0 1 0
0 0 0 0 0 0 0
x[:,:,2]
0 0 0 0 0 0 0
0 2 0 1 2 2 0
0 1 2 0 1 1 0
0 1 1 0 1 2 0
0 0 0 2 2 0 0
0 1 0 2 2 0 0
0 0 0 0 0 0 0
Filter W0 (3x3x3)
w0[:,:,0]
0 0 -1
1 0 0
0 0 1
w0[:,:,1]
0 -1 0
0 1 -1
-1 1 1
w0[:,:,2]
-1 1 0
1 0 0
0 1 0
Bias b0 (1x1x1)
b0[:,:,0]
1
Filter W1 (3x3x3)
w1[:,:,0]
-1 -1 -1
1 1 -1
1 1 0
w1[:,:,1]
1 -1 0
1 0 0
-1 0 0
w1[:,:,2]
-1 1 0
-1 1 1
1 1 -1
Bias b1 (1x1x1)
b1[:,:,0]
0
Output Volume (3x3x2)
o[:,:,0]
3 3 5
7 3 1
0 2 4
o[:,:,1]
2 5 6
0 0 5
-3 3 -3


Input Volume (+pad 1) (7x7x3)
x[:,:,0]
0 0 0 0 0 0 0
0 2 2 0 0 2 0
0 1 0 0 0 0 0
0 0 2 0 1 1 0
0 0 2 2 2 0 0
0 0 1 2 2 0 0
0 0 0 0 0 0 0
x[:,:,1]
0 0 0 0 0 0 0
0 0 0 1 2 0 0
0 0 1 2 0 1 0
0 0 0 0 0 0 0
0 1 2 0 1 0 0
0 2 0 0 0 1 0
0 0 0 0 0 0 0
x[:,:,2]
0 0 0 0 0 0 0
0 2 0 1 2 2 0
0 1 2 0 1 1 0
0 1 1 0 1 2 0
0 0 0 2 2 0 0
0 1 0 2 2 0 0
0 0 0 0 0 0 0
Filter W0 (3x3x3)
w0[:,:,0]
0 0 -1
1 0 0
0 0 1
w0[:,:,1]
0 -1 0
0 1 -1
-1 1 1
w0[:,:,2]
-1 1 0
1 0 0
0 1 0
Bias b0 (1x1x1)
b0[:,:,0]
1
Filter W1 (3x3x3)
w1[:,:,0]
-1 -1 -1
1 1 -1
1 1 0
w1[:,:,1]
1 -1 0
1 0 0
-1 0 0
w1[:,:,2]
-1 1 0
-1 1 1
1 1 -1
Bias b1 (1x1x1)
b1[:,:,0]
0
Output Volume (3x3x2)
o[:,:,0]
3 3 5
7 3 1
0 2 4
o[:,:,1]
2 5 6
0 0 5
-3 3 -3

Convolutional networks

SUBSAMPLING LAYER



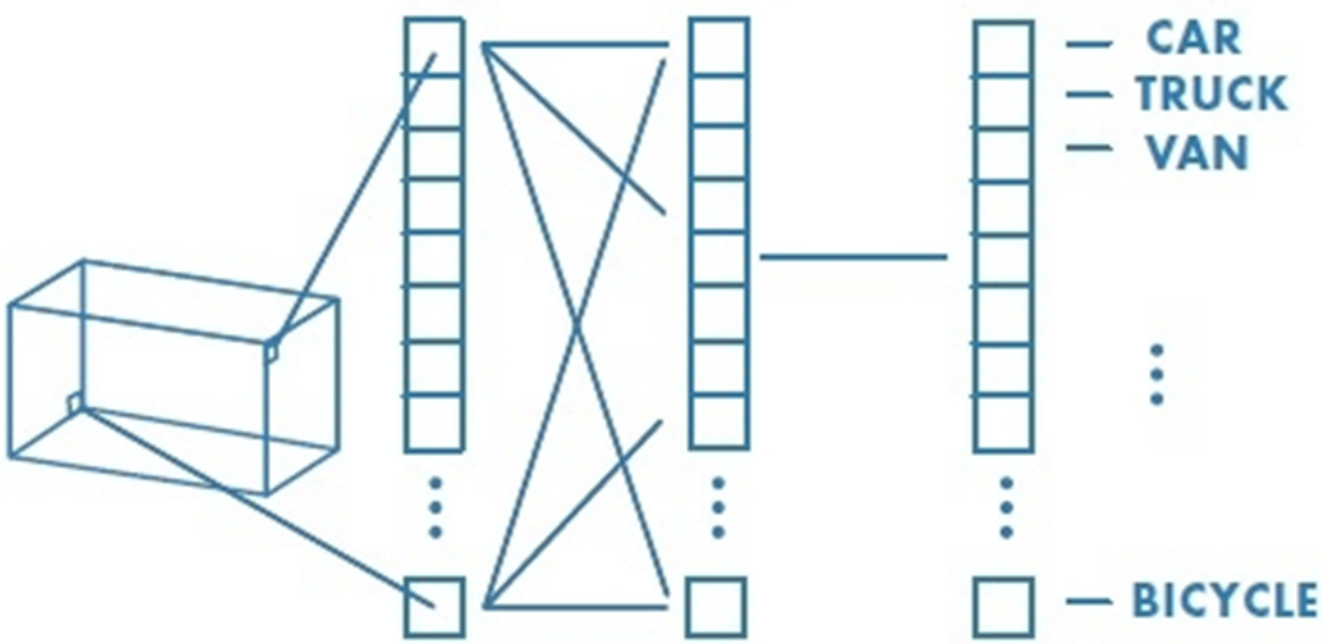


This reduces the number of inputs to the next layer of feature extraction, thus allowing us to have many more different feature maps.

- reduce the effect of noises and shift or distortion
- reduce the overfitting

Convolutional networks

FLATTEN LAYER



Flattening is the process of converting all the resultant 2 dimensional arrays into a single long continuous linear vector.

Convolutional networks

CONCLUSIONS ON PARAMETERS

Size of the convolutional kernels

- Smaller filters collect as much local information as possible, bigger filters represent more global, high-level and representative information.
- It is a common standard to use odd numbers

Padding

- Padding is generally used to add columns and rows of zeroes to keep the spatial sizes constant after convolution, doing this might improve performance as it retains the information at the borders

Convolutional networks

CONCLUSIONS ON PARAMETERS

Stride

- It is used to throw away the duplicate information.

Number of channels

- It is the equal to the number of color channels for the input but in later stages is equal to the number of filters we use for the convolution operation.
- The more the number of channels, more the number of filters used, more are the features learnt, and more is the chances to over-fit and vice-versa.

Convolutional networks

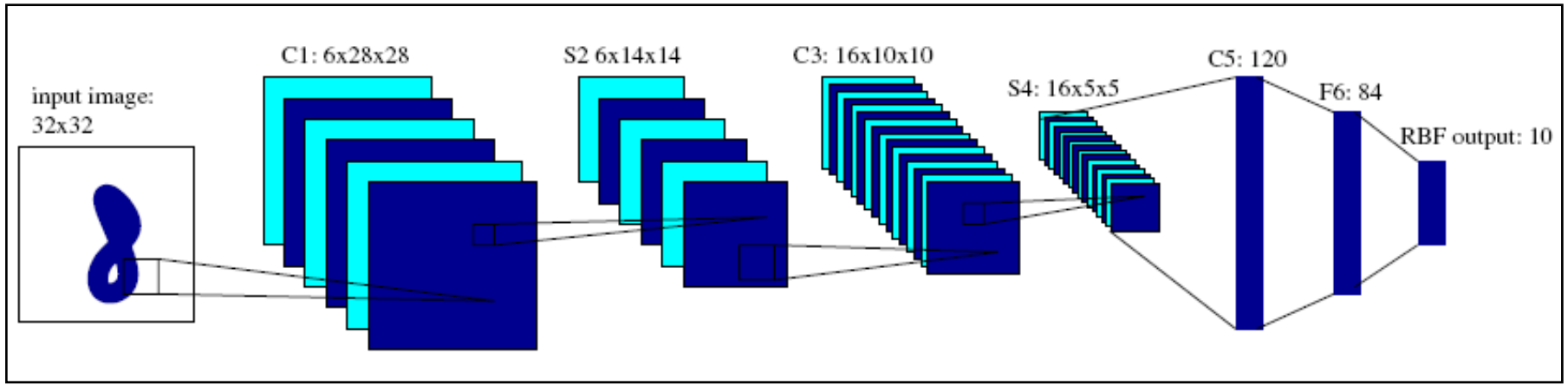
CONCLUSIONS ON PARAMETERS

- Always start by using smaller filters is to collect as much local information as possible, and then gradually increase the filter width to reduce the generated feature space width to represent more global, high-level and representative information
- The number of channels should be low in the beginning such that it detects low-level features which are combined to form many complex shapes (by increasing the number of channels) which help distinguish between classes
- Keep adding layers until you over-fit.
- Always use classic networks like LeNet, AlexNet, VGG-16, VGG-19 etc

Training Neural architectures

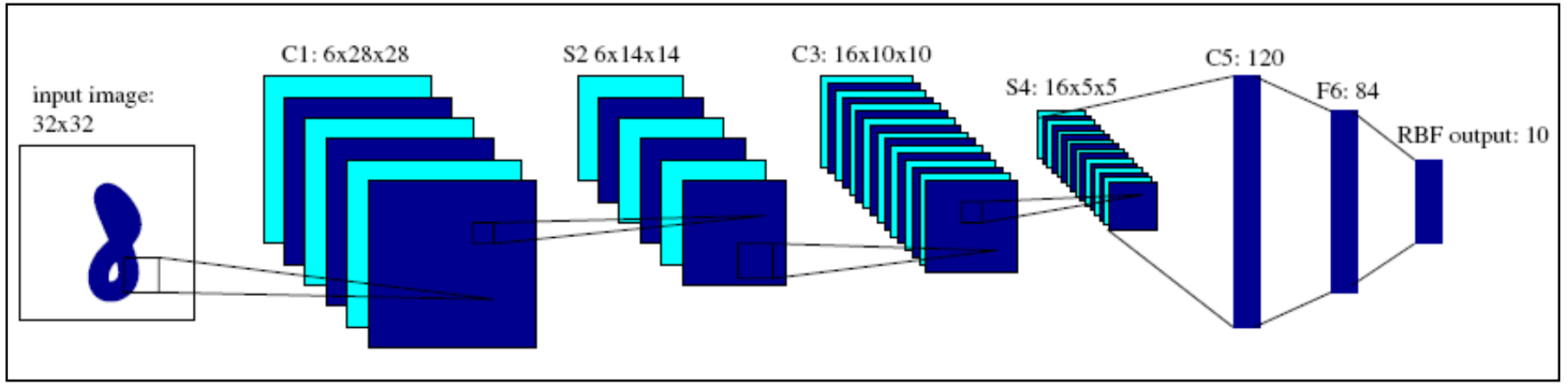
LENET 5

- C1, C3, C5: Convolutional layer.
- 5 × 5: Convolution matrix.
- S2, S4: Subsampling layer.
- Subsampling by factor 2.
- F6: Fully connected layer.



Training Neural architectures

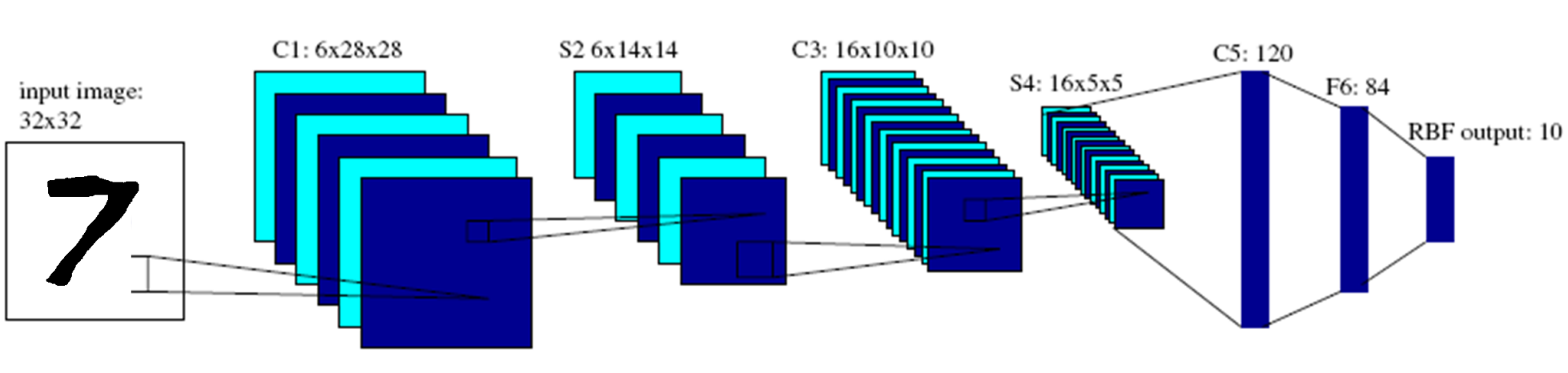
LENET 5



- About 187,000 connections
- About 14,000 trainable weights

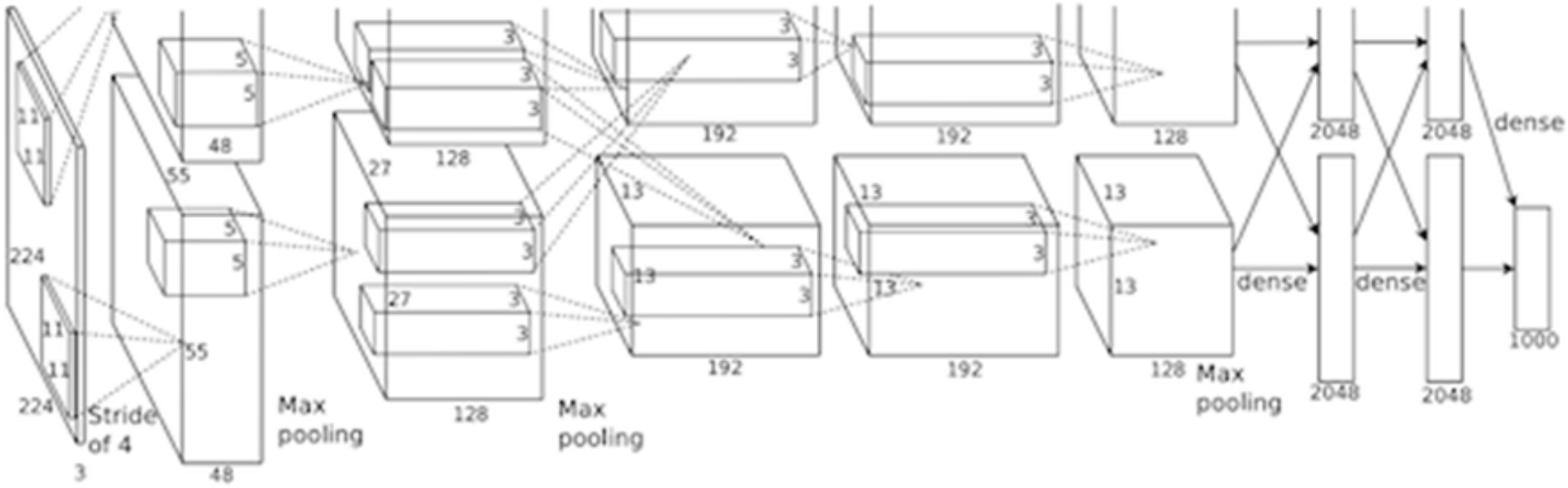
Training Neural architectures

LENET 5



Training Neural architectures

ALEX NET



- It consisted 11x11 5x5,3x3 convolutions
- max pooling
- dropout
- data augmentation
- ReLU activations
- SGD with momentum

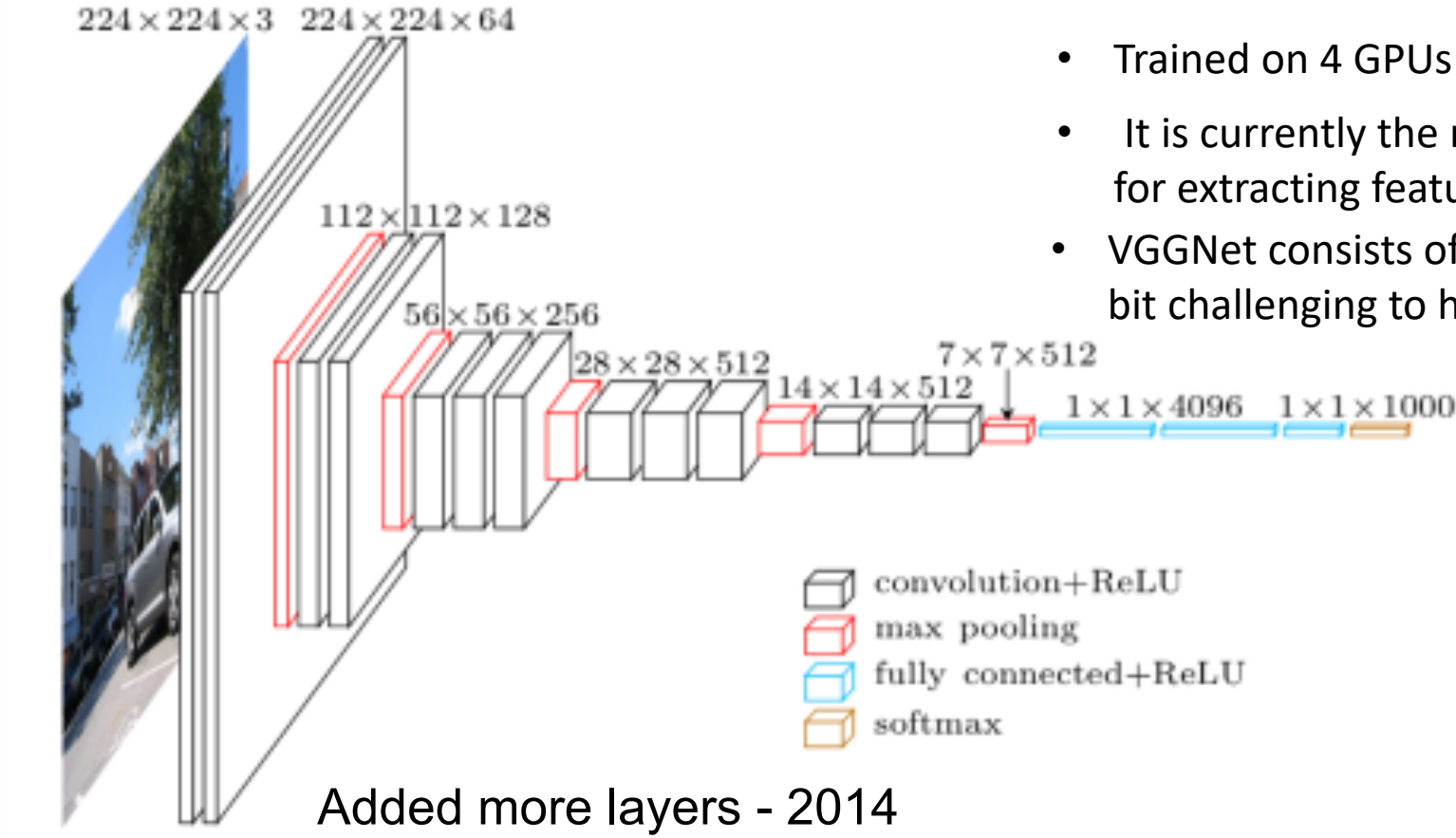
AlexNet architecture (May look weird because there are two different "streams". This is because the training process was so computationally expensive that they had to split the training onto 2 GPUs)

Introduced the concept of dropout and ReLU
Winner on LSVRC2012

[http://vision.stanford.edu/teaching/cs231b_spring1415/slides/alexnet_tugce_kyunghee.pdf]

Training Neural architectures

VGG



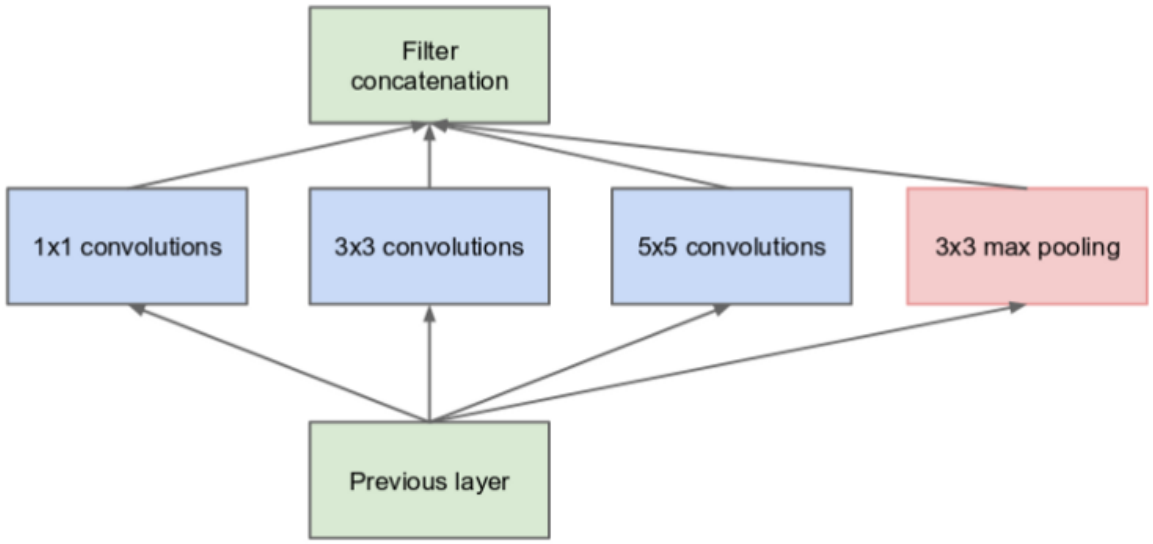
Added more layers - 2014

- Similar to AlexNet, only 3x3 convolutions, but lots of filters.
- Trained on 4 GPUs for 2–3 weeks
- It is currently the most preferred choice in the community for extracting features from images (Transfer Learning)
- VGGNet consists of **138 million parameters**, which can be a bit challenging to handle

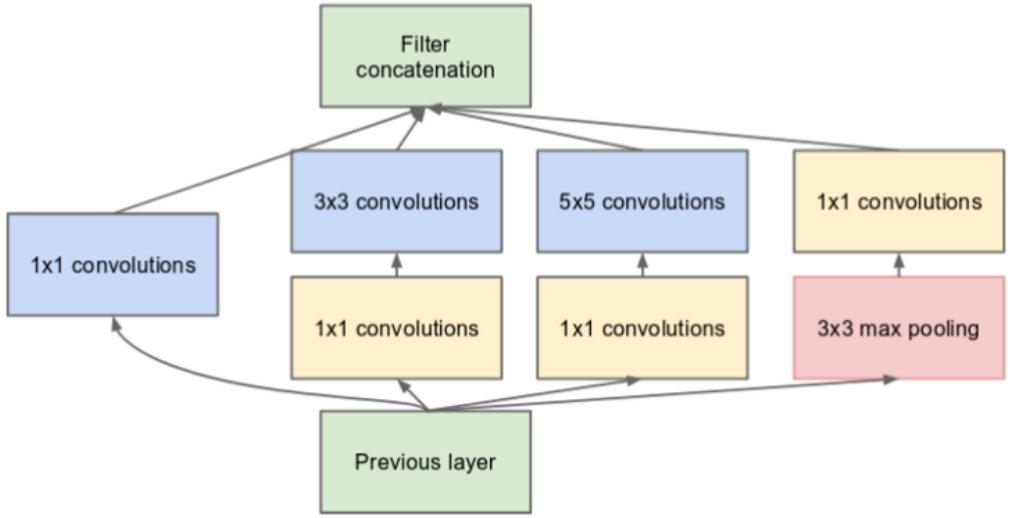
[Simonyan and Zisserman 2014]

Training Neural architectures

GOOGLE LENET (INCEPTION ARCHITECTURE)



(a) Inception module, naïve version

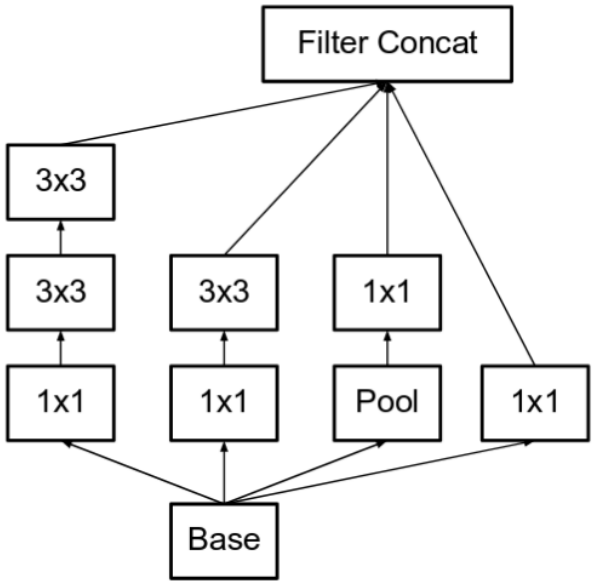


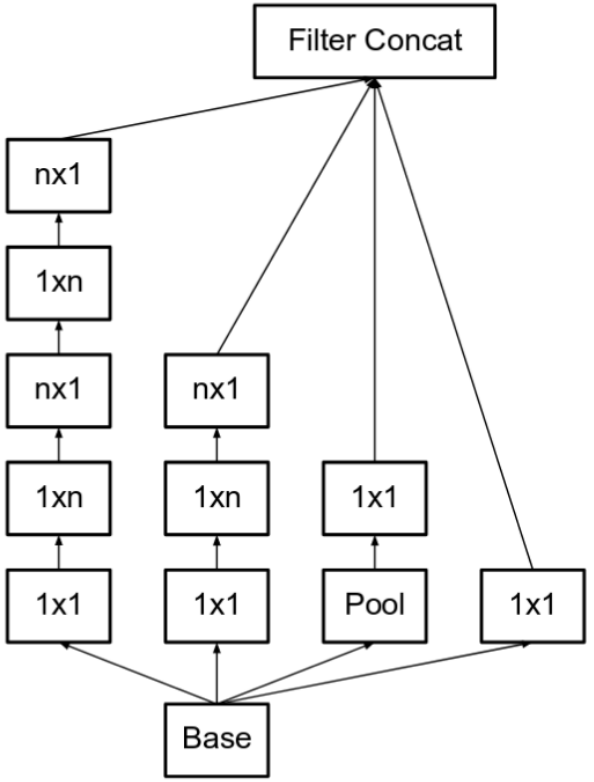
(b) Inception module with dimension reductions

Added more layers - 2015

Training Neural architectures

GOOGLE LENET (INCEPTION ARCHITECTURE) – V2/3 IMPROVEMENTS

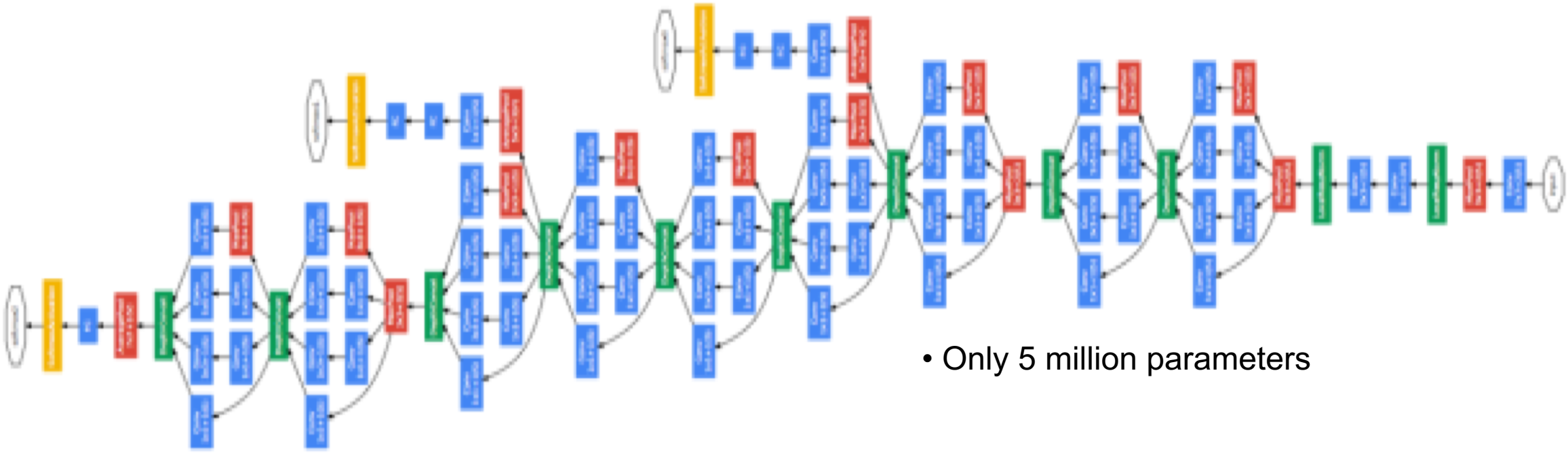




Factorize 5x5 convolution **to two 3x3** convolution operations to improve computational speed. Although this may seem counterintuitive, a 5x5 convolution is **2.78 times more expensive** than a 3x3 convolution.

Training Neural architectures

GOOGLE LENET (INCEPTION ARCHITECTURE)

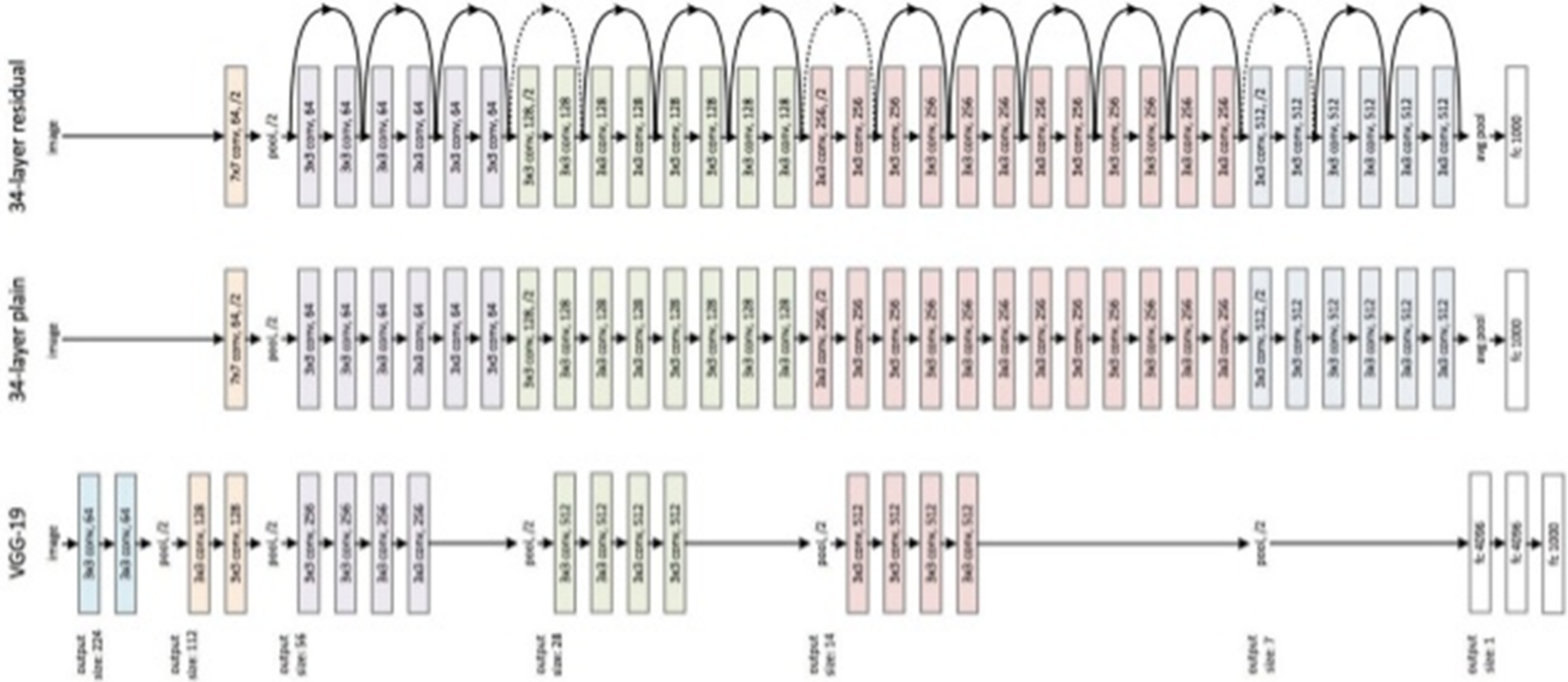


- Only 5 million parameters

Added more layers - 2015

Training Neural architectures

RESNET



[https://towardsdatascience.com]

2016

Training Neural architectures

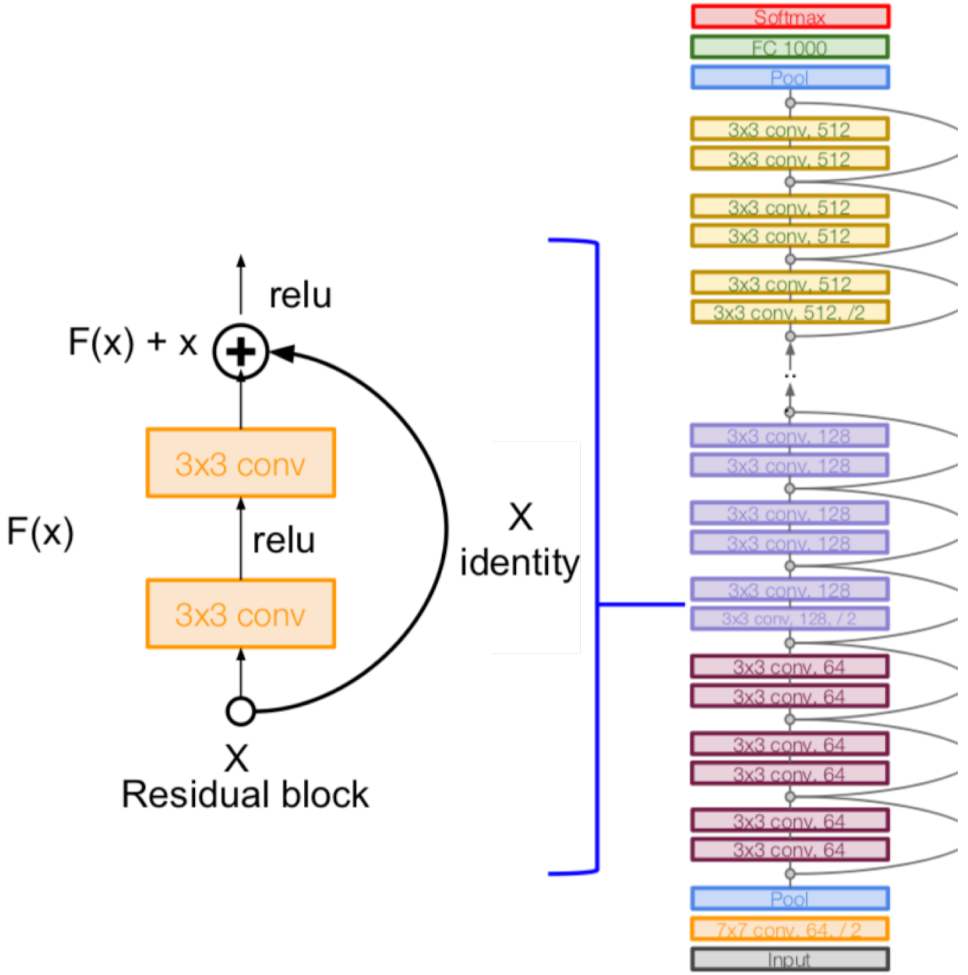
RESNET

Full ResNet architecture:

- Stack residual blocks
- Every residual block has two 3x3 conv layers
- Periodically, double # of filters and downsample

F(x) spatially using stride 2 (/2 in each dimension)

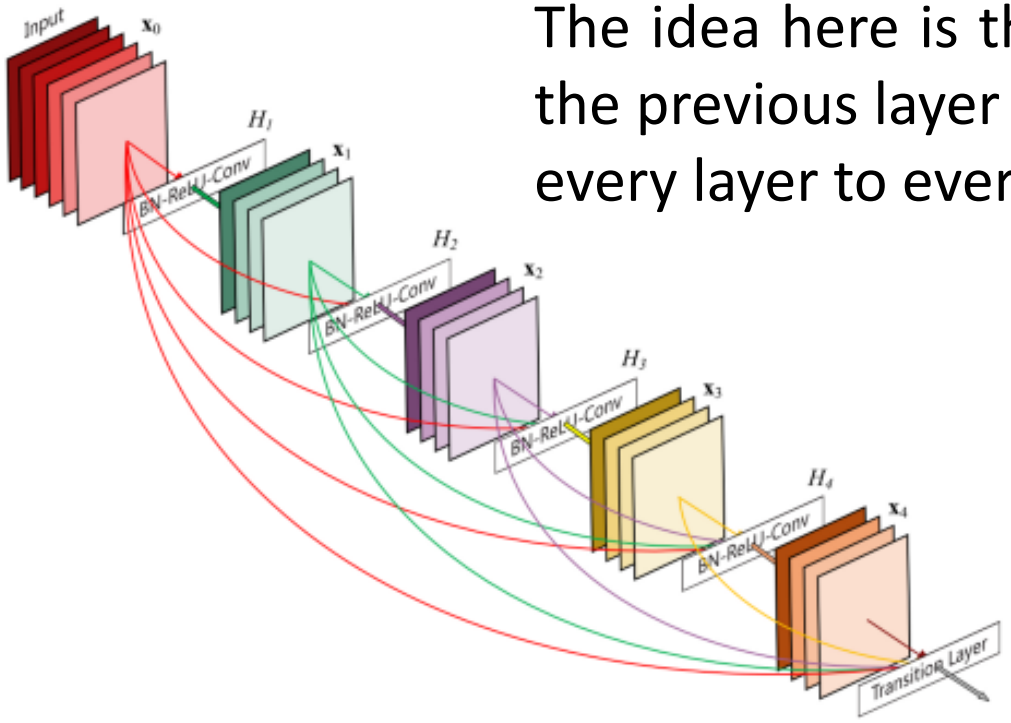
- Additional conv layer at the beginning
- No FC layers at the end (only FC 1000 to output classes)



[He et al. 2015]

Training Neural architectures

DENSENET

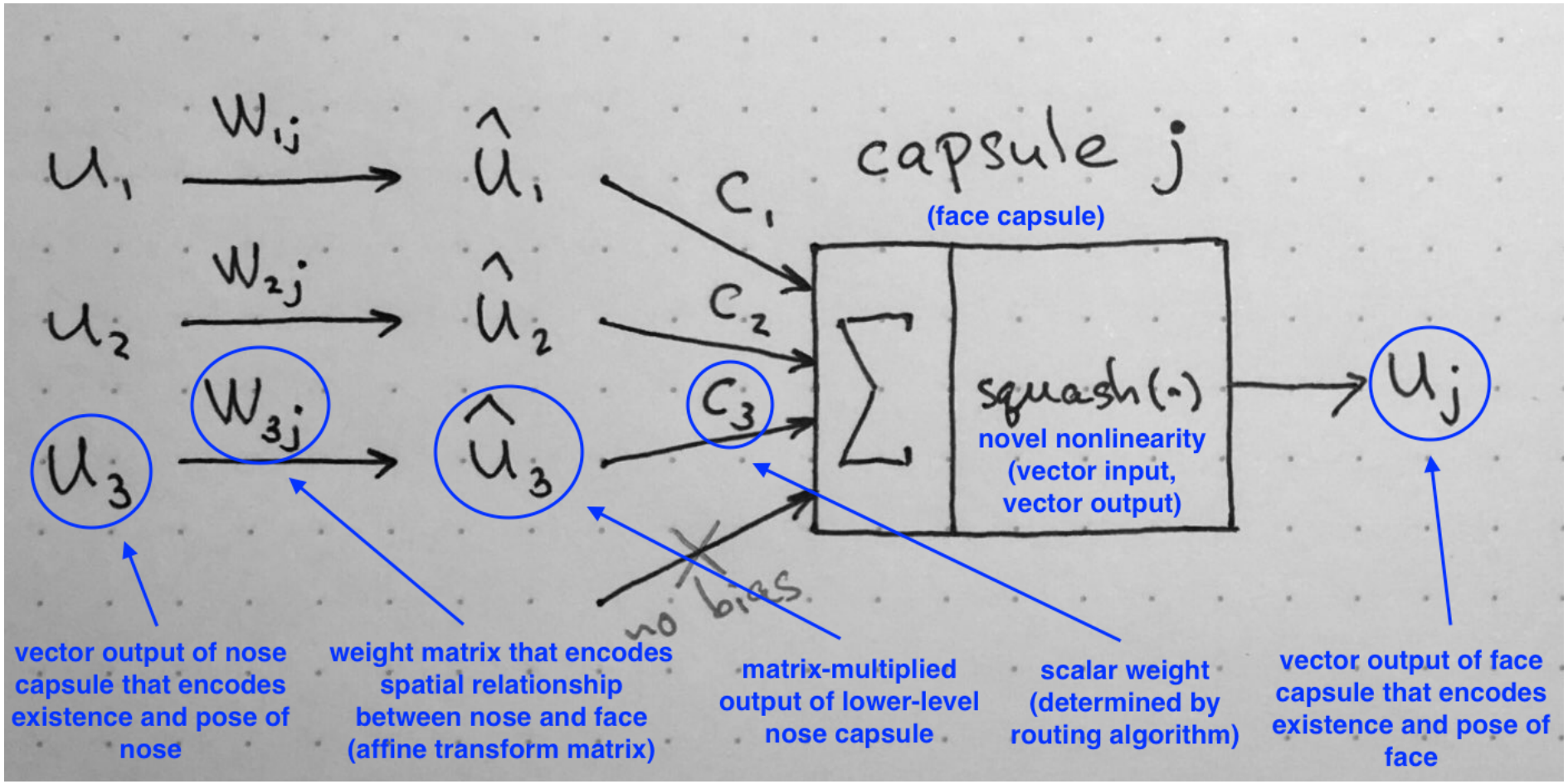


The idea here is that if connecting a skip connection from the previous layer improves performance, why not connect every layer to every other layer?

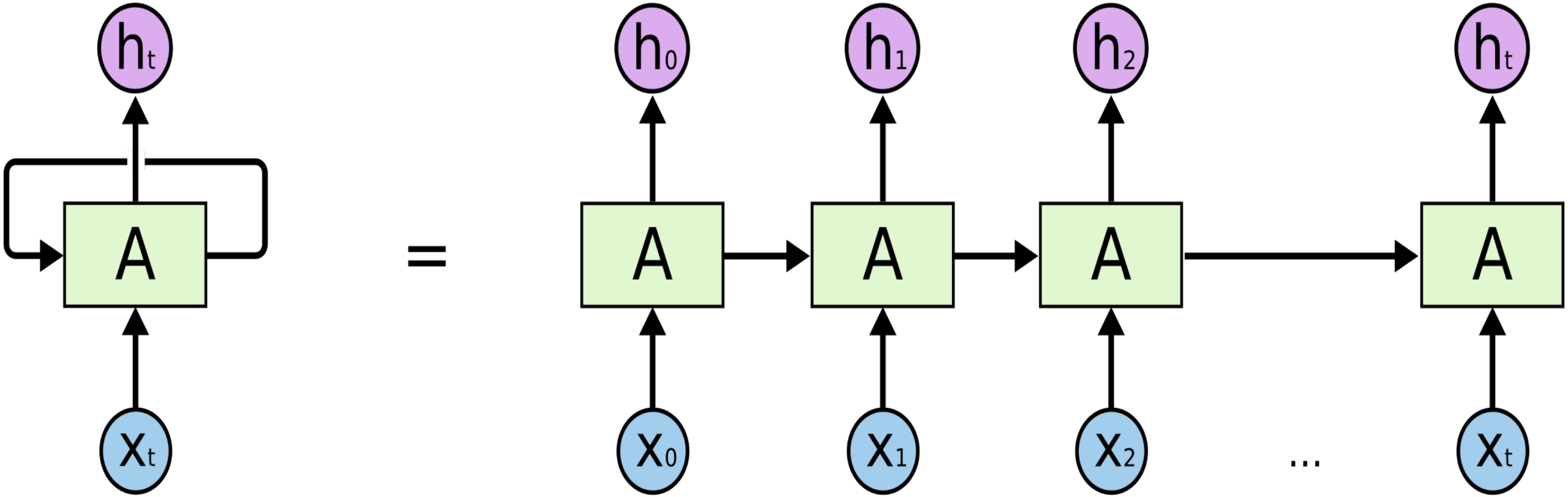
[https://towardsdatascience.com]

Training Neural architectures

CAPSULE NETWORKS

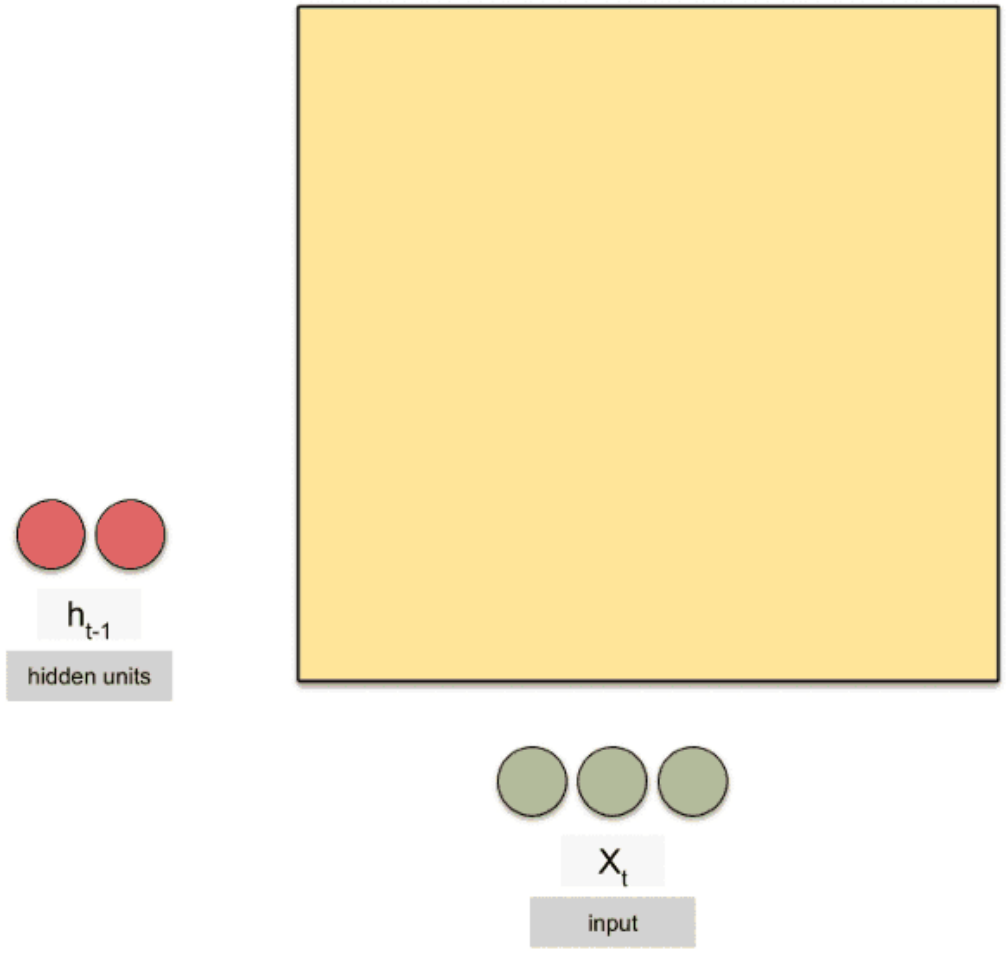


Recurrent Neural Networks



Recurrent Neural Networks

SIMPLE RNN



[https://towardsdatascience.com/animated-rnn-lstm-and-gru-ef124d06cf45]

Recurrent Neural Networks

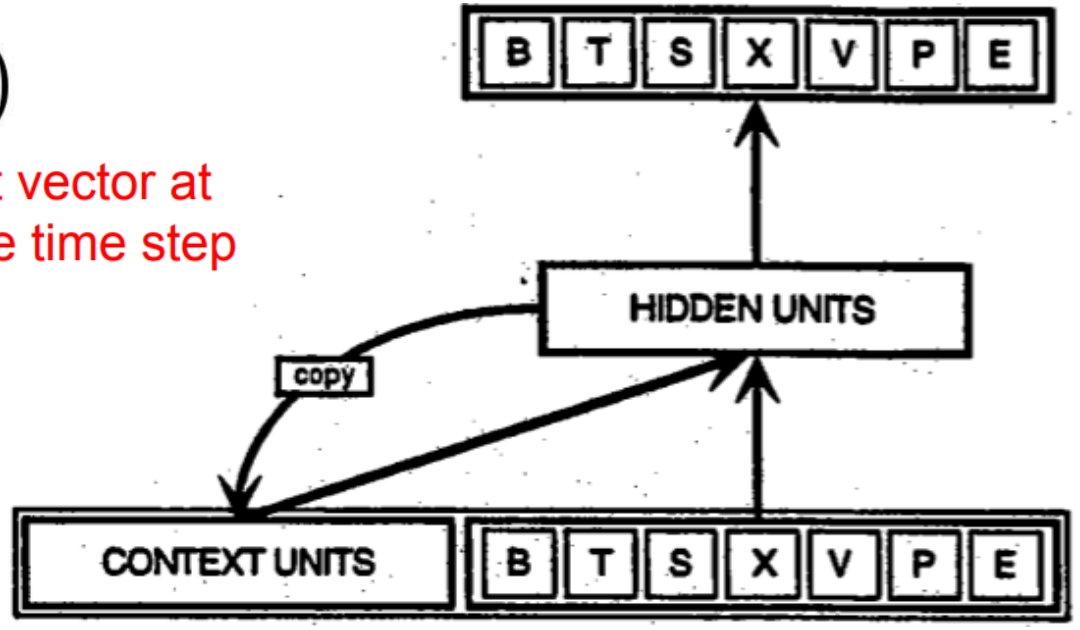
$$h_t = f_W(h_{t-1}, x_t)$$

new state

old state

input vector at some time step

some function with parameters W



Recurrent Neural Networks

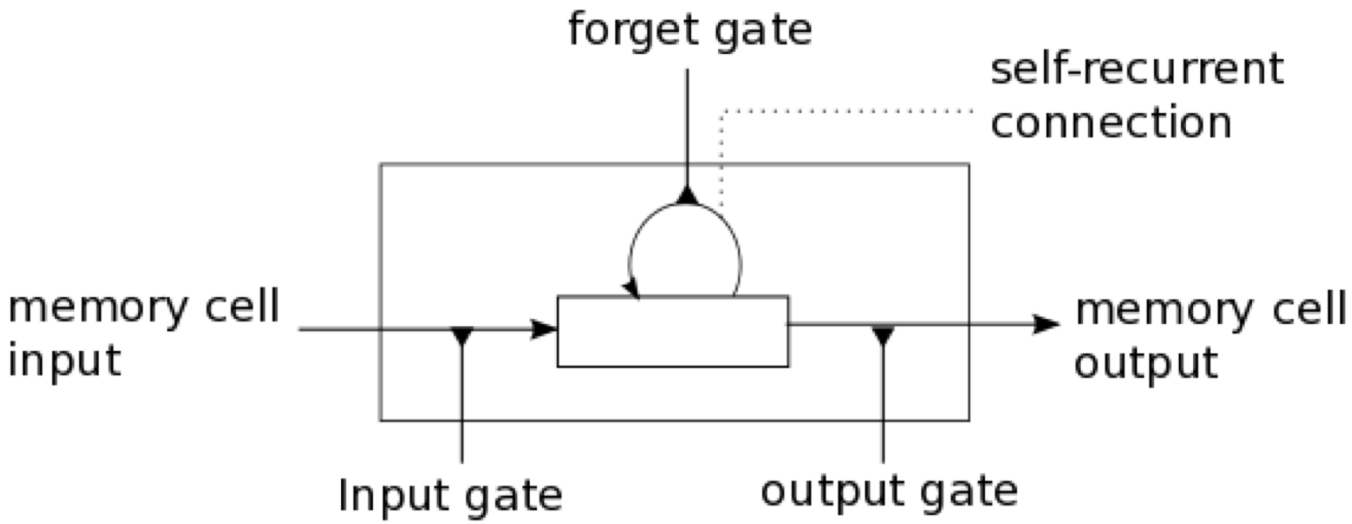
PROBLEM OF LONG-TERM DEPENDENCIES

E.g. 1: The grass is green

E.g. 2: I am Romanian. I love …(200 words)… My mother tongue is Romanian.

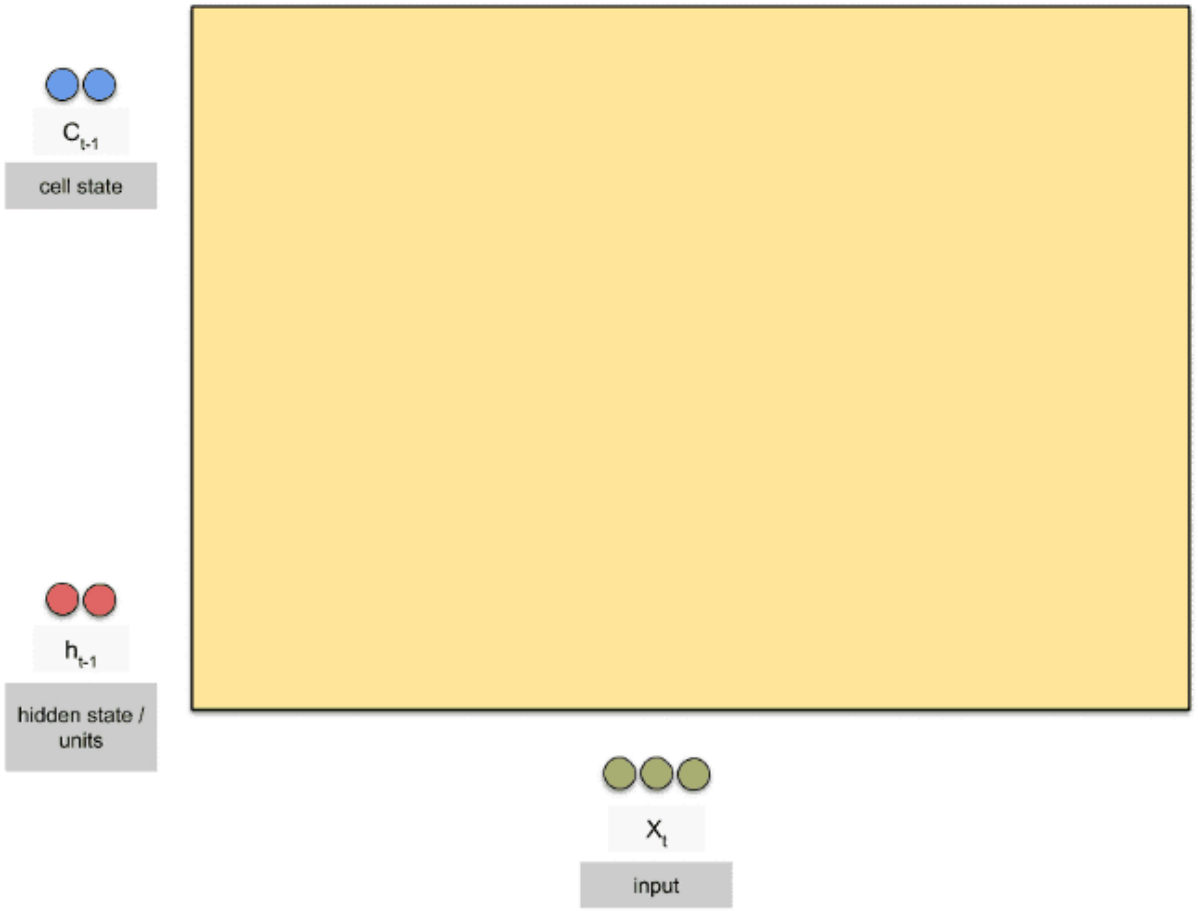
Recurrent Neural Networks

LONG-SHORT TERM MEMORY (LSTM) / GRU



Recurrent Neural Networks

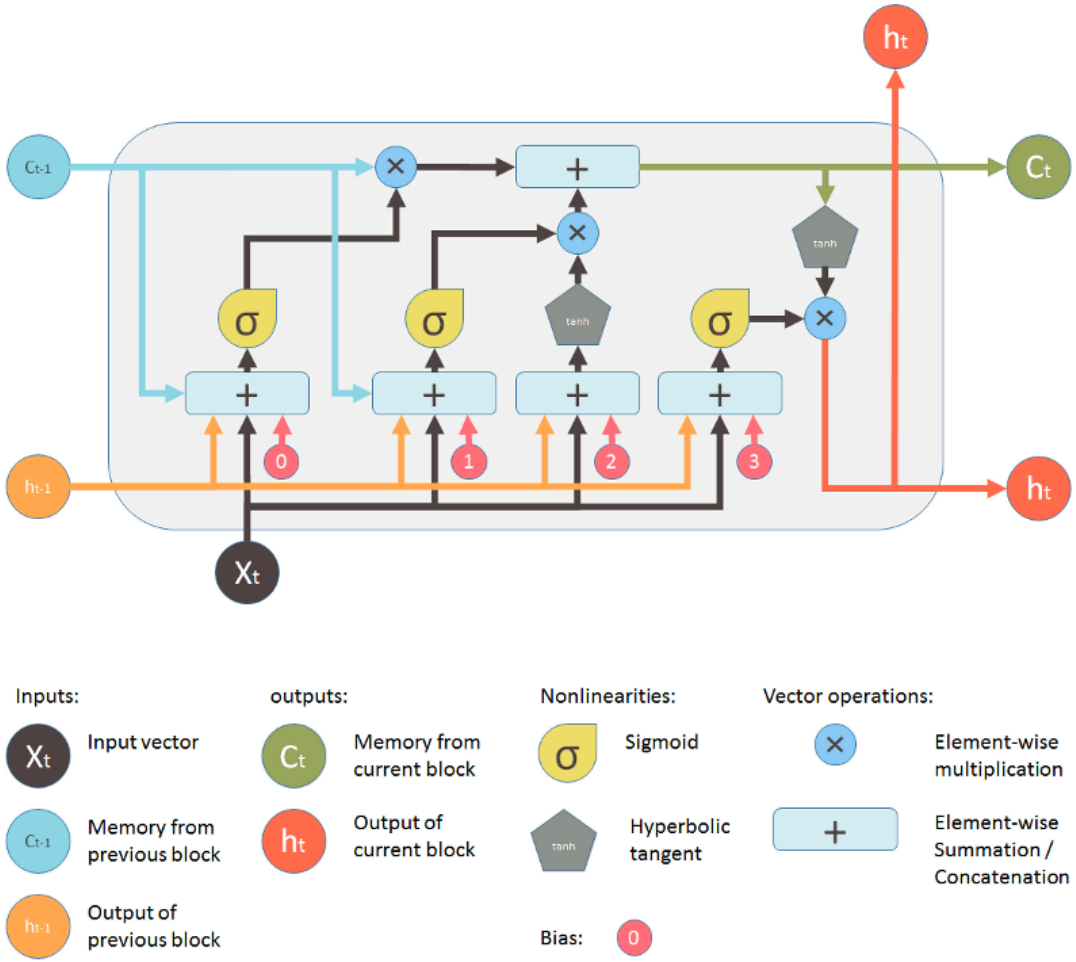
LSTM



[https://towardsdatascience.com/animated-rnn-lstm-and-gru-ef124d06cf45]

Recurrent Neural Networks

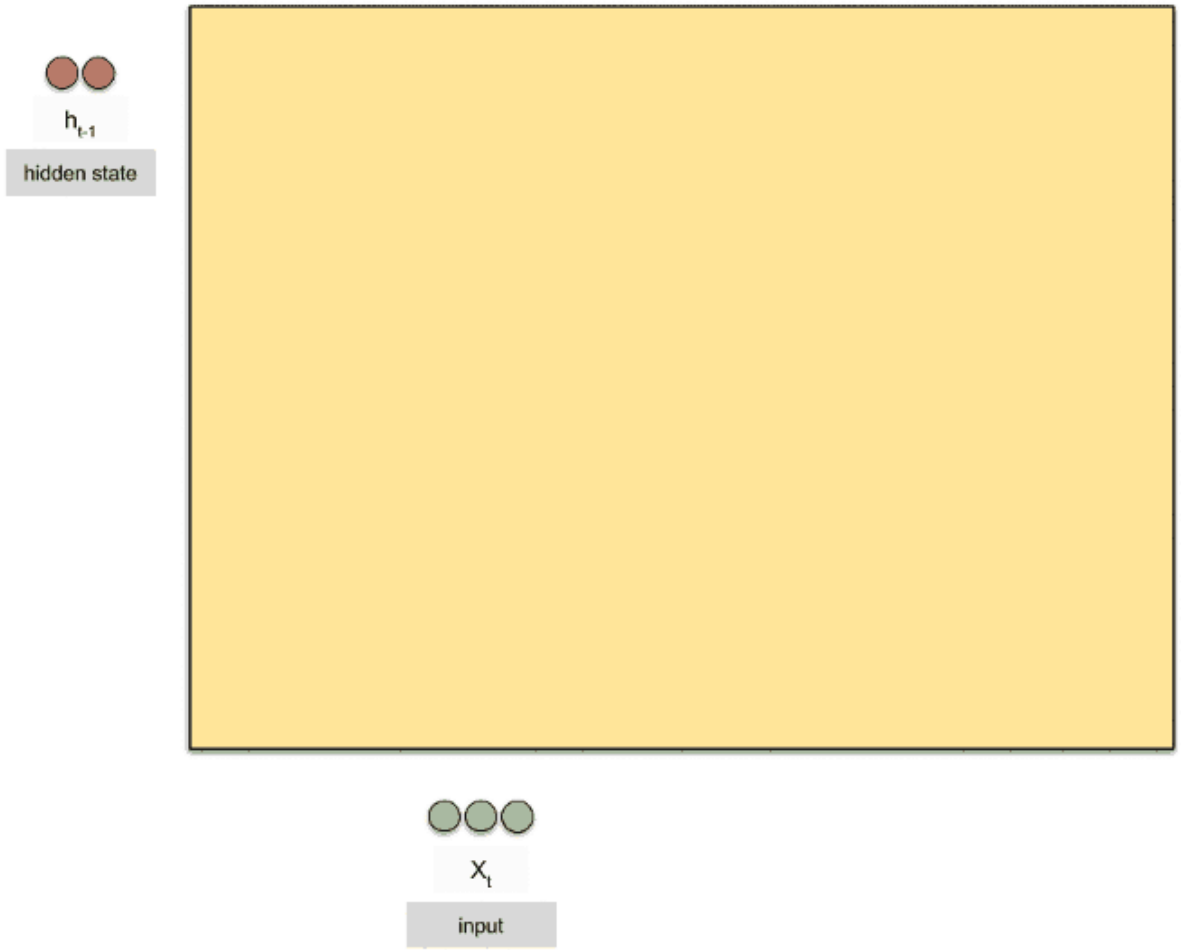
LSTM



https://medium.com/mlreview/understanding-lstm-and-its-diagrams-37e2f46f1714

Recurrent Neural Networks

GRU



[https://towardsdatascience.com/animated-rnn-lstm-and-gru-ef124d06cf45]

Transfer Learning

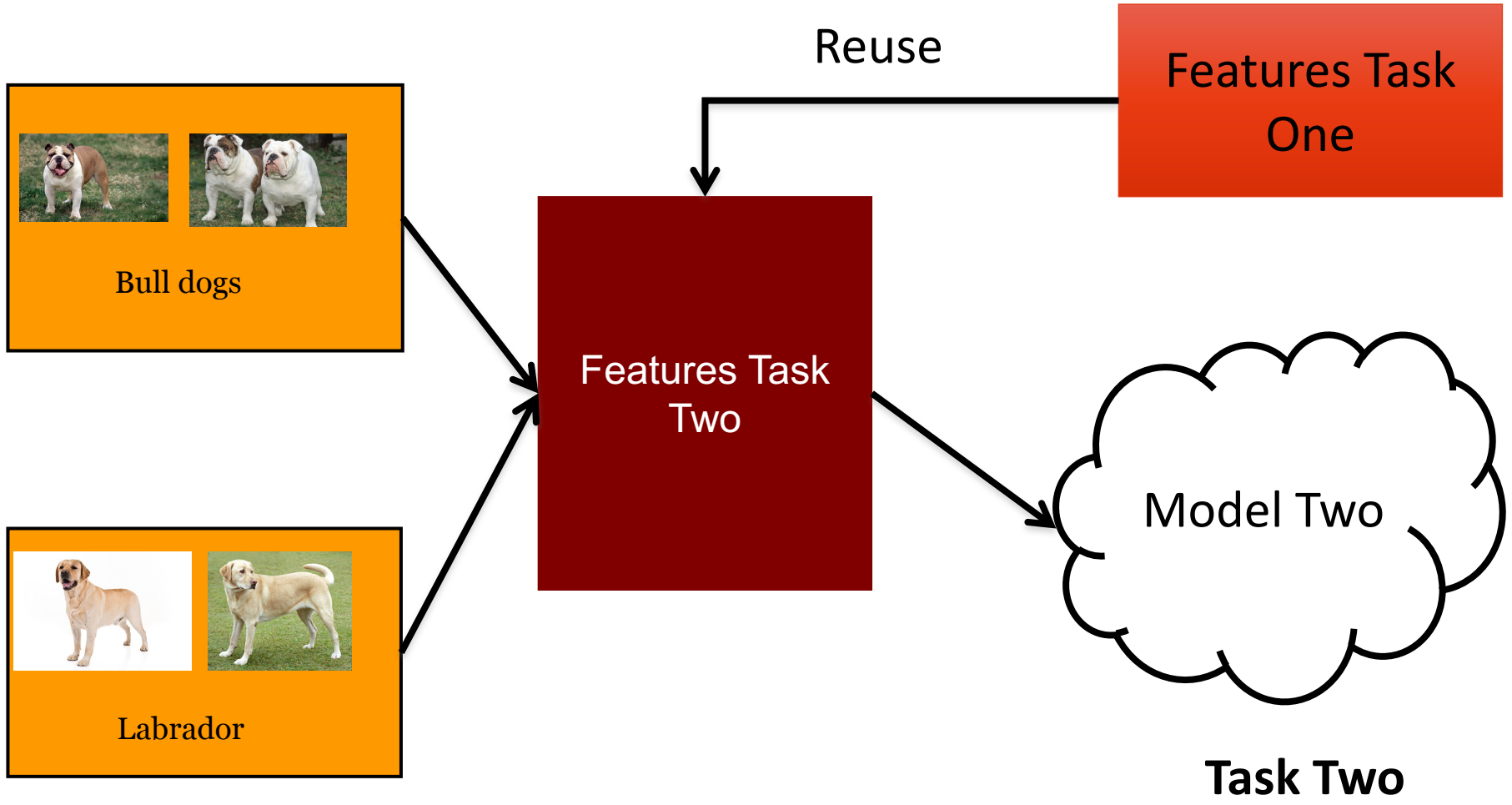
Transfer Learning (TL):

The ability of a system to recognize and apply knowledge and skills learned in previous tasks to novel tasks (in new domains)

ASSUMPTION: the source domain and target domain data use exactly the same features and labels.

MOTIVATION: Although the source domain data can not be reused directly, there are some parts of the data that can still be reused by re-weighting.

Transfer Learning



Transfer Learning

VERSION 1

Take a certain layer as feature → Use another classifier

Transfer Learning

VERSION 2

Replace the last layers and retrain only the last layers of the network

Transfer Learning

VERSION 3

Replace the last layers and retrain all the network

Applications of Neural Networks

IMAGE CLASSIFICATION, OBJECT DETECTION, LOCALIZATION, ACTION RECOGNITION, SCENE UNDERSTANDING

[Image source: www.nvidia.com]

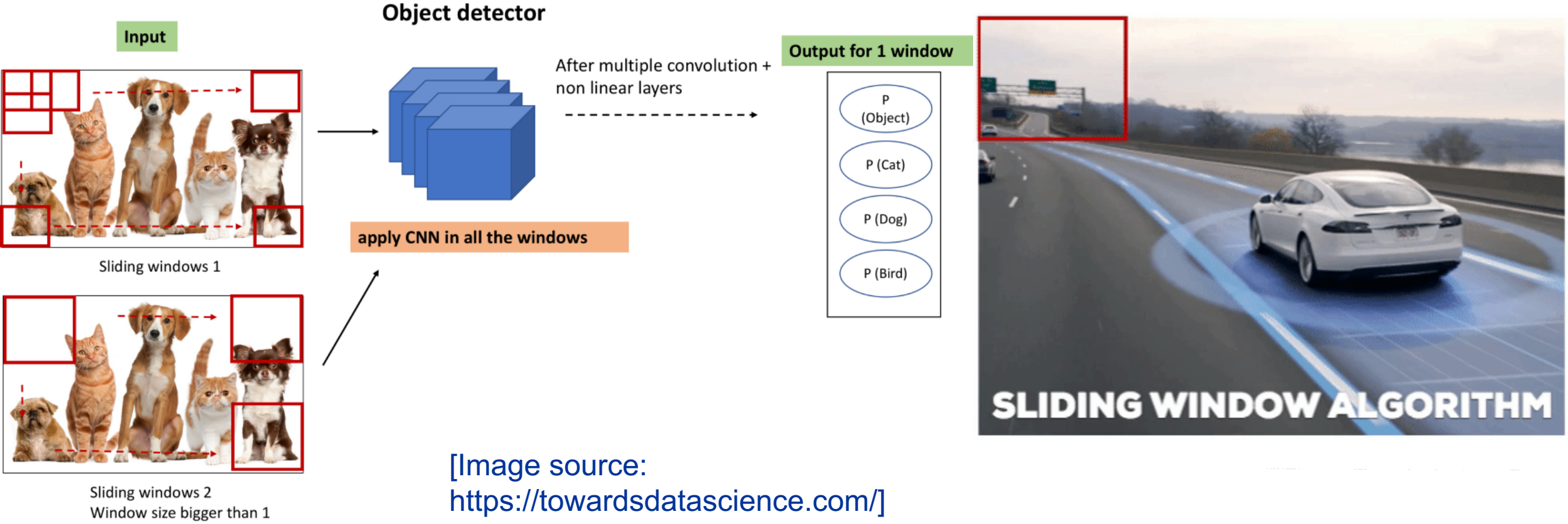
Applications of Neural Networks

PEDESTRIAN DETECTION, TRAFFIC SIGN RECOGNITION

[Image source: www.nvidia.com]

Applications of Neural Networks

IMAGE CLASSIFICATION, OBJECT DETECTION, LOCALIZATION, ACTION RECOGNITION, SCENE UNDERSTANDING



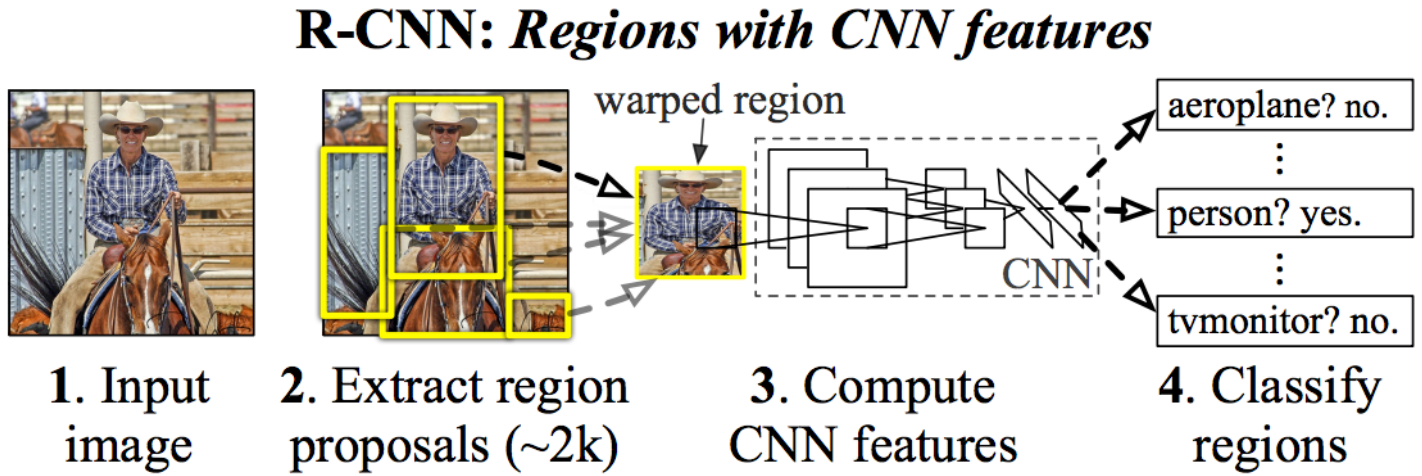
[Image source: https://towardsdatascience.com/]

Applications of Neural Networks

IMAGE CLASSIFICATION, OBJECT DETECTION, LOCALIZATION

(a)

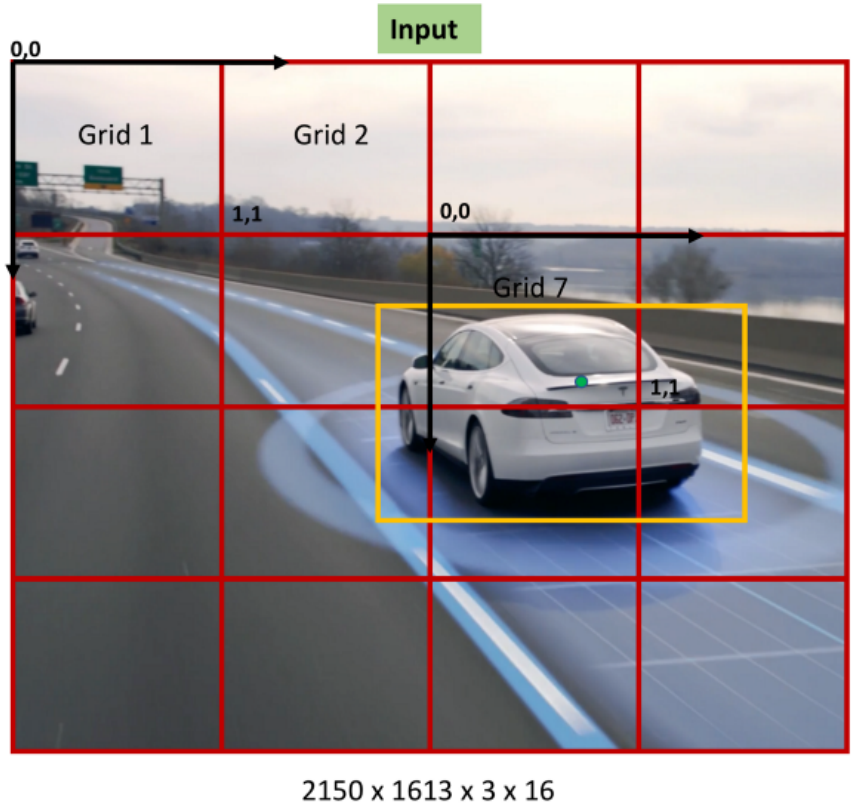
Figure 2: Two examples of our selective search showing the necessity of different scales
scales. On the right we necessarily find the objects at different scales as the girl is contai



[Images source: https://blog.athelas.com/a-brief-history-of-cnns-in-image-segmentation-from-r-cnn-to-mask-r-cnn-34ea83205de4/]

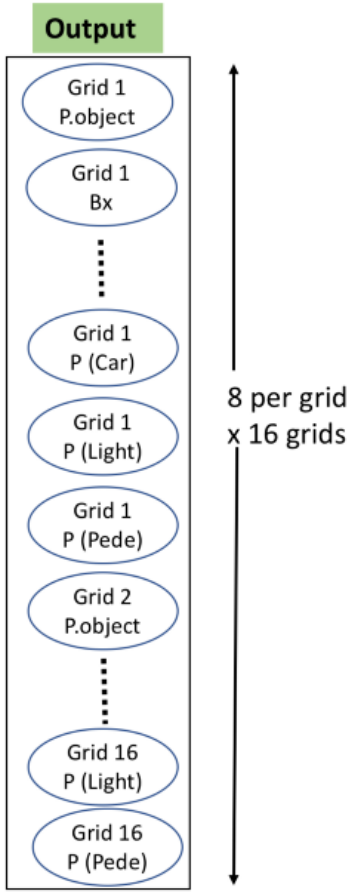
Applications of Neural Networks

IMAGE CLASSIFICATION, OBJECT DETECTION, LOCALIZATION, ACTION RECOGNITION, SCENE UNDERSTANDING



Sample output
Grid 7
P.Object : 1
Bx : 0.8
By : 0.9
Bw : 1.5
Bh : 1.2
P.Car : 1
P.Light : 0
P.Pedestrian : 0

P.Object : 0 for all other grids and we don't care about other values if P.Object is 0

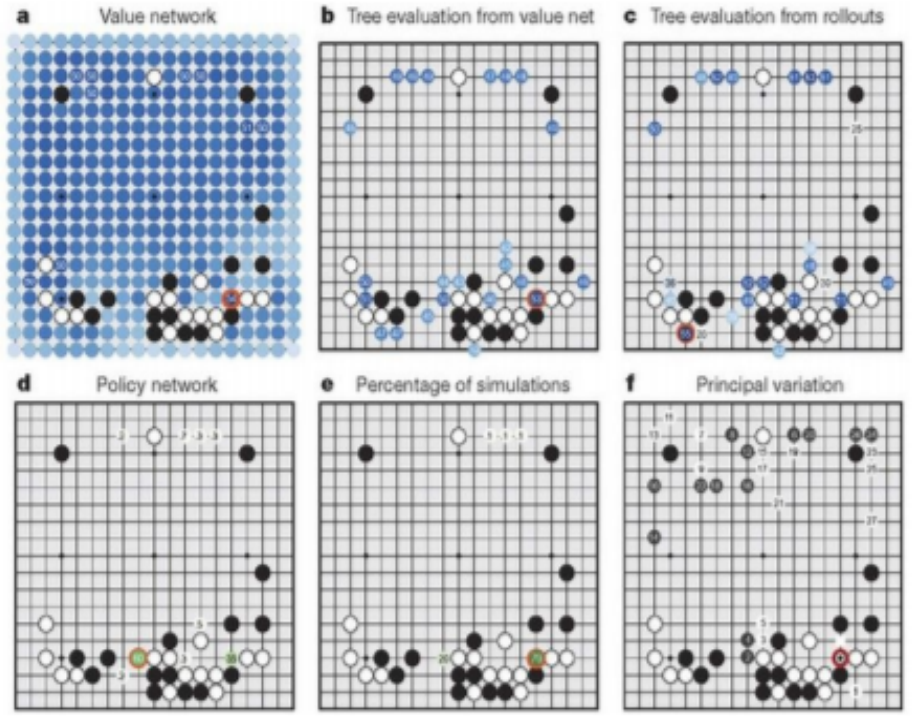


Yolo: You Only Look Once

[Image source: https://towardsdatascience.com/]

Applications of Neural Networks

DEEPMIND'S ALPHAGO



Applications of Neural Networks

ROAD FINDER

[Vlad Mnih, ICML 2012]

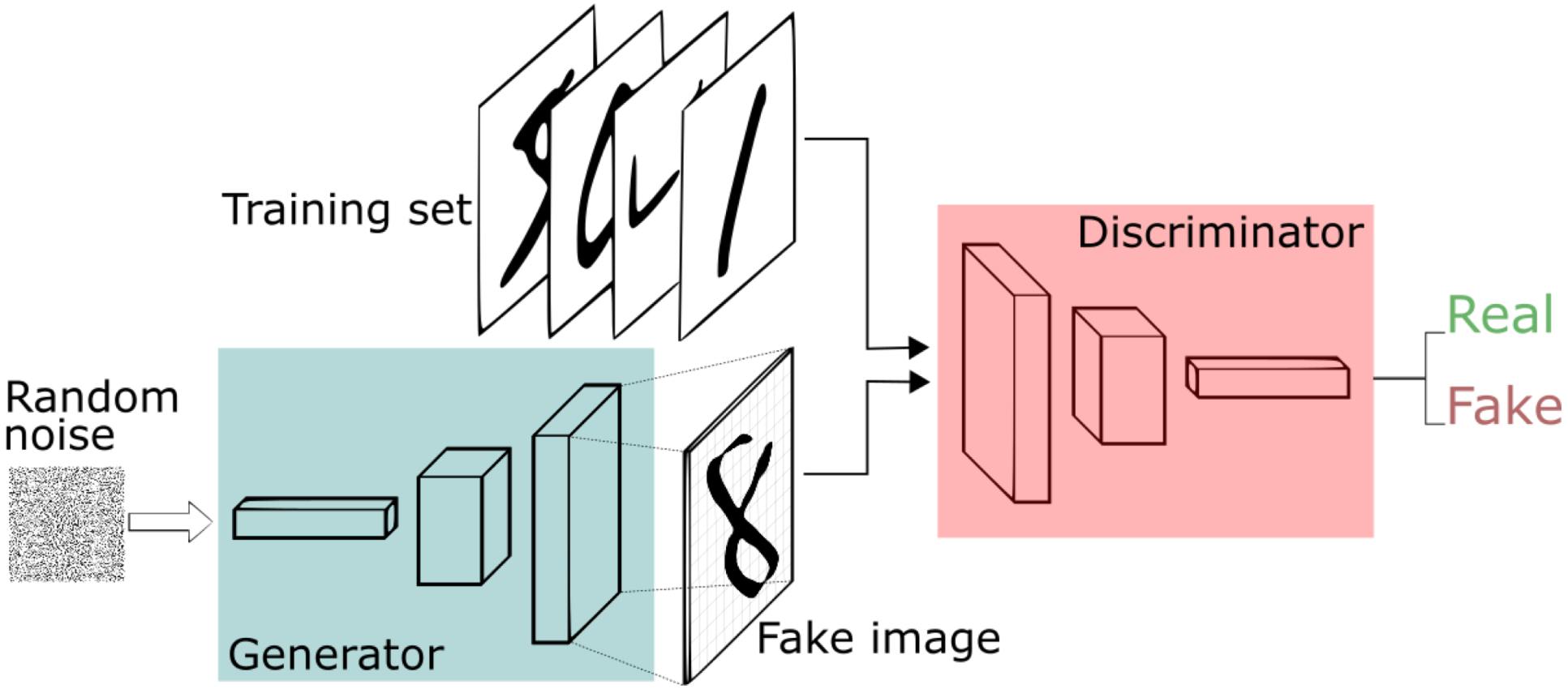
Applications of Neural Networks

ARTIFICIALLY EXPANDING THE TRAINING DATA

[Goodfellow et al.]

Applications of Neural Networks

GENERATIVE ADVERSIAL NETWORKS



[Goodfellow et al.]

[Image source O'Reilly]

Applications of Neural Networks

COMMON PROBLEMS WITH GENERATIVE ADVERSIAL NETWORKS

- The loss of the discriminator is going very rapidly to zero and the generator is not able to fool the discriminator
  - Take a look on the gradients of the generator vs discriminator (at the begining the discriminator should learn faster)

- **Mode Collapse** refers to the scenario when the Generator produces the same (or almost same) images every time and is able to successfully fool the discriminator.

Applications of Neural Networks

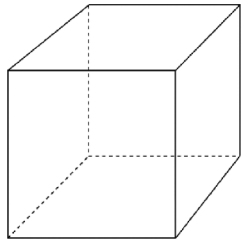
GENERATIVE ADVERSIAL NETWORKS TIPS AND TRICKS

Normalize the inputs

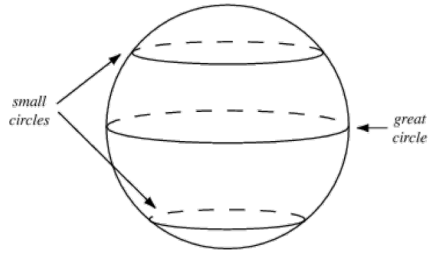
- normalize the images between -1 and 1
- Tanh as the last layer of the generator output

Generative sampling

- don’t use uniform distribution



- sample from a gaussian distribution



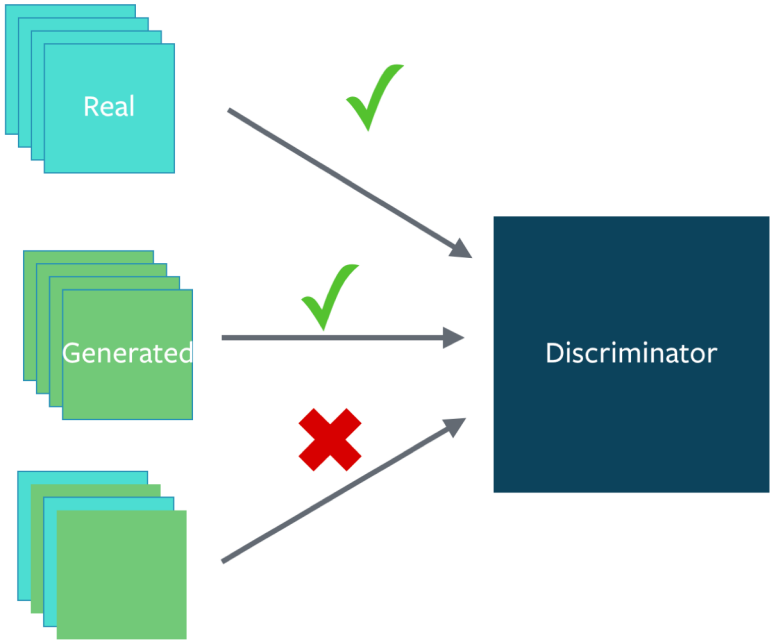
[Soumith Chintala et al.]

Applications of Neural Networks

GENERATIVE ADVERSIAL NETWORKS TIPS AND TRICKS

- Construct different mini-batches for real and fake, i.e. each mini-batch needs to contain only all real images or all generated images.

- When batchnorm is not an option use instance normalization (for each sample, subtract mean and divide by standard deviation).



[Soumith Chintala et al.]

Applications of Neural Networks

GENERATIVE ADVERSIAL NETWORKS TIPS AND TRICKS

- the stability of the GAN game suffers if you have sparse gradients
- LeakyReLU = good (in both G and D)
- for Downsampling, use: Average Pooling, Conv2d + stride
- for Upsampling, use: PixelShuffle, ConvTranspose2d + stride
- avoid Sparse Gradients: ReLU, MaxPool

[Soumith Chintala et al.]

Applications of Neural Networks

GENERATIVE ADVERSIAL NETWORKS TIPS AND TRICKS

Use Soft and Noisy Labels

• *Label Smoothing*, i.e. if you have two target labels: Real=1 and Fake=0, then for each incoming sample, if it is real, then replace the label with a random number between 0.7 and 1.2, and if it is a fake sample, replace it with 0.0 and 0.3 (for example).

- Salimans et. al. 2016

• make the labels the noisy for the discriminator: occasionally flip the labels when training the discriminator

[Soumith Chintala et al.]

Applications of Neural Networks

GENERATIVE ADVERSIAL NETWORKS TIPS AND TRICKS

- Use Adam optimizer
  - See Radford et. al. 2015
- Use SGD for discriminator and ADAM for generator

- Add some artificial noise to inputs to D (Arjovsky et. al., Huszar, 2016)
  - http://www.inference.vc/instance-noise-a-trick-for-stabilising-gan-training/
  - https://openreview.net/forum?id=Hk4_qw5xe
- Adding gaussian noise to every layer of generator (Zhao et. al. EBGAN)
  - Improved GANs: OpenAI code also has it (commented out)

[Soumith Chintala et al.]

Applications of Neural Networks

GAN - MUSIC GENERATION

http://www.asimovinstitute.org/analyzing-deep-learning-tools-music/

GAN – TEXT GENERATION

Applications of Neural Networks

AUTOMATIC TEXT GENERATION

Given "my name is ivan and i live in stockholm" generated:

my name is ivan and i live in stockholm and i can do it and i wanna get closer so call me all alone i said its all i act so good and i wanna know the time i could stay in the world, youre always the only one that i wanna be...

https://towardsdatascience.com/yet-another-text-generation-project-5cfb59b26255

```
[I, Donald J. Trump, president of the United ]States. Were going to
have you to say what happens to try to make America great again.
We're going to tell you that we have to win. We owe $100.
```

```
[Despite the constant negative press cov]erage. I was going to be
great. But if they're going to stop the good guy who has a big
night.

[Despite the constant negative press cov]erage. Looking forward to
it. The fundraisers are totally promised by the Central Park
South...

[Despite the constant negative press cov]erage lines to Trump Tower.
That's not the relationships. It doesn't have to be a disaster to
Iraq.

[Despite the constant negative press cov]erage car millions of
people and spot. We are going to come back. We have to go to golf.
```

Applications of Neural Networks

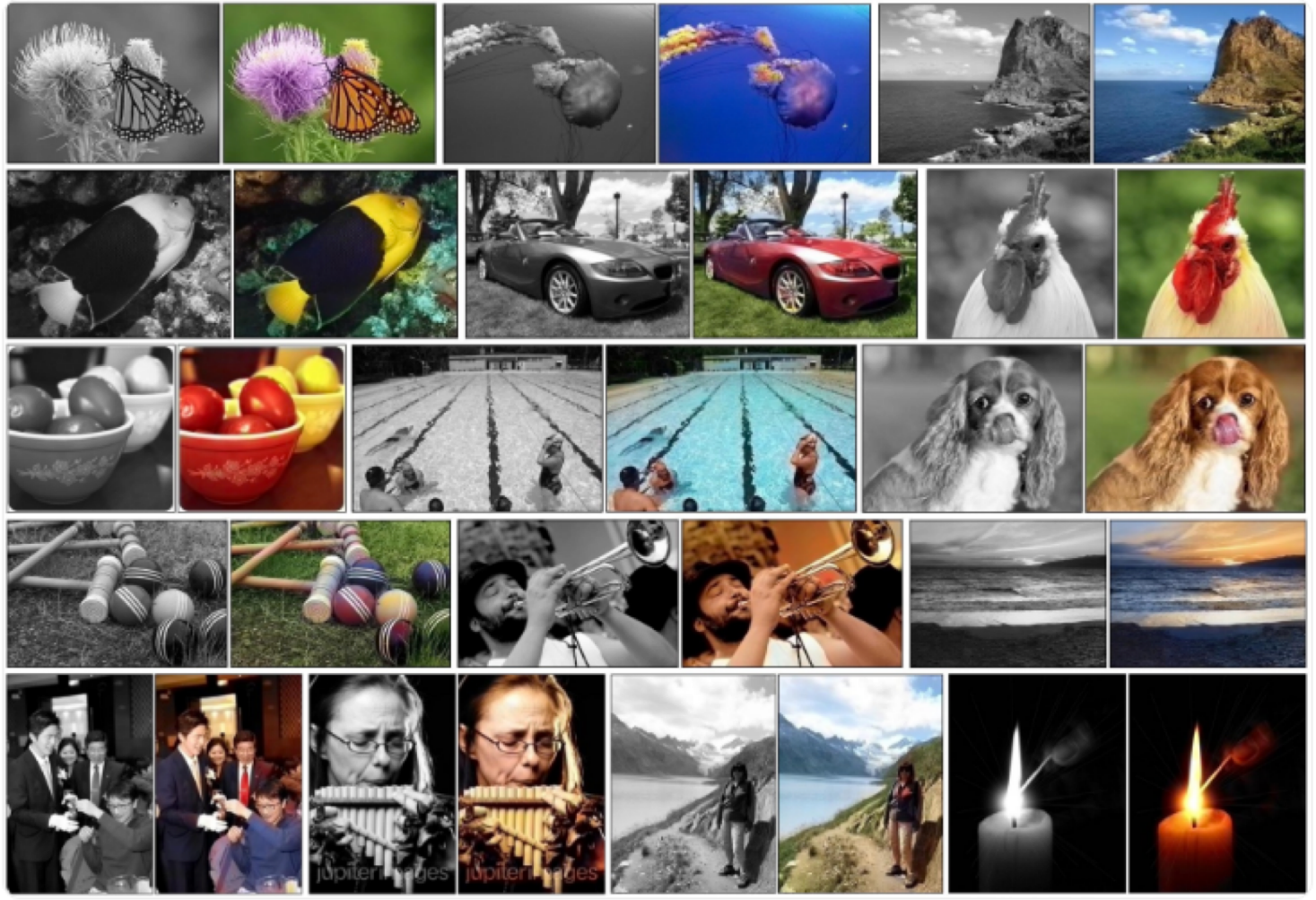
HARRY POTTER: WRITTEN BY ARTIFICIAL INTELLIGENCE

"The Malfoys!" said Hermione.
Harry was watching him. He looked like Madame Maxime. When she strode up the wrong staircase to visit himself.
"I'm afraid I've definitely been suspended from power, no chance — indeed?" said Snape. He put his head back behind them and read groups as they crossed a corner and fluttered down onto their ink lamp, and picked up his spoon. The doorbell rang. It was a lot cleaner down in London.
Hermione yelled. The party must be thrown by Krum, of course.
Harry collected fingers once more, with Malfoy. "Why, didn't she never tell me. …" She vanished. And then, Ron, Harry noticed, was nearly right.
"Now, be off," said Sirius, "I can't trace a new voice."
He punished Uncle Vernon, so loudly that she could barely use it with Viktor Krum, these faces in one side of their previous year. Mr. Dursley again was this champion. He was about to give the Remembrall there…. every time he …..

https://medium.com/deep-writing/harry-potter-written-by-artificial-intelligence-8a9431803da6

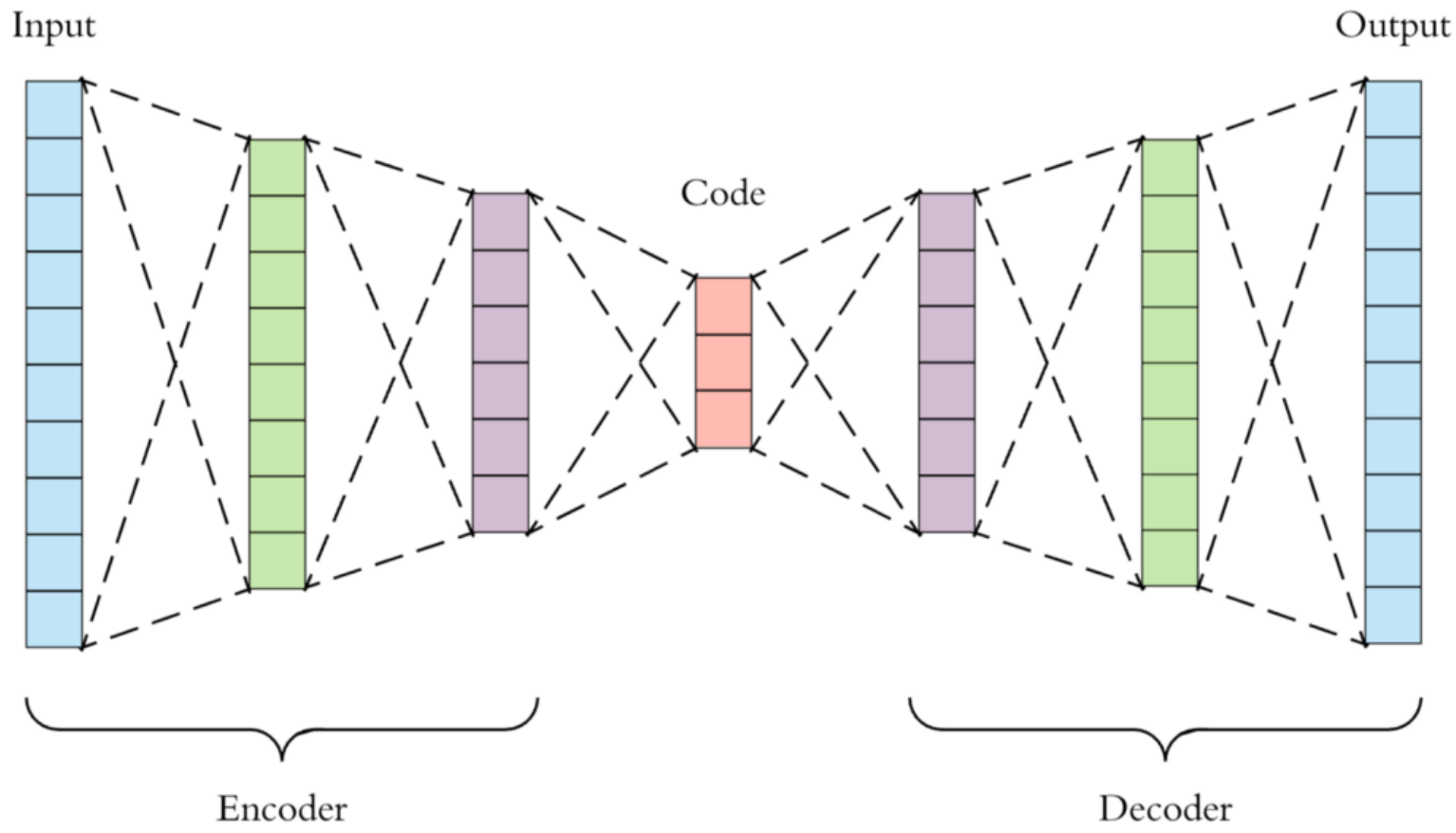
Applications of Neural Networks

AUTOMATIC COLORIZATION OF BLACK AND WHITE IMAGES



Applications of Neural Networks

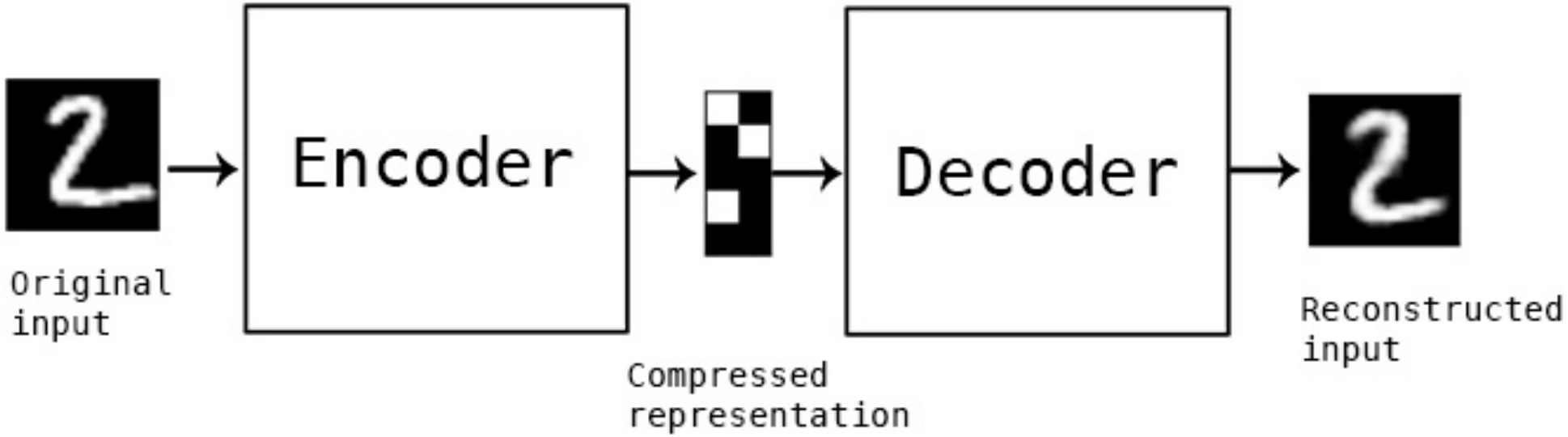
AUTO-ENCONDERS



[Goodfellow et al.]

Applications of Neural Networks

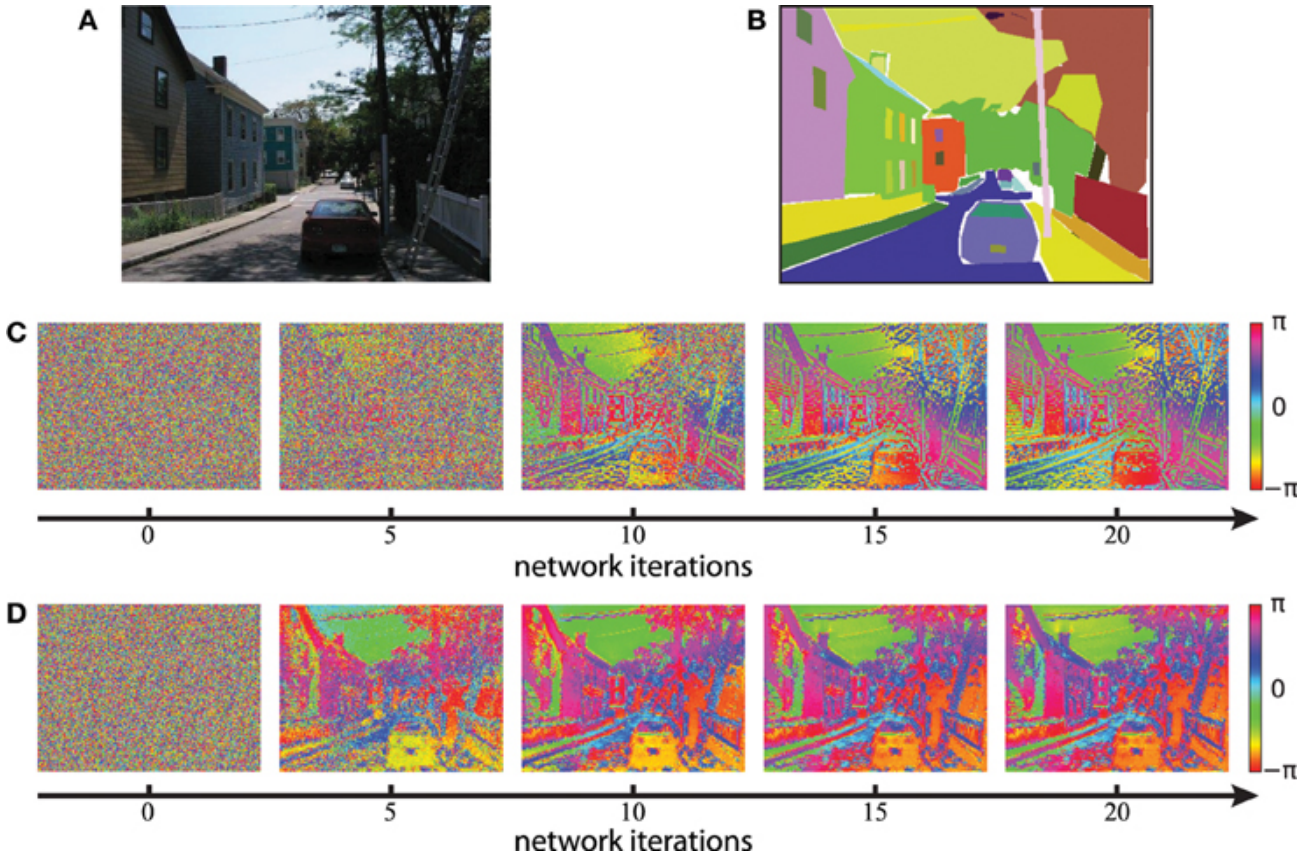
AUTO-ENCONDERS



[Goodfellow et al.]

Applications of Neural Networks

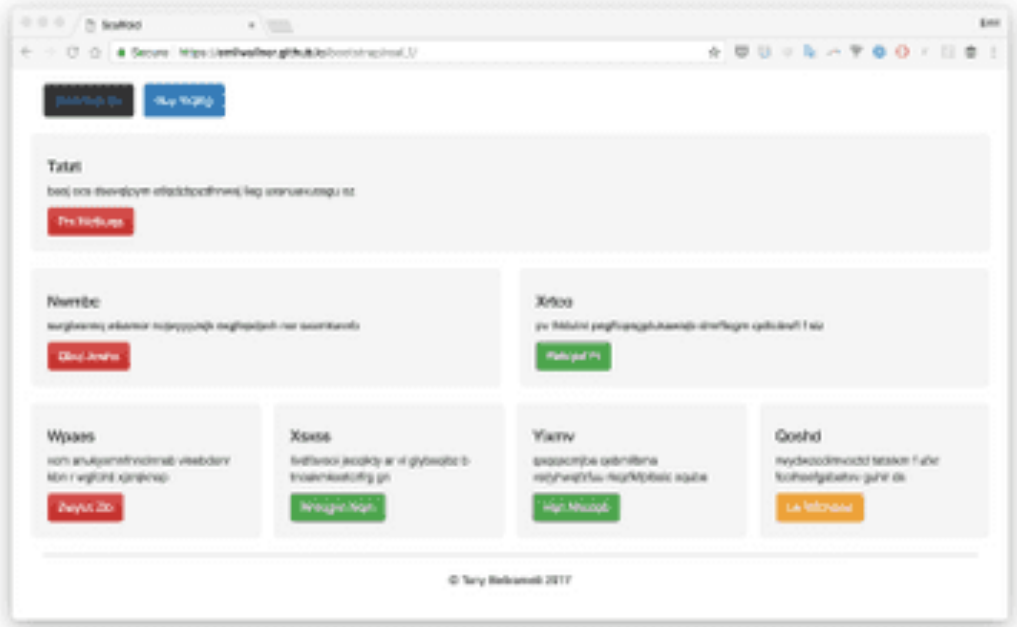
AUTO-ENCONDERS – IMAGE SEGMENTATIONS



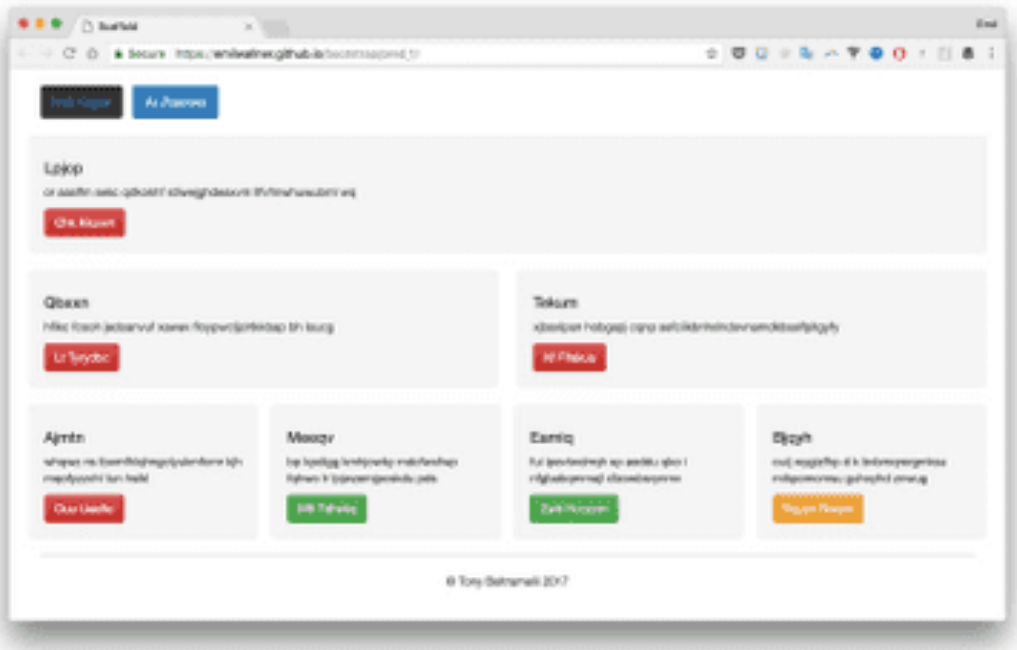
[image from https://www.frontiersin.org]

Applications of Neural Networks

CODE GENERATION



Original



Prediction

https://blog.floydhub.com/turning-design-mockups-into-code-with-deep-learning

Applications of Neural Networks

OSCAR PREDICTION 2018

Big ML was able to predict all the winners on 2018 Oscars using:

- details about the film (duration, budget, genre, etc.),
- IMDB ratings
- nominations in previous awards (Golden Globes, BAFTA, Screen Actors Guild and Critics Choice) from 2000 to 2017 used in last year's predictions.

Let’s make things happen!

Languages in Machine Learning

Machine learning frameworks

PYTHON LIBRARIES

Machine learning frameworks

LIBRARIES

scikits learn
machine learning in Python

H_2O.ai

PySpark

XGBoost

LightGBM

NLTK

TextBlob

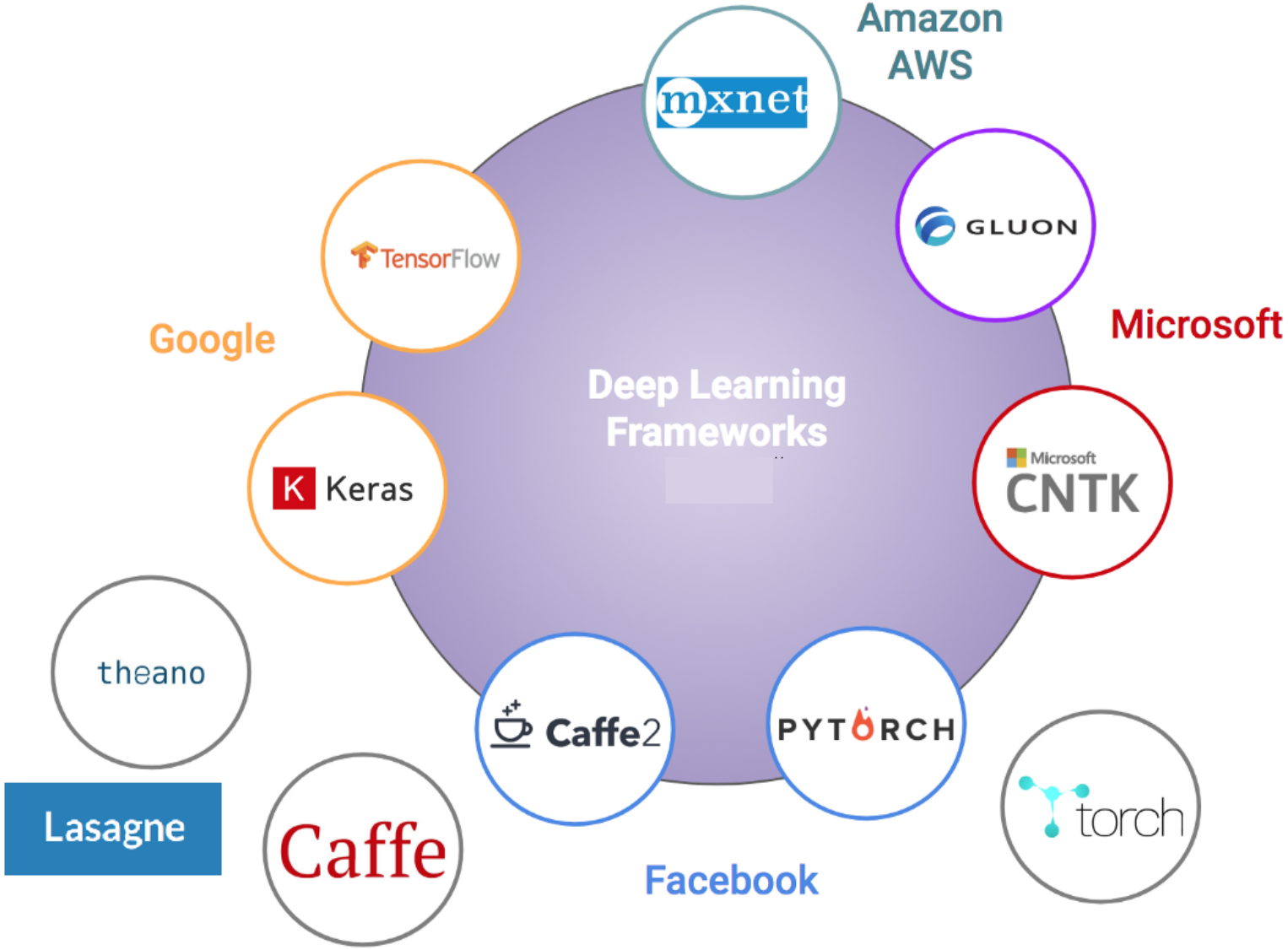
gensim

Standford NLP

S NLP

WEKA
The University of Waikato

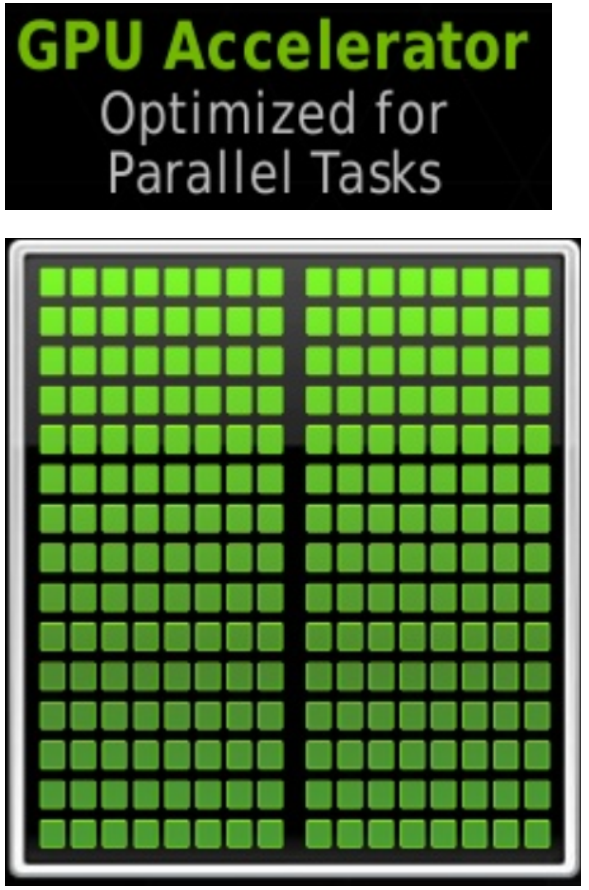
Deep learning frameworks



Cloud platforms

GPU programming

GPU may provide more than 10x performance and 5x energy efficiency

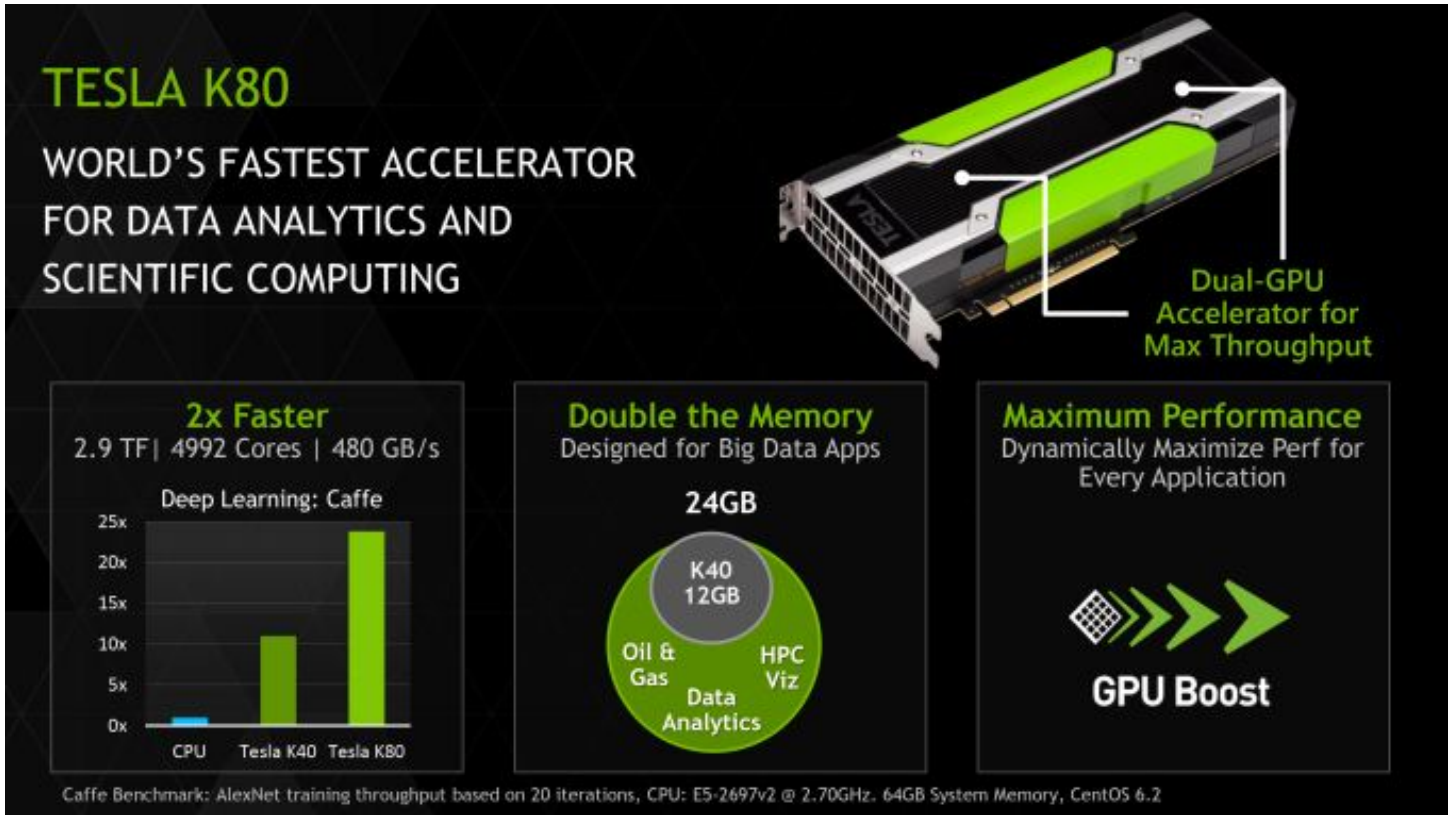


GPU cards

TESLA K80

Specs:
24GB GDDR5 CUDA
4992 processors

Launch price: $5000



[images and content from www.nvidia.com]

GPU cards

GTX 1080

Specs

VRAM: 8 GB

Memory bandwidth: 320 GBs/second

Processing power: 2560 cores @ 1733 MHz

GTX 1080 TI

Specs

VRAM: 11 GB

Memory bandwidth: 484 GBs/sec

Processing power: 3584 cores @ 1582 MHz

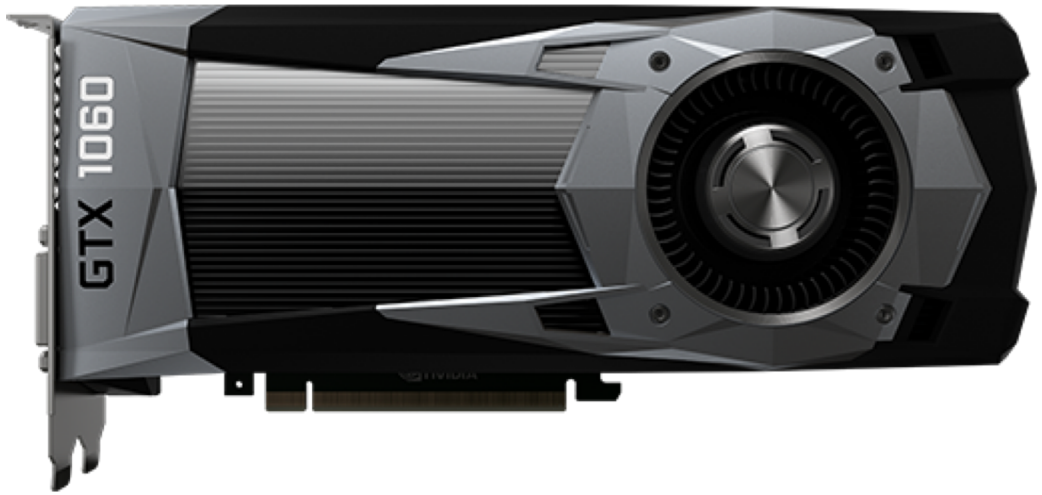
[images from www.nvidia.com]

GPU cards

GTX 1060 / 1050 TI (6 / 4 GB)

Specs

Processing power: 1280 cores @ 1708 MHz (~ 2,19 M CUDA Core Clocks)



[images from www.nvidia.com]

GPU cards

- I work with data sets > 250GB: K80 or multiple GTX 1080 Ti.
- I have little money: GTX 1060 (6GB).
- I have almost no money: GTX 1050 Ti (4GB).
- I do Kaggle: GTX 1060 (6GB) for any “normal” competition, or GTX 1080 Ti for “deep learning competitions”.
- I am a competitive computer vision researcher: NVIDIA Titan Xp or at least few GTX 1080 Ti.

Learn resources

Udemy

coursera

UDACITY

kaggle

Medium

root:~$Siraj Raval

Artificial Intelligence Education

New Video Every Week!

Measuring performance in Machine Learning

Measure the performance of your system

TWO CLASSES

Actual

Predicted	TRUE	FALSE
TRUE	True positive (TP)	False pozitive (FP)
FALSE	False negative (FN)	True negative (TN)

$$Accuracy = \frac{TP+TN}{TP+TN+FN+FP}$$

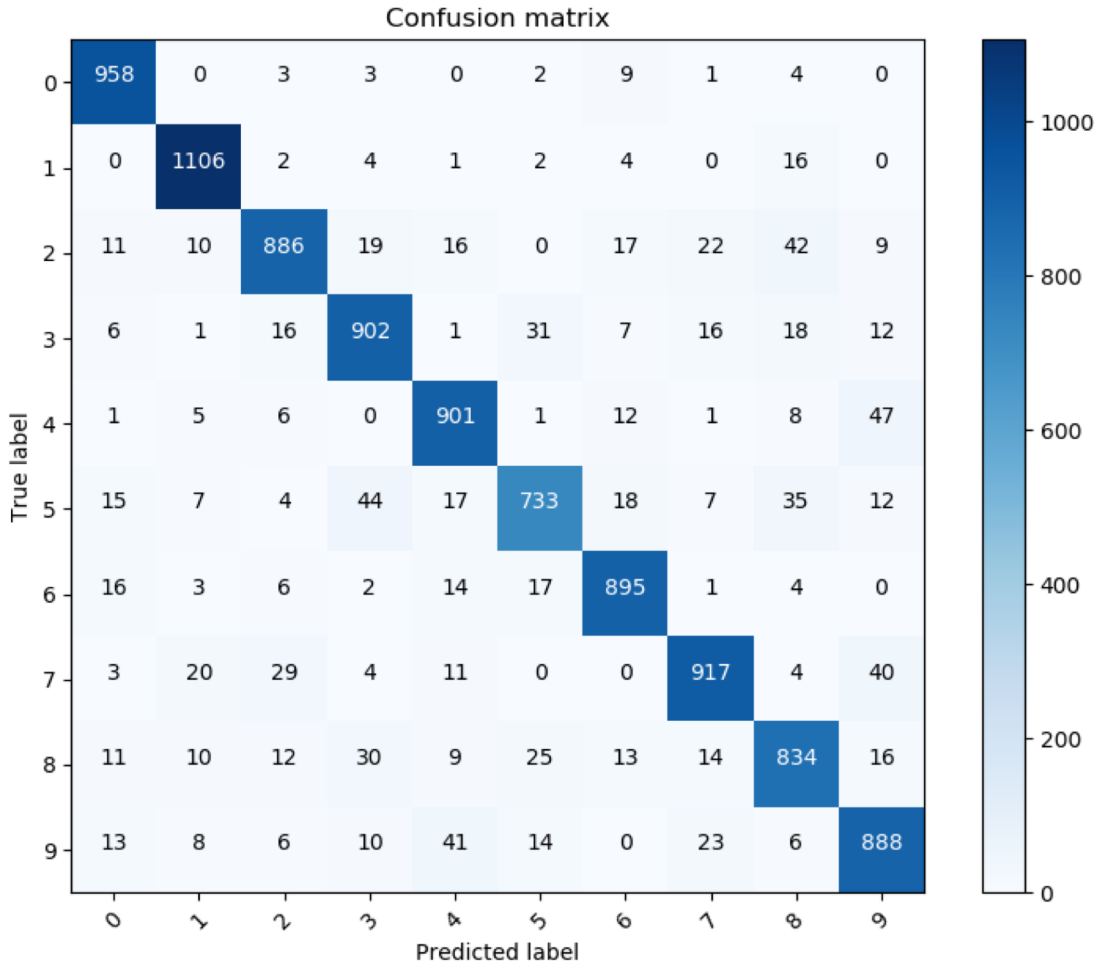
$$\text{Precision} = \frac{TP}{TP+FP} \qquad \text{Recall} = \frac{TP}{TP+FN}$$

Are these parameters OK for every measure?

What is more important: precision or recall?

Measure the performance of your system

CONFUSION MATRIX



Measure the performance of your system

Ex: fraud detection

Predicted \ Actual	TRUE	FALSE
TRUE	3	97
FALSE	0	0

Recall = 100%

Precision = 3%

~~Mean = (P+R)/2 = 51.5%~~

$$\text{F1 score} = \frac{2 * Precision * Recall}{Precision + Recall}$$

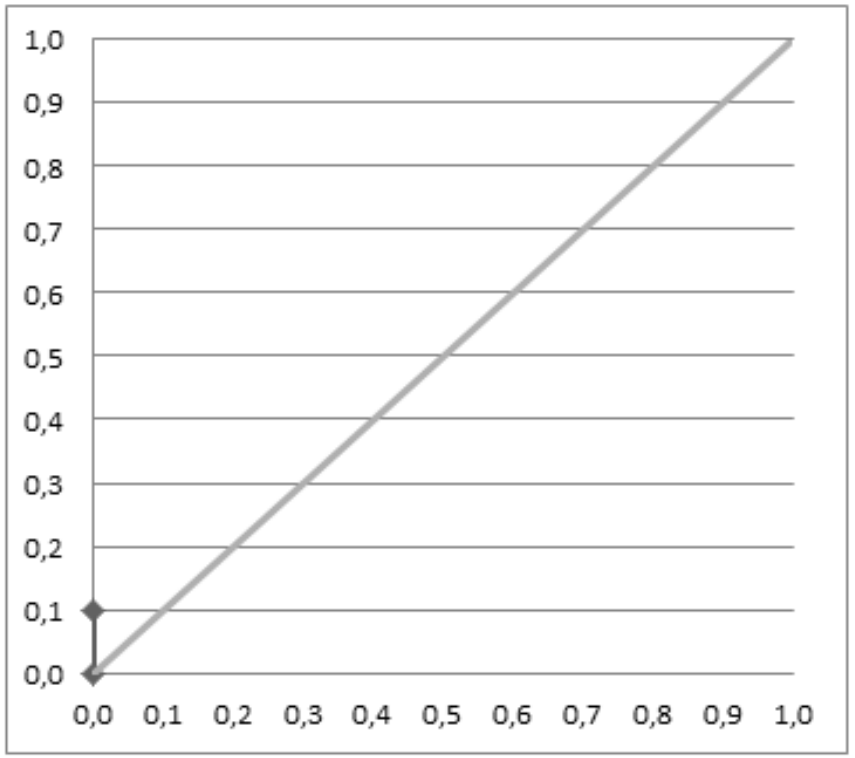
Measure the performance of your system

ROC CURVES

ROC Curves are used to see how well your classifier can separate positive and negative examples and to identify the best threshold for separating them.

To be able to use the ROC curve, your classifier has to be *ranking* - that is, it should be able to rank examples such that the ones with higher rank are more likely to be positive.

#	C	Score
1	P	0,9
2	P	0,8
3	N	0,7
4	P	0,6
5	P	0,55
6	P	0,54
7	N	0,53
8	N	0,52
9	P	0,51
10	N	0,505
11	P	0,4
12	N	0,39
13	P	0,38
14	N	0,37
15	N	0,36
16	N	0,35
17	P	0,34
18	N	0,33
19	P	0,3
20	N	0,1

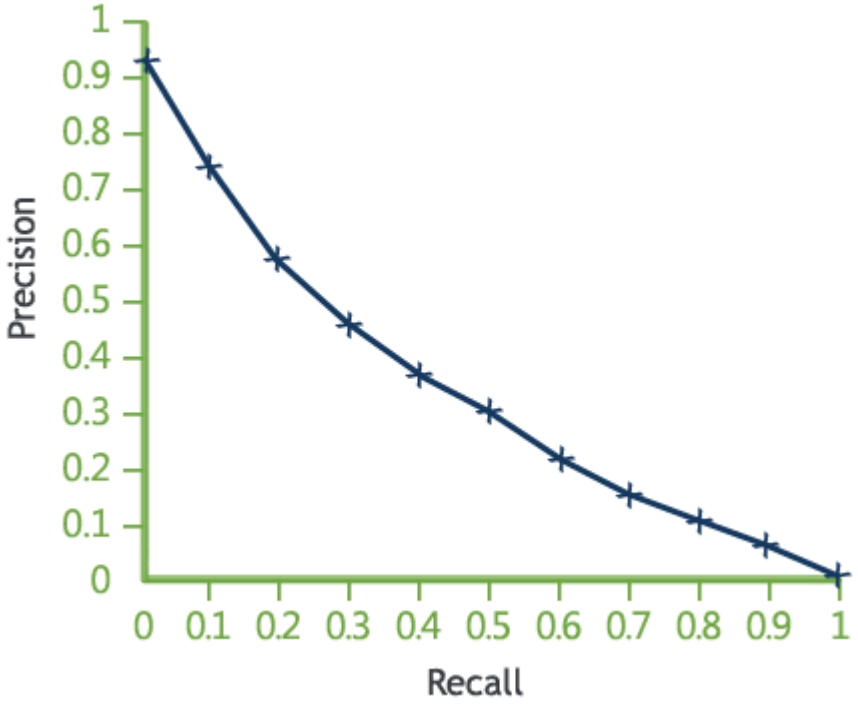


[Image from habrastorage.org]

Measure the performance of your system

PRECISION / RECALL CURVES

- If we retrieve more document, we improve recall (if return all docs, R=1)
- if we retrieve fewer documents, we improve precision, but reduce recall
- so there's a trade-off between them

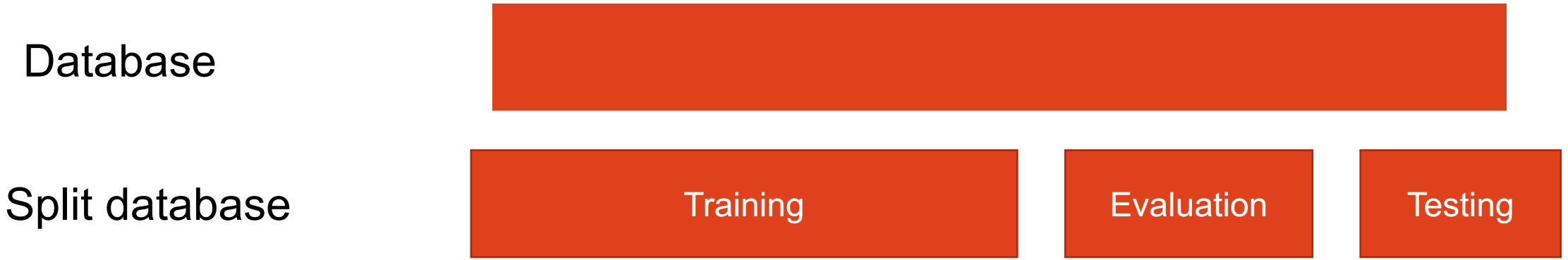


- The are under the curve is also known as Mean Average Precision

[Image from habrastorage.org]

Measure the performance of your system

MODEL EVALUATION TECHNIQUES



- Training set consists of records with known class labels used to build a classification model
- Evaluation set is used to measure the performance of the system
- A labeled test set of previously unseen data records is used to evaluate the quality of the model.

Measure the performance of your system

K-FOLDS (CROSS VALIDATION)

Combining more algorithms together

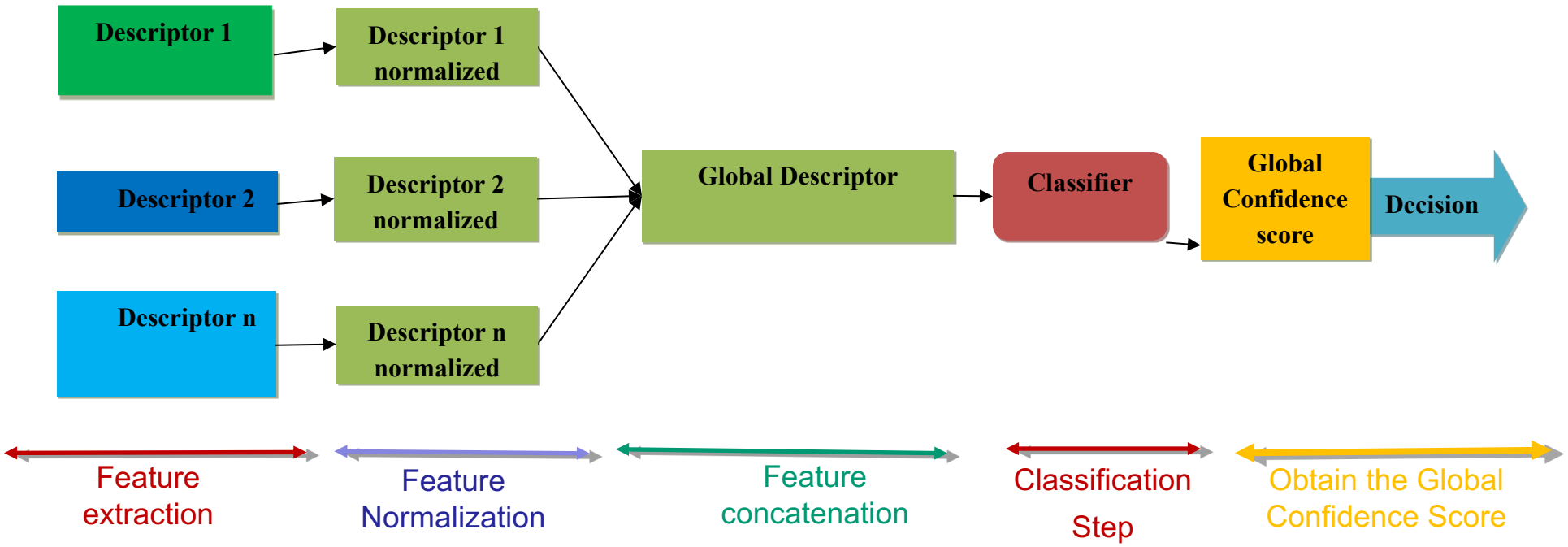
- Ensemble methods are based around the hypothesis that an aggregated decision from multiple experts can be superior to a decision from a single system.

- More accurate ONLY if the individual classifiers disagree.

more info on:
[http://ionut.mironica.ro/publications/CBMI2013_Mironica_CameraReady.pdf]

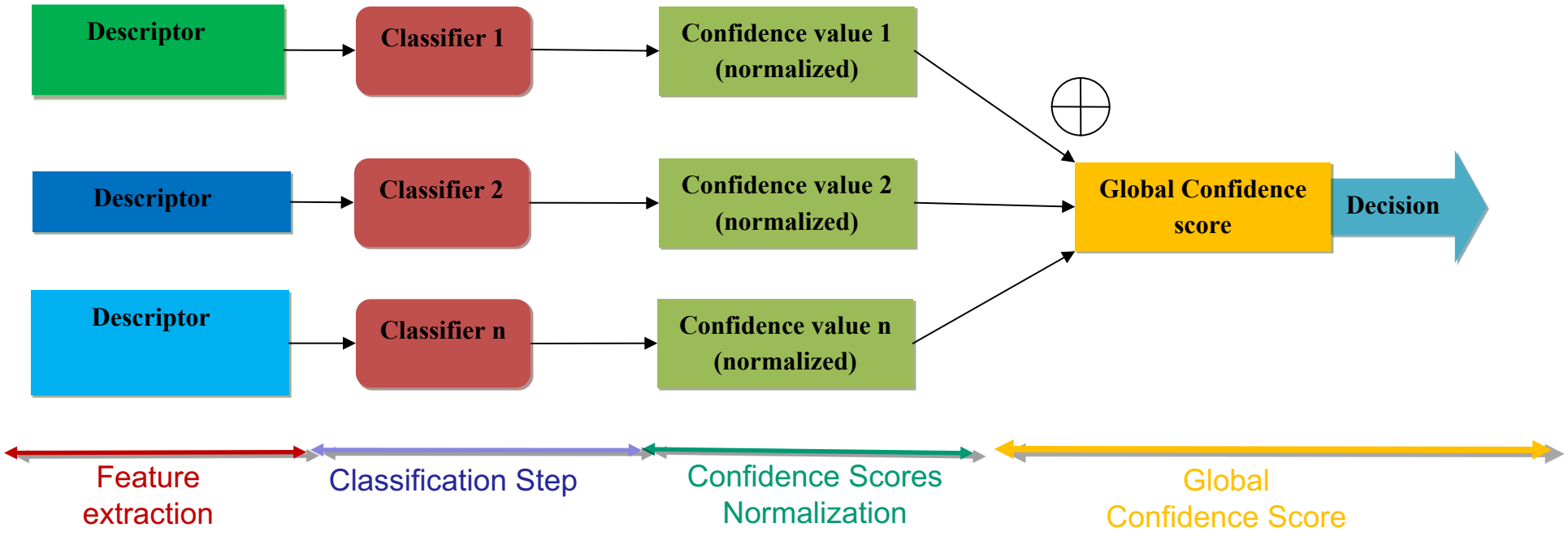
Combining more algorithms together

Early Fusion



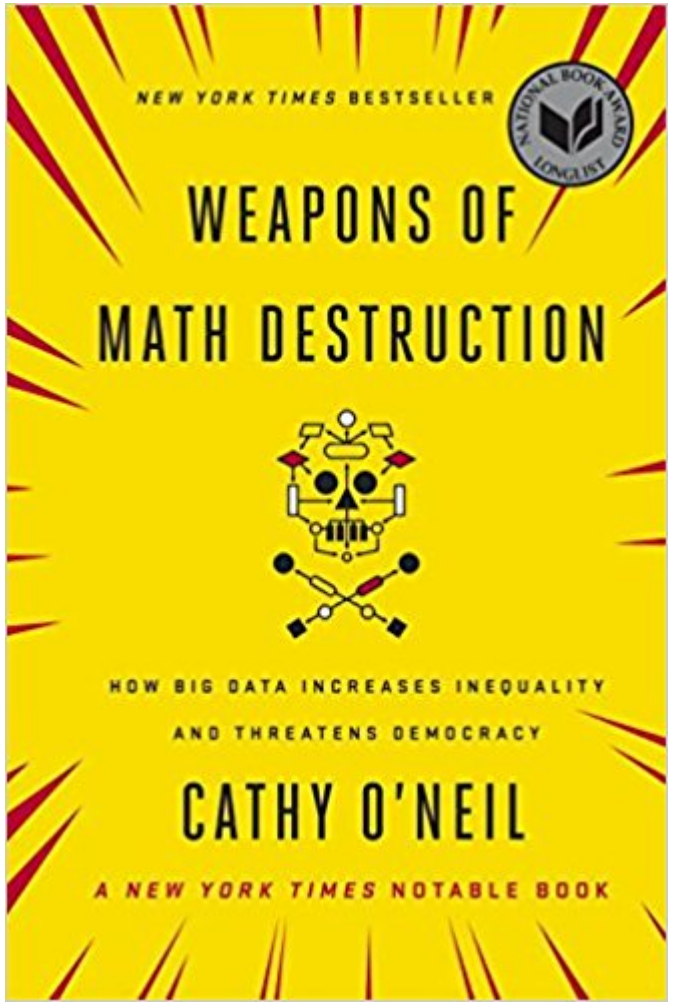
Combining more algorithms together

Late Fusion



Hard voting vs Soft Voting

Ethics in Machine Learning



- Big Data Increases Inequality and Threatens Democracy, pointed out that predictive analytics based on algorithms tend to punish the poor, using algorithmic hiring practices, insurance and credit risk analysis, bespoke offers and advertisement.
- Prevent using discriminatory information: sex, religion, race, ideology
- How many jobs will disappear because of AI?
- A Hippocratic oath for AI developers?
- Is the accuracy principle equal to equity principle?

Conclusion

[image from http://www.quotehd.com]

Conclusion

- No free lunch: machine learning algorithms are tools with advantages and disadvantages.

- First try the simple classifiers. If they don't map well on the application then other more complicated models can be used.

- It is better to have simple classifiers and more intelligent data than complex classifiers and simple data.

- Use complex classifiers when we have diverse and/or large data set.

PART 2

Hands-on time

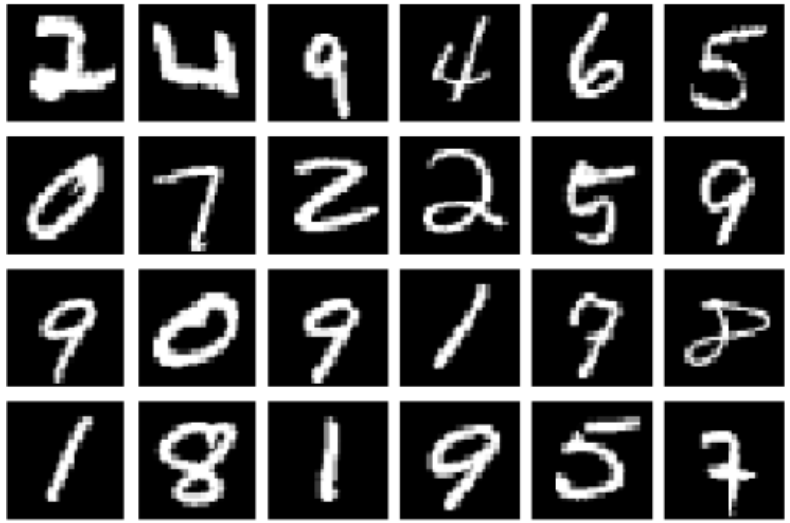
Exercise 1:

https://github.com/imironica/keras-without-a-phd

Demo details

MNIST DIGIT DATASET

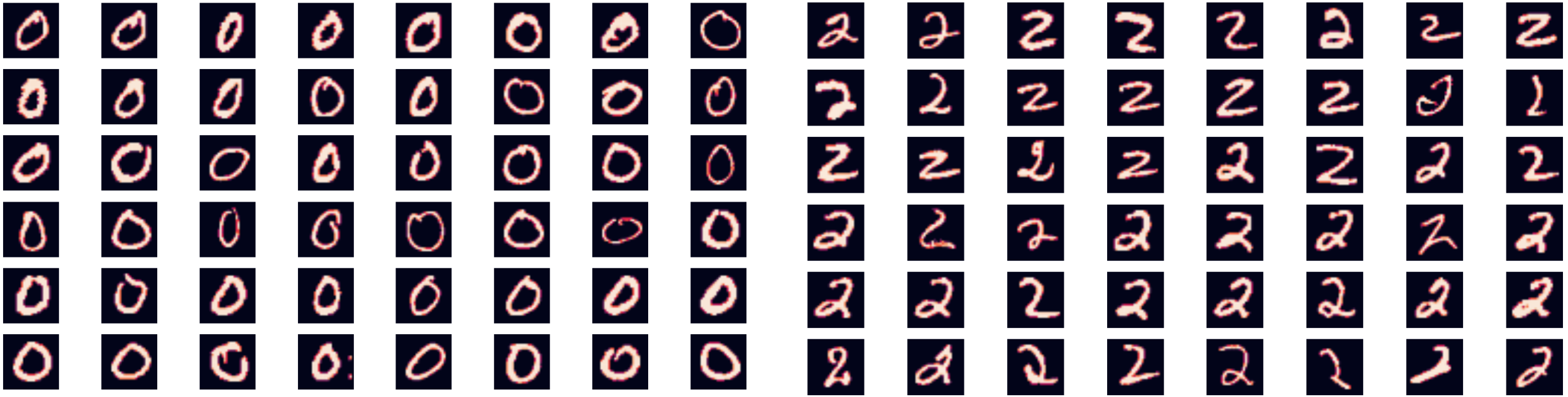
- The MNIST database of handwritten digits, available from NIST
- Training set: 60,000 image of digits of size 28x28
- Test set: 10,000 images



[more details on http://yann.lecun.com/exdb/mnist/]

Demo details

MNIST DIGIT DATASET



[more details on http://yann.lecun.com/exdb/mnist/]

Hands-on time

User / password: deeplearning

cd /home/deeplearning/work/keras-without-a-phd

sudo mkdir keras-without-a-phd
cd keras-without-a-phd
git clone https://github.com/imironica/keras-without-a-phd.git

sudo python3 0.1_visualizeDataset.py –v 1

Hands-on time

Exercise 2:

https://github.com/imironica/traffic-signs-keras

Demo details

TRAFFIC SIGN RECOGNITION

Demo details

TRAFFIC SIGN RECOGNITION

THANK YOU

Ionuț Mironică

ionut.mironica

ionut.mironica

http://ionut.mironica.ro

GitHub imironica