UNIVERSITÀ DEGLI STUDI DI MILANO

Artificial Intelligence

History of AI

The field of AI research was born at a workshop at Dartmouth College in 1956 where the term "Artificial Intelligence" was coined by John McCarthy to avoid restricting the term to fields such as cybernetics.

At the workshop, researchers from several disciplines met to clarify, define ideas and establish a research program concerning “**thinking machines**”.

They chose the name “Artificial Intelligence” for its broad sense, to avoid restricting the interests of this field to subjects such as cybernetics, automata theory and complex information processing.

“Dartmouth Summer Research Project on Artificial Intelligence” is now considered by many [1], [2] the seminal event where Artificial Intelligence (AI) was officially declared a research field.

[1] R. J. Solomonoff, "Artificial intelligence social effects future developments", *Hum. Syst. Manage.*, vol. 32, pp. 149-153, 1985.

[2] J. Moore, "The dartmouth college artificial intelligence conference: The next fifty years", *AI Mag.*, vol. 27, no. 4, pp. 87-91, 2006.

DEFINITIONS

Today, AI:

(Merriam-Webster Dictionary:) “concerns the theory and development of computerized systems able to imitate and simulate human intelligence and behavior”

essentially “being human-like rather than becoming human” [3], and

(English Oxford Living Dictionary:) “performing tasks normally requiring human intelligence, such as visual perception, speech recognition, decision-making, and translation between languages”

[3] B. Marr, The Key Definitions Of Artificial Intelligence (AI) That Explain Its Importance, Feb. 2018, [online] Available: https://www.tobepublishedforbes.com/sites/bernardmarr/2018/02/14/the-key-definitions-of-artificial-intelligence-ai-that-explain-its-importance.

Computer science studies defitions

[4,5,6,7] Define AI researches (called “computational intelligence” researches by some authors [4]):

studies about “intelligent agents” or “rational agents [5]”: any device that perceives its environment and takes actions that maximize its chance of successfully achieving its goals.

[4] Poole D. and Mackworth A. and Goebel, R. (1998). “Computational Intelligence: A Logical Approach”. New York: Oxford University Press. ISBN 978-0-19-510270-3.

[5] Russell, Stuart J.; Norvig, Peter (2003), Artificial Intelligence: A Modern Approach (2nd ed.), Upper Saddle River, New Jersey: Prentice Hall, ISBN 0-13-790395-2.

They use the term rational agents instead of intelligent systems and write (page. 55): "The whole-agent view is now widely accepted in the field"

[6] Jackson, Philip (1985): Introduction to Artificial Intelligence (2nd ed.). Dover. ISBN 978-0-486-24864-6.

[7] Legg S. and Hutter M. (15 June 2007). A Collection of Definitions of Intelligence (Technical report). IDSIA. arXiv:0706.3639. Bibcode:2007arXiv0706.3639L. 07-07.

In [8] AI is defined in a way that seems introducing the field of machine learning:

a (computational agent) “system's ability to correctly interpret external data, to learn from such data, and to use those learnings to achieve specific goals and tasks through flexible adaptation”.

[8] Kaplan A. and Haenlein M. (2019), "Siri, Siri, in my hand: Who's the fairest in the land? On the interpretations, illustrations, and implications of artificial intelligence". *Business Horizons*. **62** (1): 15–25. doi:10.1016/j.bushor.2018.08.004

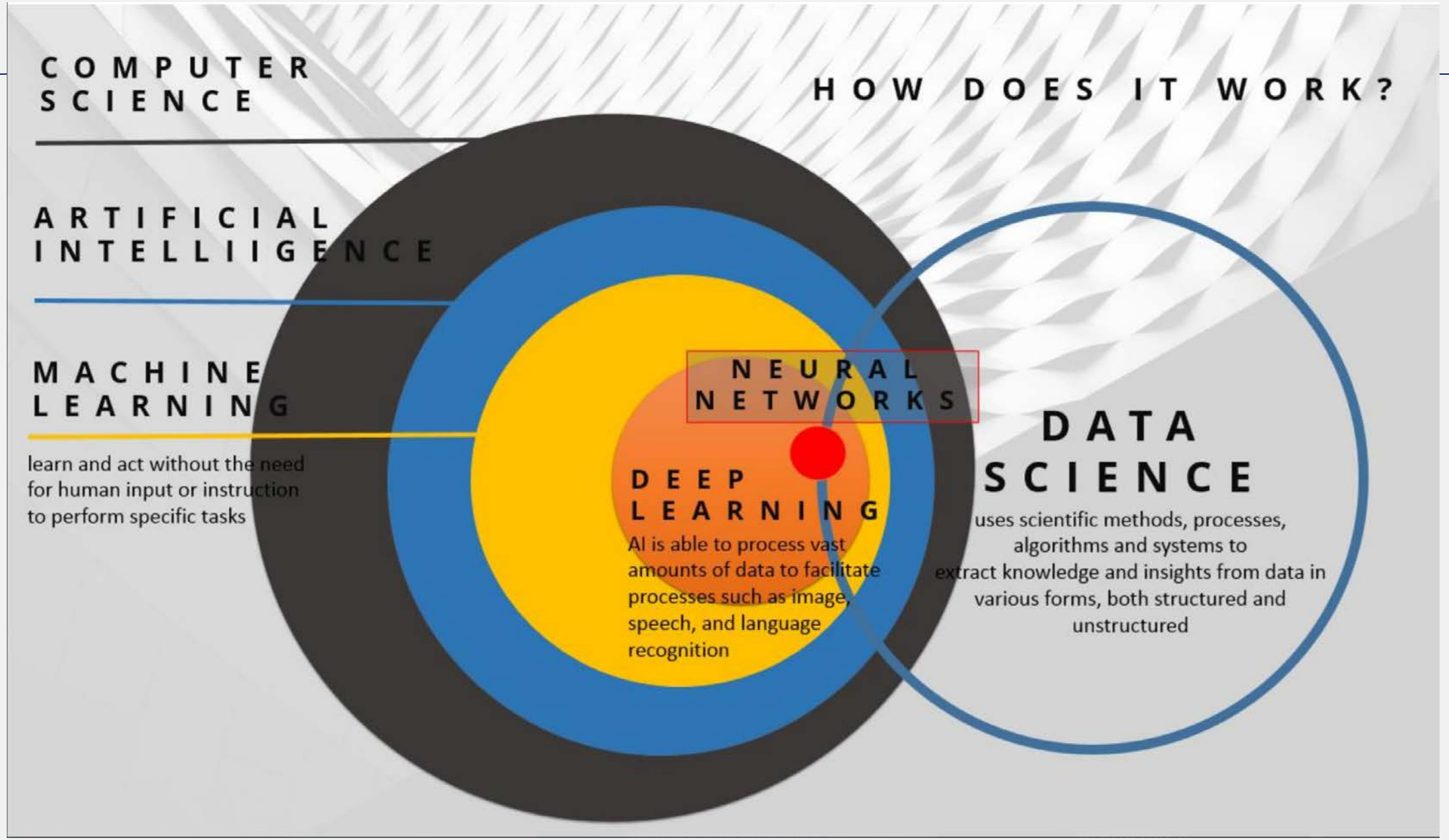
“Artificial intelligence (AI) can be described as the ability of a computer or robot-controlled computer to perform tasks that are commonly associated with intelligent creatures”

“scientific discipline that involves building computer systems whose behavior can be interpreted intelligently”

Catching-up

Artificial intelligence (AI), sometimes called machine intelligence, is intelligence implemented into, and demonstrated by, machines

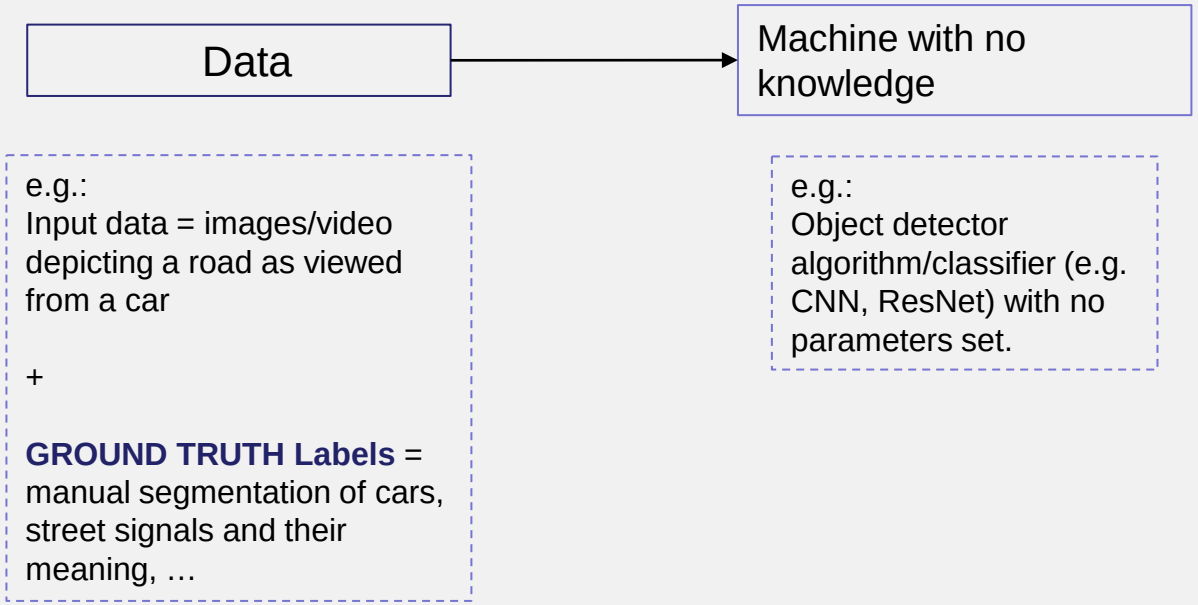
It differs from Natural intelligence, which is the one displayed by humans and other animals



Machine Learning (Supervised learning)

AI is formed through learning (by viewing examples).

Once the machine has learned it can view novel (unknown – never seen) data to generate its own opinions (in the form of predictions or classifications).



Data is split into Training and Test sets

Training set:

Training Set

Validation Set

Machine training: learning algorithms choose the best machine setting

Learning refers to choosing the algorithm parameters that allow achieving the best performance on both the training and the validation set

Test Set:

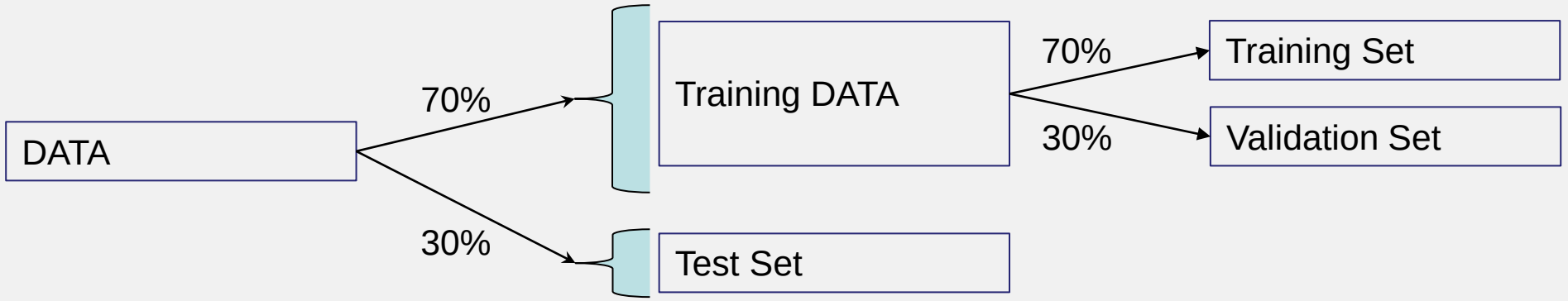
Test Set

AI (trained) Machine Testing

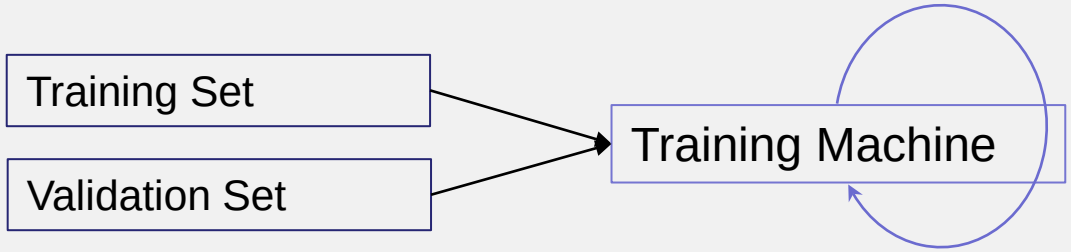
Testing refers to the application of the trained machine to unseen data contained in the test set.

Training data is further slit into:

- Training set (for training the machine through iterative methods)
- Validation set (for validating the machine and interrupt learning)



(Supervised) Training: a simplified training algorithm



FOR Epoch = 1:N

The algorithm executes N iterations (epochs); generally N = 1000

- For each subsets of the training set:

In each epoch, all the subsets composing the training set are analyzed.

  - compute predictions for all the points in the subset

Predictions are computed by using the machine at its current status

  - measure the prediction error

Remember the training set contains the ground truth labels

  - Adapts the algorithm parameters to diminish the error

Different techniques for doing so..

END

- compute prediction for all the examples in the validation set and evaluate the error in the validation set.
- IF the error on the validation set grows or hasn't been changing for a while TRAINING STOPS;

  ELSE training continues with the next epoch

END

(Supervised) Training

We call it supervised learning because our example have GROUND TRUTH labels

Supervised learning is applied to:

- Neural models (multilayer perceptrons - MLPs, feed forward neural networks – FFNNs)
- Deep neural network
- Support Vector Machines (SVMs)
- Bayesian Trees (BTs) and Random Forests (RFs)
- ….

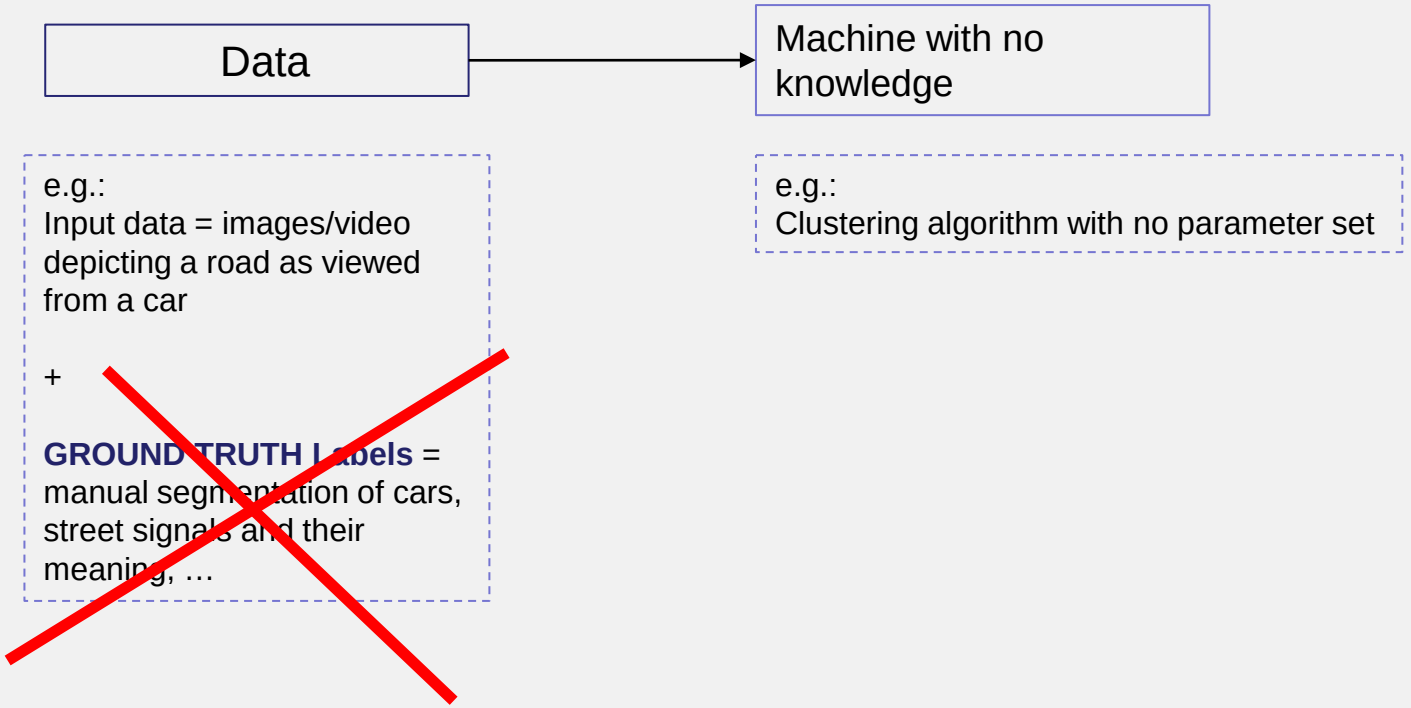
And many other (combinations) of them

https://github.com/LucaCappelletti94/bioinformatics_practice

Machine Learning (UNSUPERVISED learning)

AI is formed through learning (by viewing examples WITHOUT GROUND TRUTH LABELS).

Once the machine has learned it can view novel (unknown – never seen) data to generate its own opinions (in the form of predictions or classifications).



AGAIN

The dataset is split

The algorithm parameters are estimated on the training set

The algorithm is tested on the test set

Again Data is split into Training and Test sets

Training set:

Training Set

Validation Set

Machine training: learning algorithms choose the best machine setting

Learning refers to choosing the algorithm parameters that allow achieving the best performance on both the training and the validation set

Test Set:

Test Set

AI (trained) Machine Testing

Testing refers to the application of the trained machine to unseen data contained in the test set.

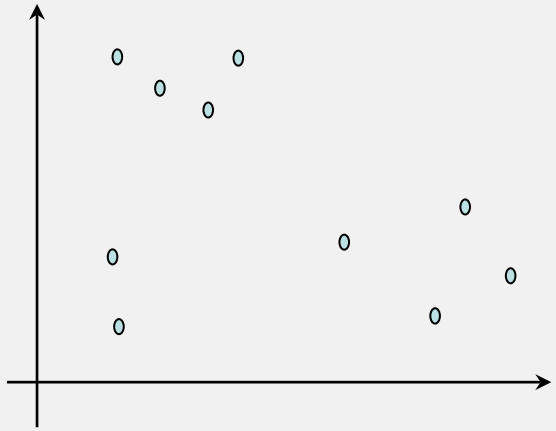
Here training requires setting the parameters of, e.g clustering algorithms.

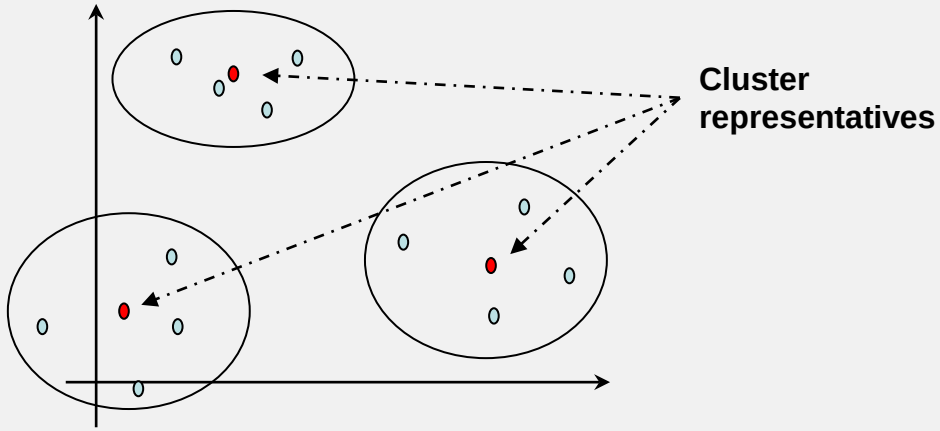
Clustering algorithms analyze the data to form groups.

Examples, Arbib, k-means.

Example of parameters required:

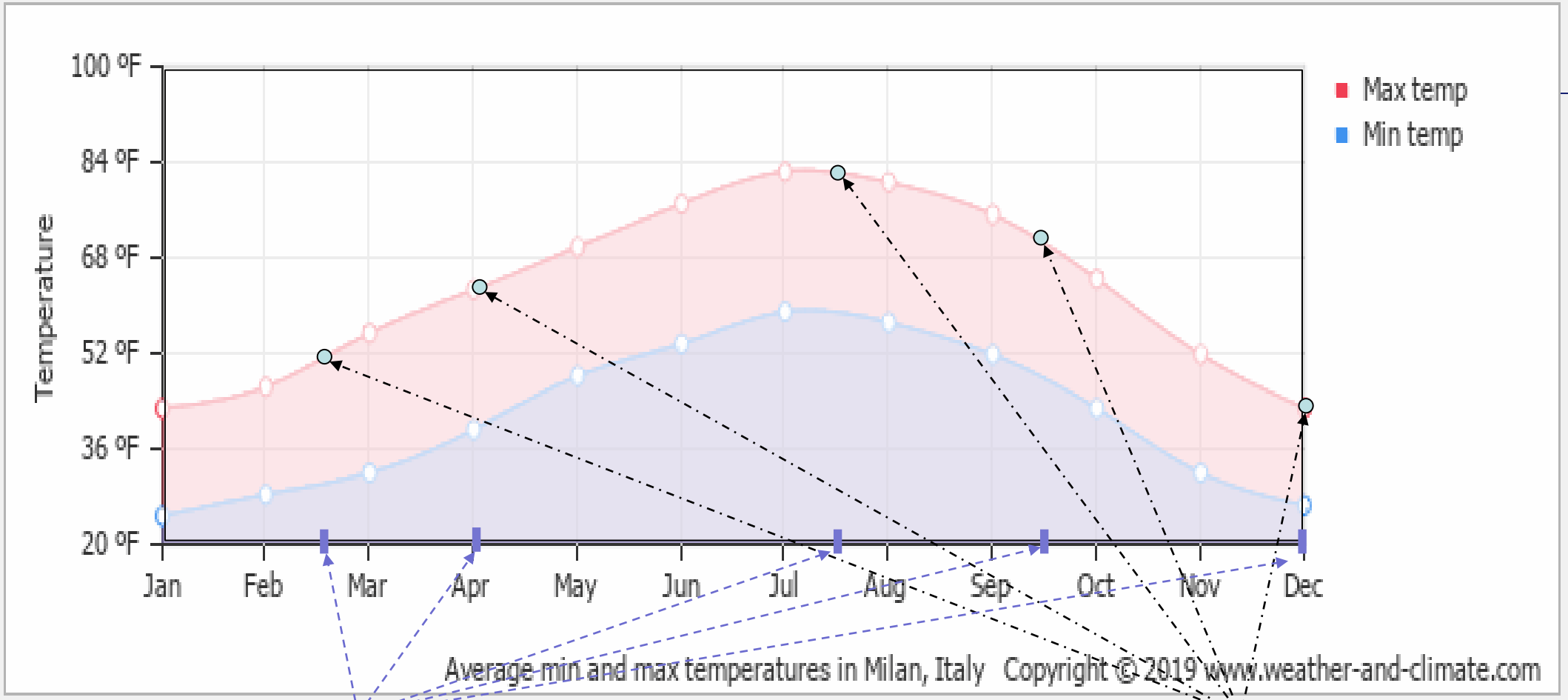
- number of clusters to search for
- cluster initialization
- learning rate





When the machine learns how to predict classes for each input data we call it Classifier.
Learning a Classification “means” somehow learning a data grouping.

When the machine must infer a continuous number from the data, we call it regressor (regression algortihms are used for training).
Learning a Regression model means learning a function which associate to each input point e function.
e.g.: given date, learn the expected temperature in Lombardy (Italy)



Points:
17th February
2nd of April
15th of July
13th of September
31st of December

Labels (for each point are the correct temperature estimates for each day)
52° F
60° F
80° F
70° F
45° F

Today we experience with a genomic classification problem

https://github.com/LucaCappelletti94/bioinformatics_practice

Points: genome sequence

4 classes: Active Enhancers, Active Promoters, Inactive Enhancers, Inactive Promoters

DNA

Genes influence what we look like on the outside and how we work on the inside.
They contain the information our bodies need to make chemicals called proteins.
Proteins form the structure of our bodies, as well playing an important role in the processes that keep us alive.

Genes are made of a molecule (chemical) called **DNA**, which is short for 'deoxyribonucleic acid'.
The **DNA molecule is a double helix**: that is, two long, thin strands twisted around each other like a spiral staircase.

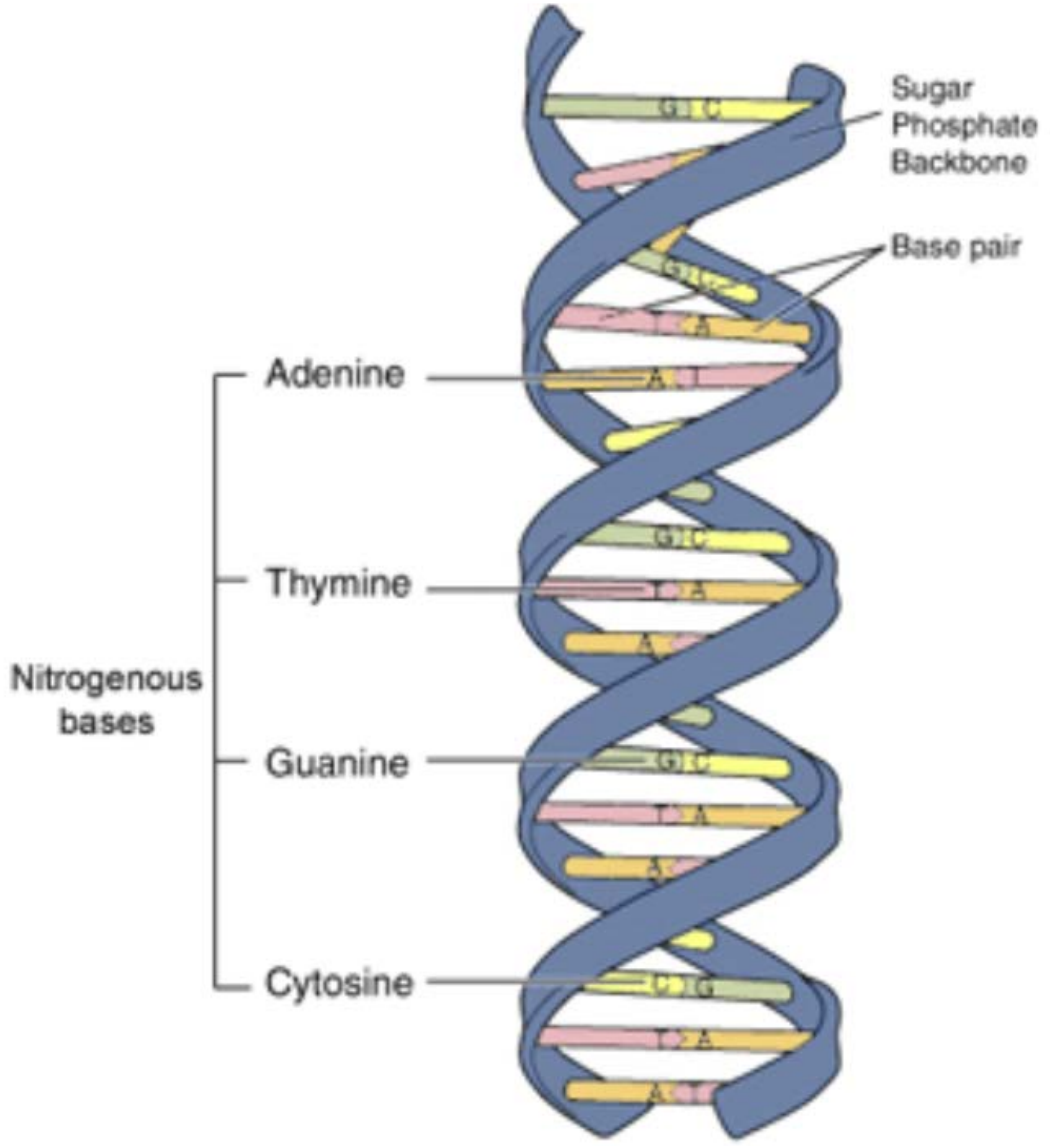


Image adapted from: National Human Genome Research Institute.

The DNA double helix showing base pairs

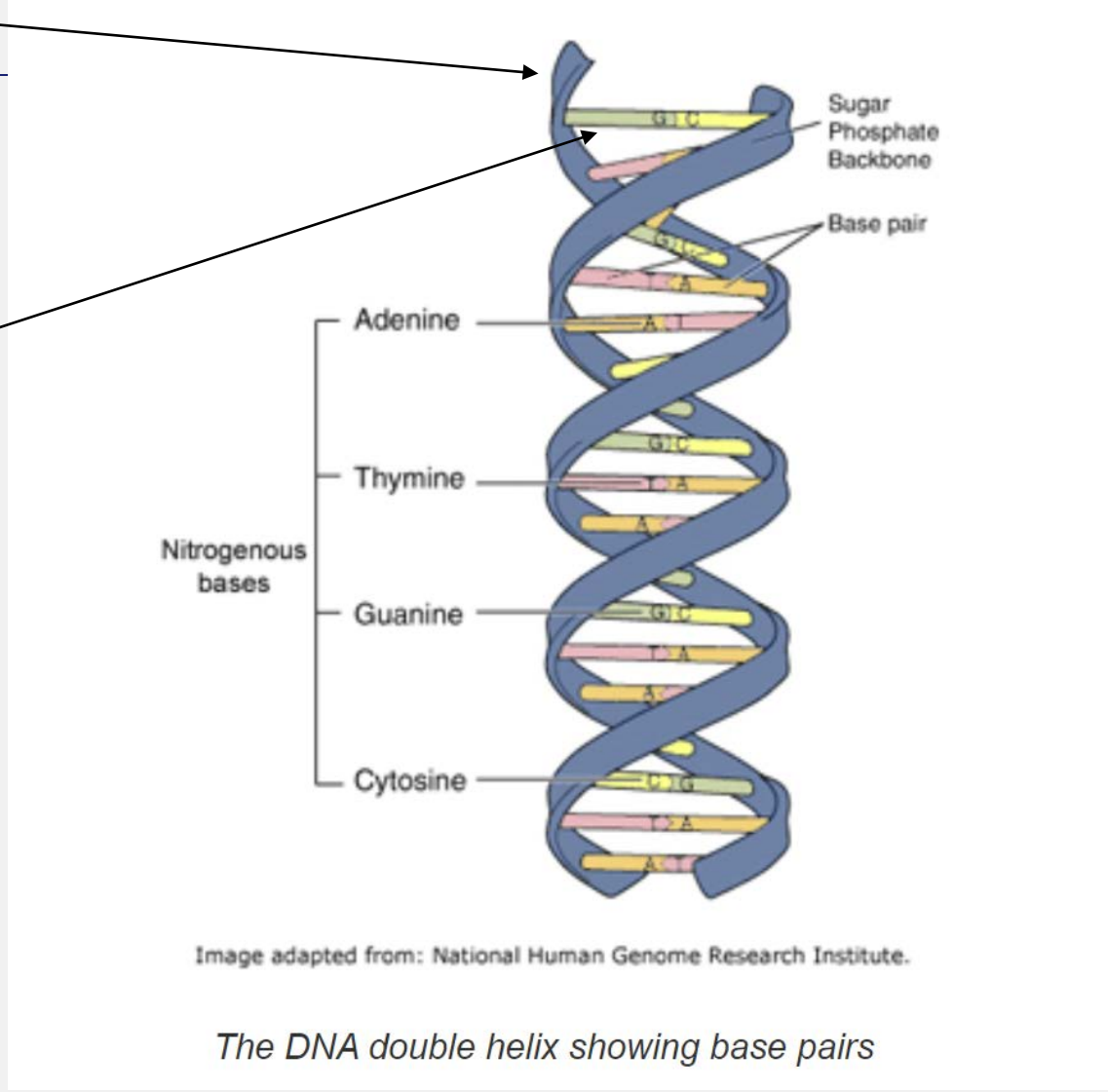
The sides are sugar and phosphate molecules.

The rungs are pairs of chemicals called **'nitrogenous bases**', or **'bases'** for short.

There are four types of base: adenine (A), thymine (T), guanine (G) and cytosine (C).
These bases link in a very specific way: A always pairs with T, and C always pairs with G.

The DNA molecule has two important properties.

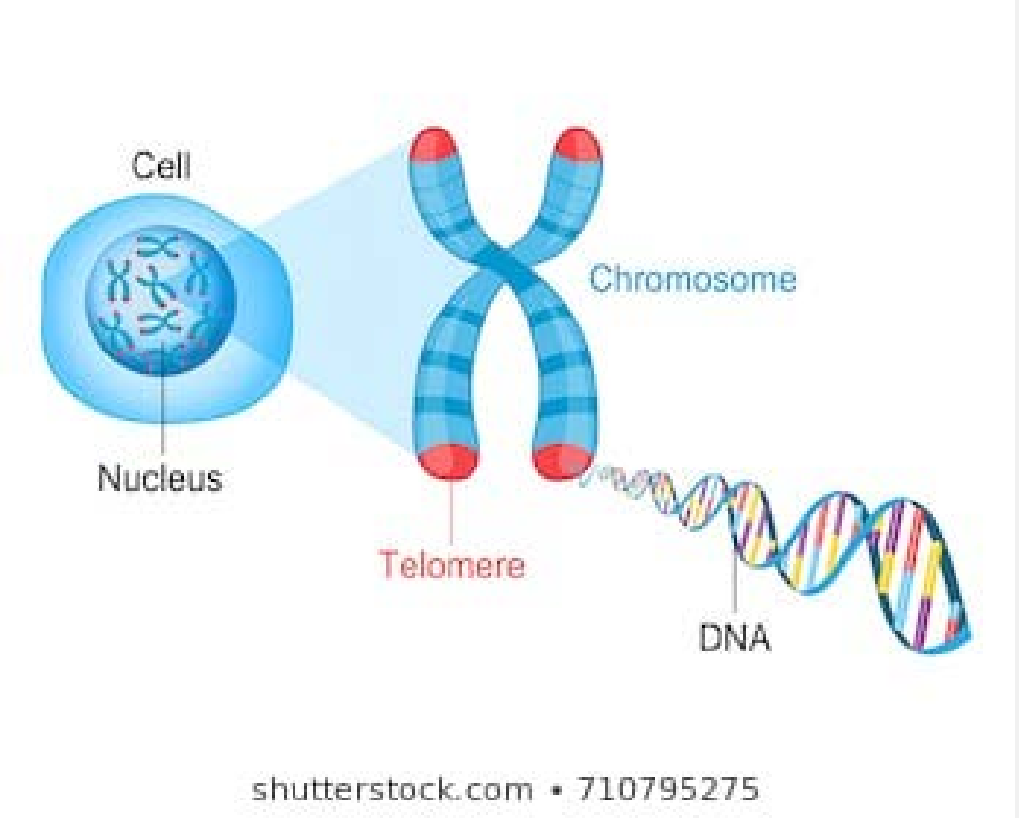
- **It can make copies of itself**. If you pull the two strands apart, each can be used to make the other one (and a new DNA molecule).
- **It can carry information**. The order of the bases along a strand is a code - a code for making proteins.



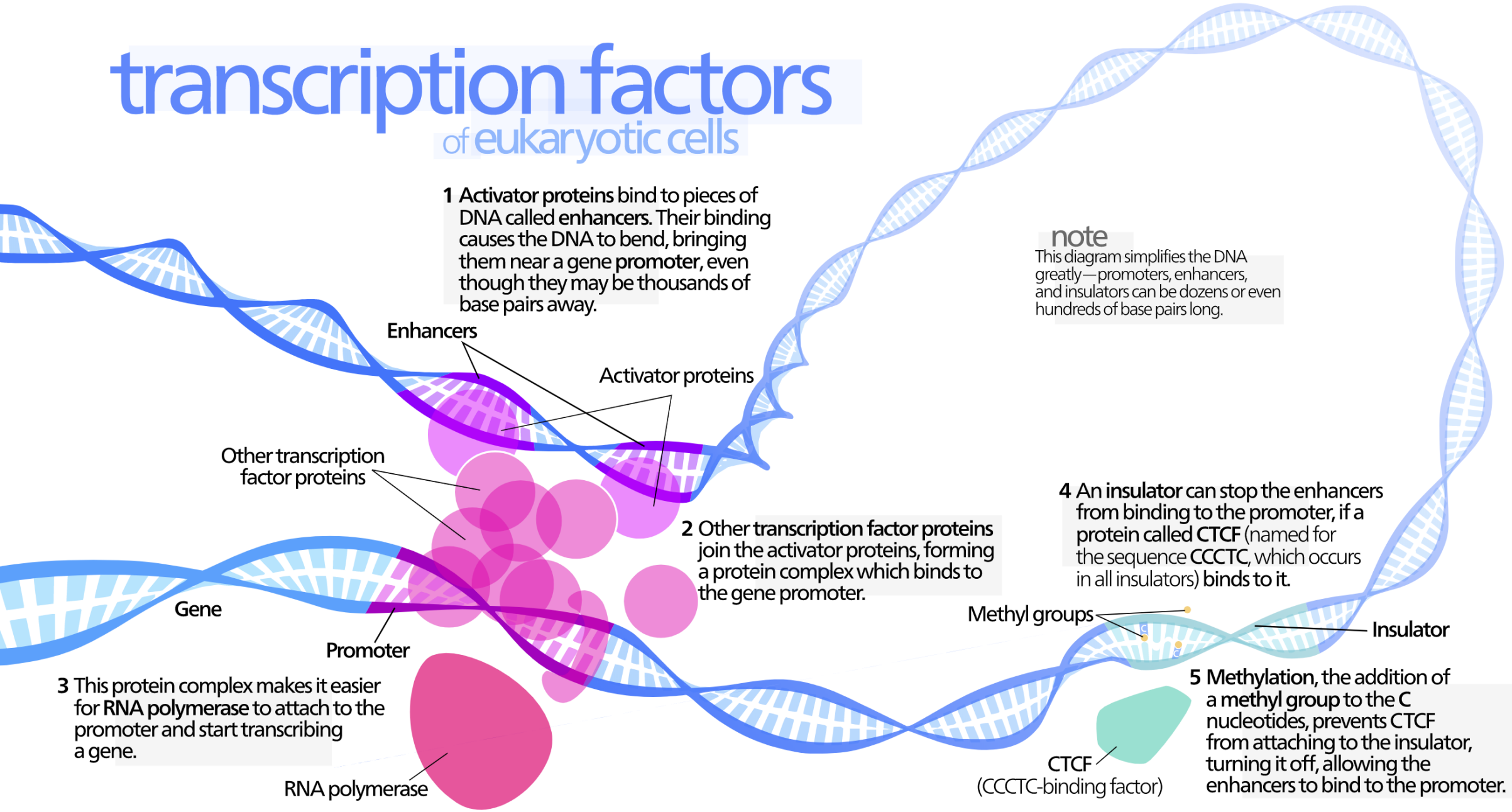
The DNA double helix showing base pairs

A **gene** is a length of DNA that codes for a specific protein. So, for example, one gene will code for the protein insulin, which is important role in helping your body to control the amount of sugar in your blood.

Chromosomes are 46 (types) in each individual, they are long filaments of genes and proteins with an x-shape.



transcription factors
of eukaryotic cells



We describe each genetic sequence (POINT) of active/inactive enhancer/promoter through **epigenomic data**

??? EPIGENOMIC DATA ???

Genotype vs Phenotype

Before, we must see what genotype and phenotypes are

An organism's genotype is the set of genes that it carries.
An organism's phenotype is all of its observable characteristics — which are influenced both by its genotype and by the environment.

So in defining evolution, we are really concerned with changes in the genotypes that make up a population from generation to generation.
However, since an organism's genotype generally affects its phenotype, the phenotypes that make up the population are also likely to change.

For example, differences in the genotypes can produce different phenotypes. In these house cats, the genes for ear form are different, causing one of these cats to have normal ears and the other to have curled ears.

A change in the environment also can affect the phenotype. Although we often think of flamingos as being pink, pinkness is not encoded into their genotype. The food they eat makes their phenotype white or pink

Epigenomics

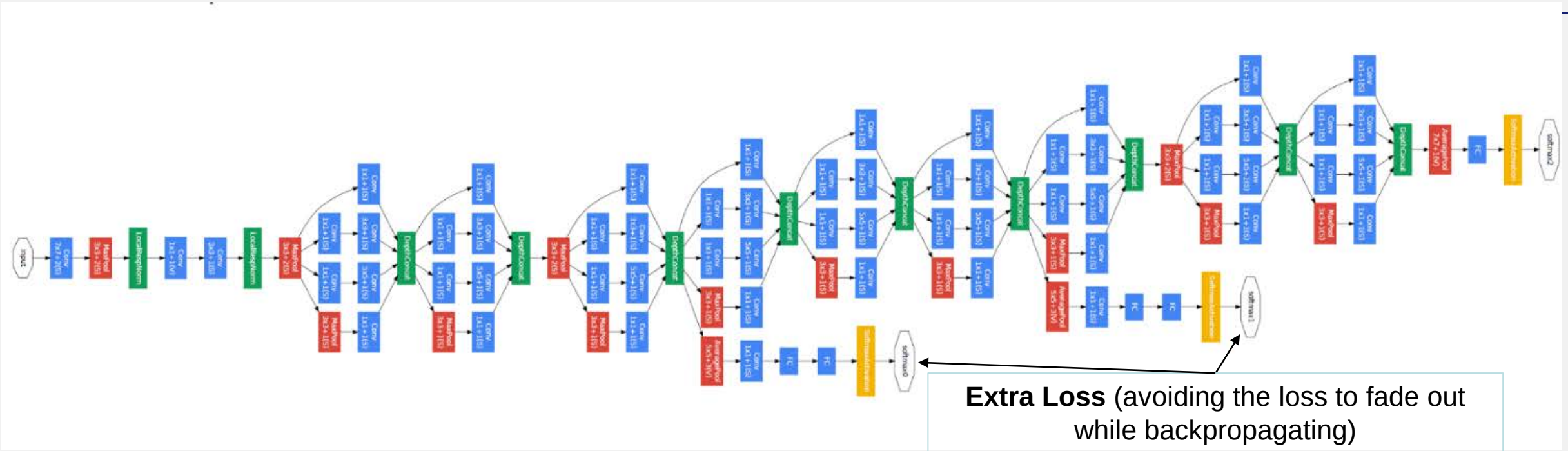
Epigenomics is the study of the complete set of epigenetic modifications on the genetic material of a cell (epigenome).

Epigenetic modifications are reversible modifications on a cell's DNA or histones that affect gene expression without altering the DNA sequence.

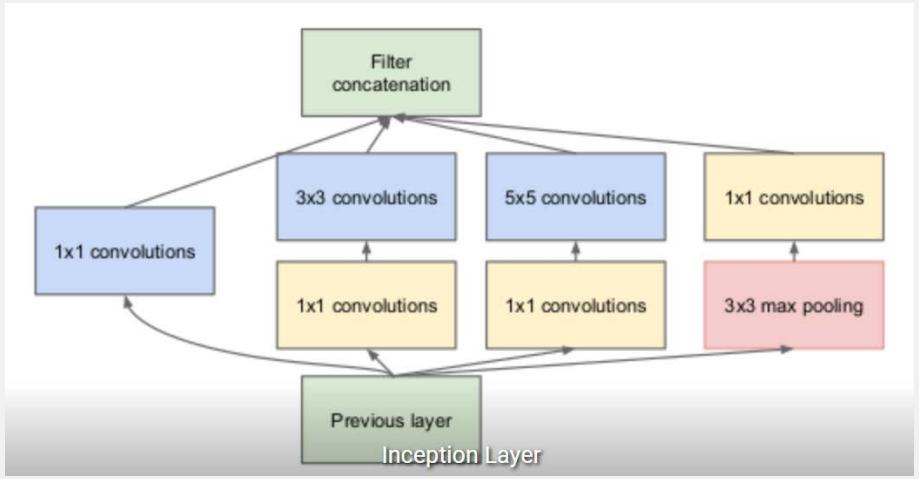
For each sequence, epigenomic data express how much that sequence is involved is a list of genes-protein interactions.

XAI = Explainable Machine Learning

Inception Network (v1)



Extra Loss (avoiding the loss to fade out while backpropagating)



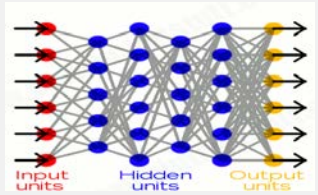
Inception layer: essentially chooses the best filter size

QUITE COMPLICATE…

ISN’T IT?

A. Introduction - The Need for Explainable AI

AI System



Watson

AlphaGo

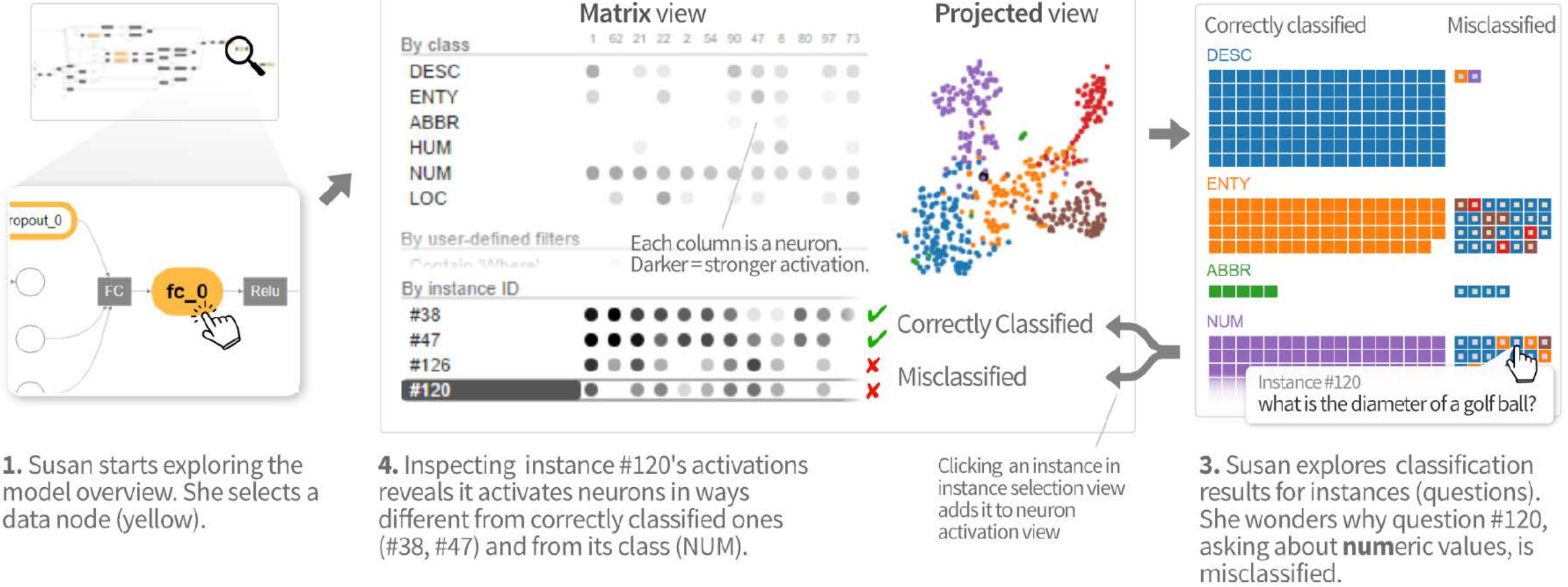
Sensemaking

Operations

User

• Which architecture is used?	Methods for visualizing network architecture: ActiVis
• Is this complex architecture really needed?	Methods for visualizing network architecture: DeepTest
• Is each feature really important?	Explanation vectors
• Why did you do that? • Why not something else? • How do you obtain something else?	Confidence maps, Saliency maps, Relevance Computation
• When do you succeed? • When do you fail?	Counterfactuals Explanations Anchoring explanations (LIME)

ActiVis


A Model Architecture
B Neuron Activation
2. Examines activation patterns for classes and instance subsets
C Instance Selection
Matrix view
Projected view
By class
1 62 21 22 2 54 90 47 8 80 97 73
DESC
ENTY
ABBR
HUM
NUM
LOC
By user-defined filters
By instance ID
#38
#47
#126
#120
Each column is a neuron.
Darker = stronger activation.
Correctly Classified
Misclassified
ropout_0
FC
fc_0
Relu
Correctly classified
Misclassified
DESC
ENTY
ABBR
NUM
Instance #120
what is the diameter of a golf ball?
1. Susan starts exploring the model overview. She selects a data node (yellow).
4. Inspecting instance #120's activations reveals it activates neurons in ways different from correctly classified ones (#38, #47) and from its class (NUM).
Clicking an instance in instance selection view adds it to neuron activation view
3. Susan explores classification results for instances (questions). She wonders why question #120, asking about numeric values, is misclassified.

DeepTest

Activated neuron = neuron whose activation is above a threshold thr

Neuron Coverage for input x: $Cov_{neuron}(x) = \frac{number\ of\ activated\ neurons}{number\ of\ network\ neurons}$

Layer Coverage for input x: $Cov_{layer}(x) = \frac{number\ of\ activated\ neurons\ in\ layer}{number\ of\ neuros\ in\ layer}$

If a neuron/ layer doesn't get coverage during training: is it really needed??

If a neuron/layer has very different coverage on small perturbations of input… is it a stable layer??

If a neuron/layer has same coverage on different classes … is it really discriminating?

AFTER TRAINING: is there a correlation between Neuron/Layer Coverage and classes?

Which are the equivalence classes in the input space is Neuron/Layer Coverage is used as partitioning rule?

Explanation Vectors

$X = \{x_1, \ldots, x_n\}$ are the n input points, where $x_i \in \Re^d$

$Y = \{y_1, \ldots, y_n\}$, $y_i \in \{1,\ldots,C\}$ are the class labels

$$g^*(x) = \arg \min_{c \in \{1,\ldots,C\}} P(Y \neq c \mid X = x)$$

the explanation vector of a data point x_0 is the derivative in x_0 of the conditional probability

$P(Y \neq g^*(x_0) \mid X = x)$, which is:

$$\zeta(x_0) := \frac{\partial}{\partial x} P(Y \neq g^*(x) \mid X = x)\Big|_{x=x_0} .$$

If the classifier is a Gaussian Process Classifier the classification function is explicitly known, therefore we can compute its derivative.

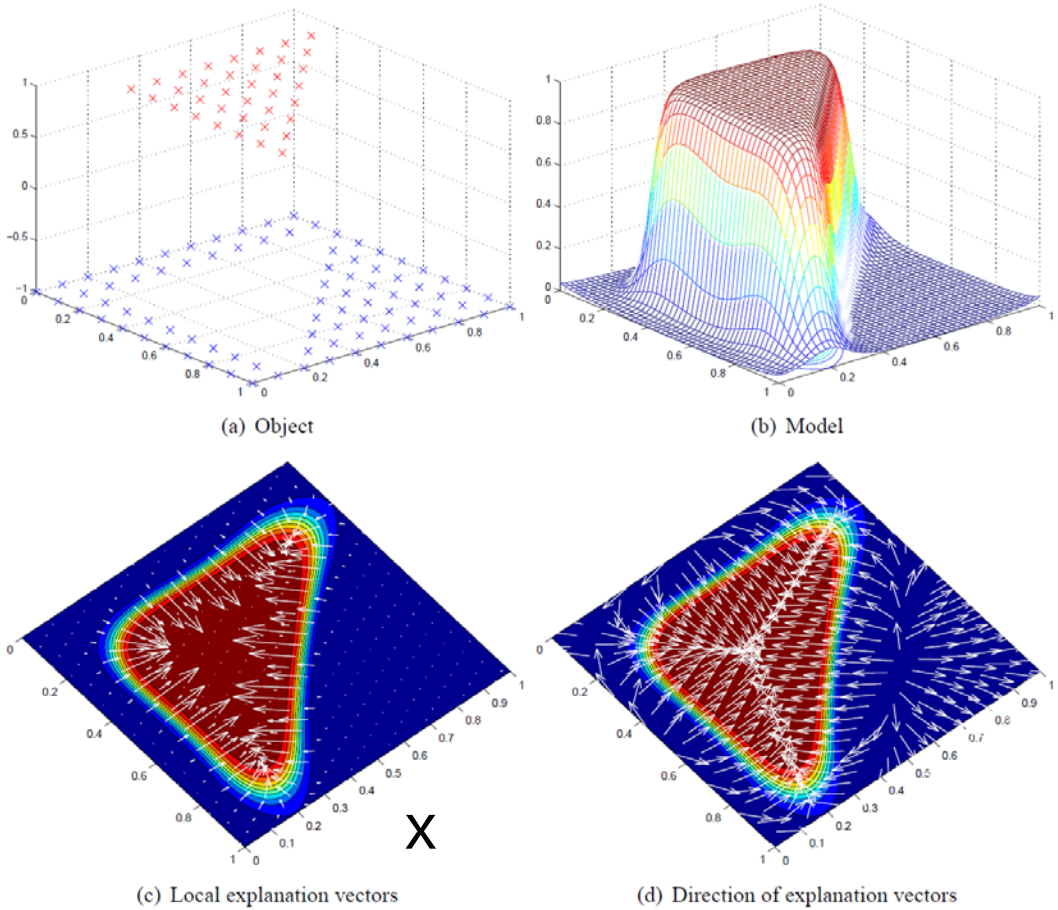


Figure 1: Explaining simple object classification with Gaussian Processes

Panel (a) of Figure 1 shows the training data of a simple object classification task and panel (b) shows the model learned using GPC.[4] The data is labeled -1 for the blue points and $+1$ for the red points. As illustrated in panel (b) the model is a probability function for the positive class which gives every data point a probability of being in this class. Panel (c) shows the probability gradient of the model together with the local gradient explanation vectors. Along the hypotenuse and at the corners of the triangle explanations from both features interact towards the triangle class while along the edges the importance of one of the two feature dimensions dominates. At the transition from the negative to the positive class the length of the local gradient vectors represents the increased importance of the relevant features. In panel (d) we see that explanations close to the edges of the plot (especially in the right hand side corner) point away from the positive class. However, panel (c) shows that their magnitude is very small. For discussion of this issue see Section 8.

If the classifier does not explicitly express the function g*(x) it learns:

$X = \{x_1, \ldots, x_n\}$ are the n input points, where $x_i \in \Re^d$

$Y = \{y_1, \ldots, y_n\}$, $y_i \in \{1,\ldots,C\}$ are the class labels

training →

g*(x) unknown

Create a surrogate classifier

$X = \{x_1, \ldots, x_n\}$ are the n input points, where $x_i \in \Re^d$

$Y = \{g^*(x_1), \ldots, g^*(x_n)\}$, $g^*(x_i)$ are the labels!

training →

$P_{surrogate}(g^*(x) \neq c \mid X = x)$ known!!

Explanation vector on surrogate

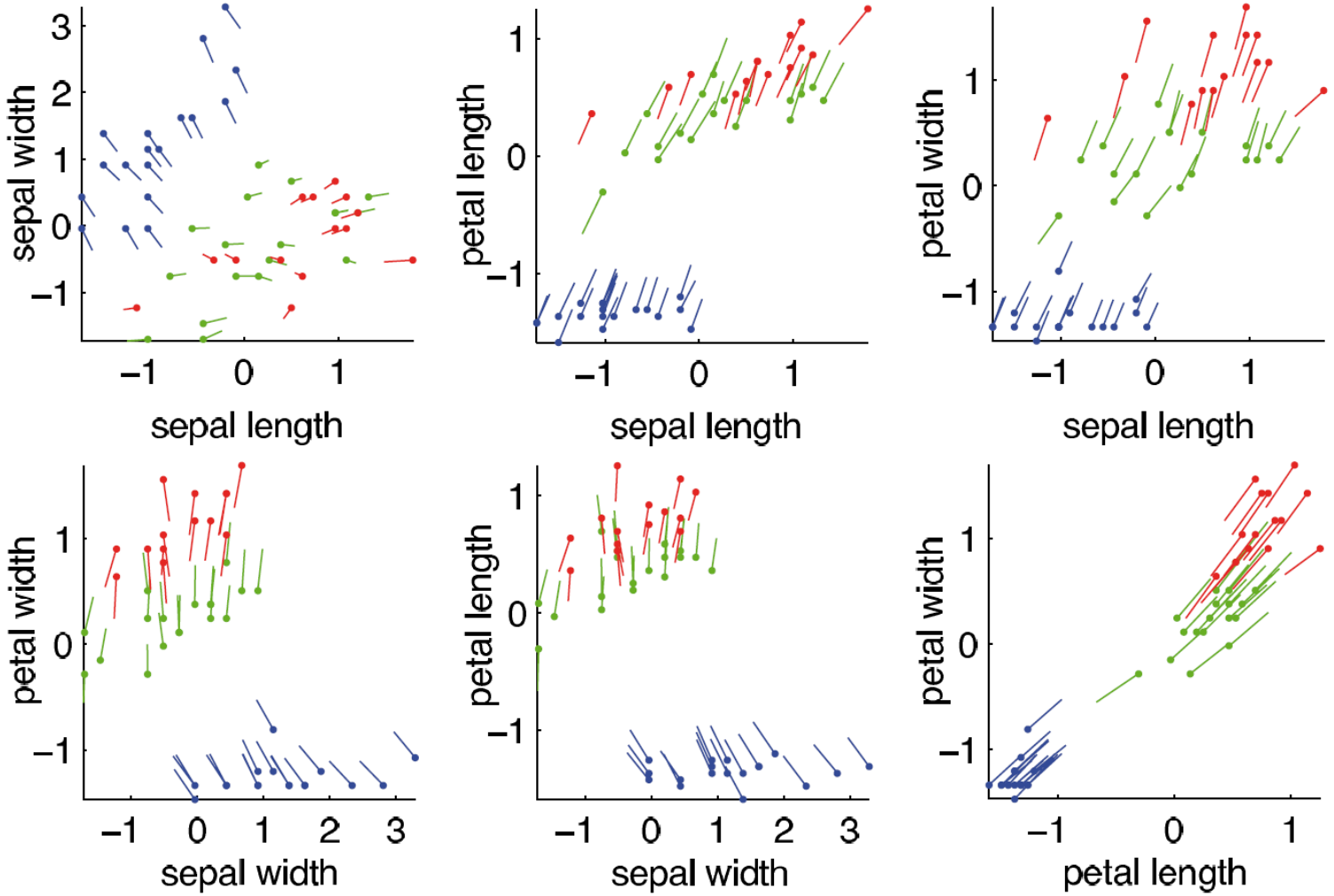


Figure 3: Scatter plots of the explanation vectors for the test data. Shown are all explanation vectors for both classes: class 1 containing *Iris setosa* (shown in blue) and *Iris virginica* (shown in red) versus class 0 containing only the species *Iris versicolor* (shown in green). Note that the explanations why an Iris flower is not an *Iris versicolor* is different for *Iris setosa* and *Iris virginica*.

Sensitivity analysis

A first approach to identify the most important input features is sensitivity analysis [77,66,29]. It is based on the model's locally evaluated gradient or some other local measure of variation. A common formulation of sensitivity analysis defines relevance scores as

$$R_i(\boldsymbol{x}) = \left(\frac{\partial f}{\partial x_i}\right)^2, \tag{1}$$

The Taylor decomposition [9,5] is a method that explains the model's decision by decomposing the function value $f(\boldsymbol{x})$ as a sum of relevance scores. The relevance scores are obtained by identification of the terms of a first-order Taylor expansion of the function at some root point $\widetilde{\boldsymbol{x}}$ for which $f(\widetilde{\boldsymbol{x}}) = 0$. The root point should remove the information in the input that causes $f(\boldsymbol{x})$ to be positive, e.g. the pattern in a given input image that is responsible for class membership as modeled by the function. Taylor expansion lets us rewrite the function as:

$$f(\boldsymbol{x}) = \sum_{i=1}^{d} R_i(\boldsymbol{x}) + O(\boldsymbol{x}\boldsymbol{x}^\top)$$

where the relevance scores

$$R_i(\boldsymbol{x}) = \left.\frac{\partial f}{\partial x_i}\right|_{\boldsymbol{x}=\widetilde{\boldsymbol{x}}} \cdot (x_i - \widetilde{x}_i)$$

Saliency Maps

A trained network is defined by fixed parameters ϑ

The activation $h_{(i,j)}$ of a unit i at layer j depends both on the parameters ϑ and on the input x

$h_{(i,j)} = h_{(i,j)} (\vartheta , x)$

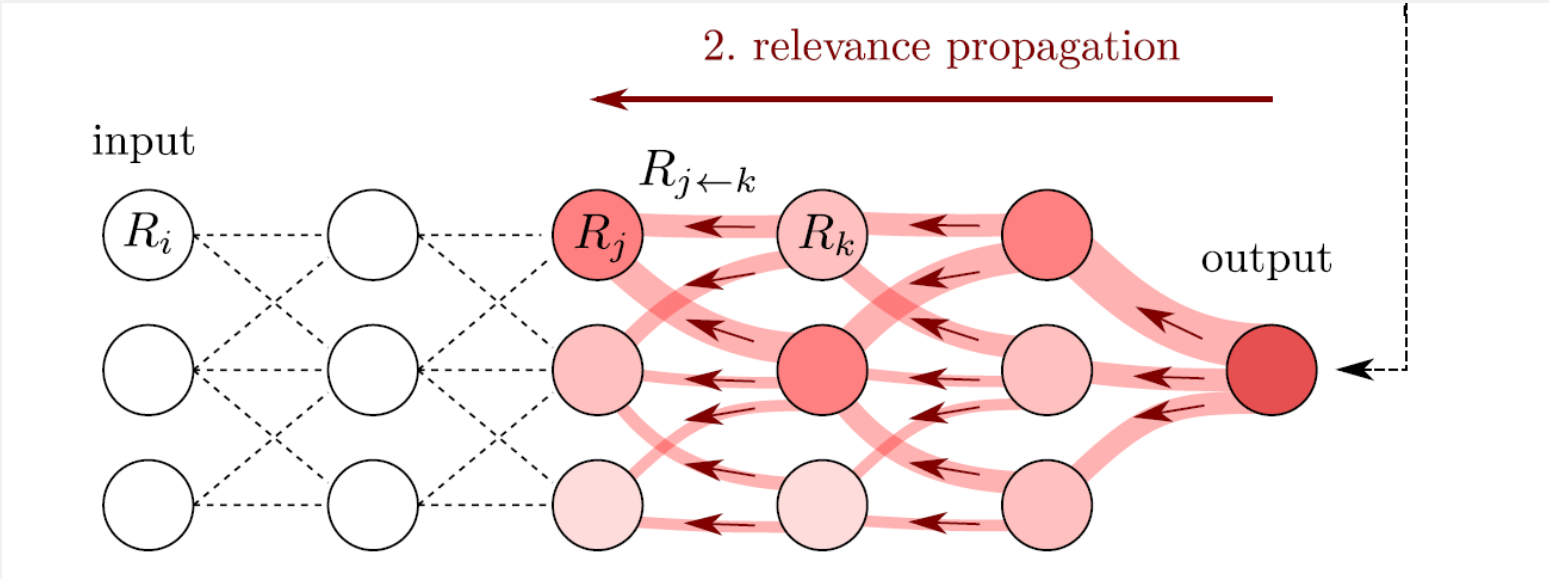
Since ϑ is fixed, to find the x that maximize $h_{(i,j)}$, find x that maximizes $h_{(i,j)}$

by computing the derivative and finding a (local) maxima.

In the nets.. Derivative is computed by backpropagation!!

And Therefore…

Max Activation



Instance-level explanations (Layer-wise relevance propagation alias LRP)

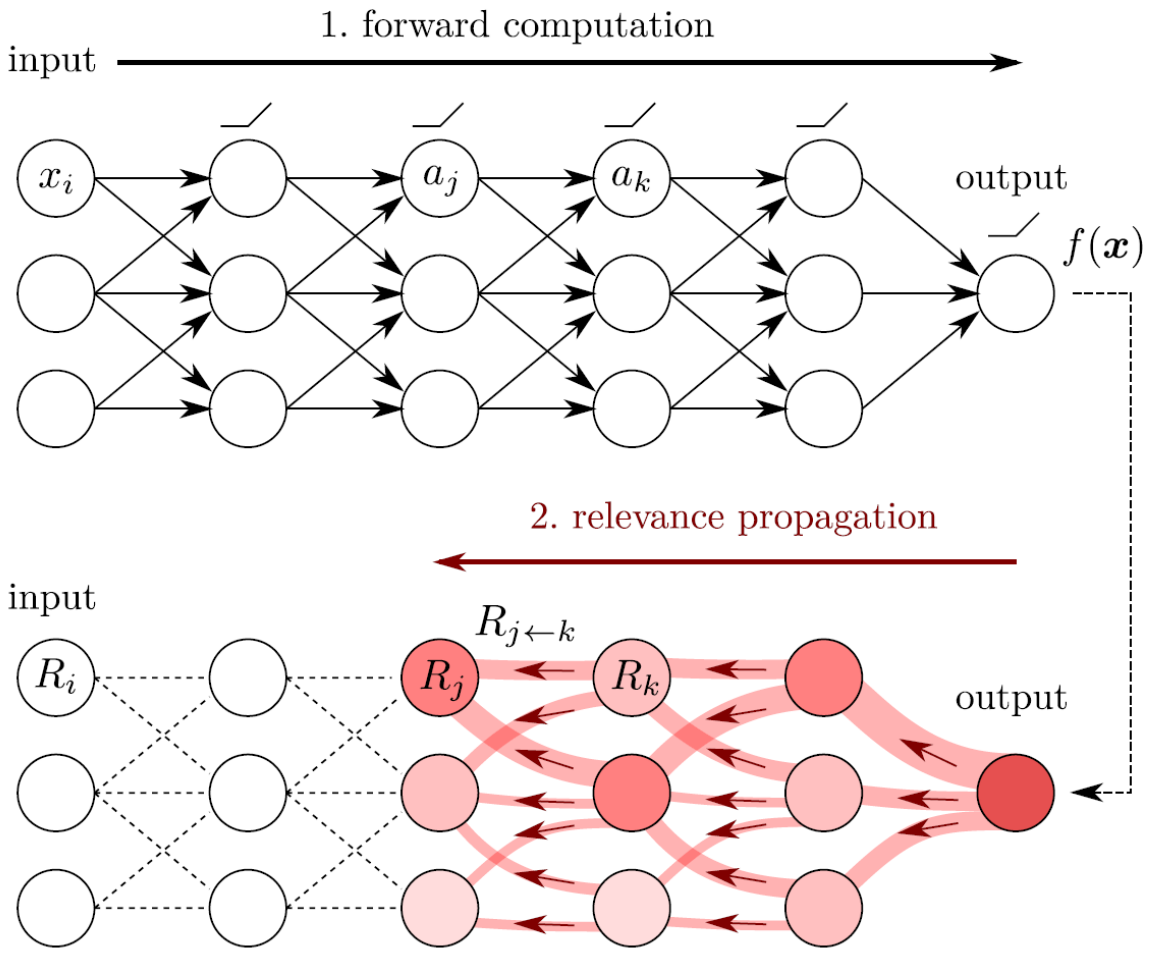


Fig. 11. Diagram of the LRP procedure (here after three steps of redistribution). Red arrows indicate the relevance propagation flow. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

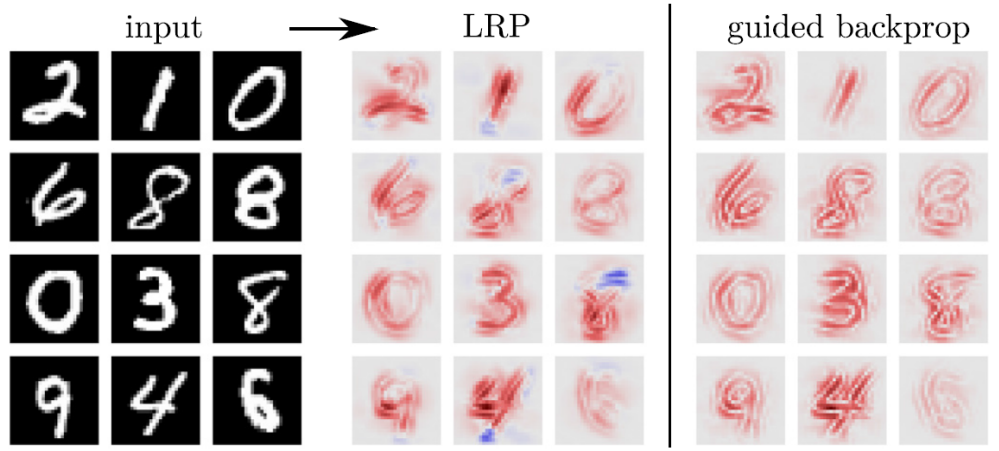


Fig. 10. LRP applied to a convolutional DNN trained on MNIST, and resulting explanations for selected digits. Red and blue colors indicate positive and negative relevance scores respectively. Heatmaps are shown next to those produced by guided backprop. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Wha makes a good visual explanation of a model?

a ‘good’ visual explanation of a model for justifying any target category should be:

(a) Class – discriminative (i.e. localize the category in the image)

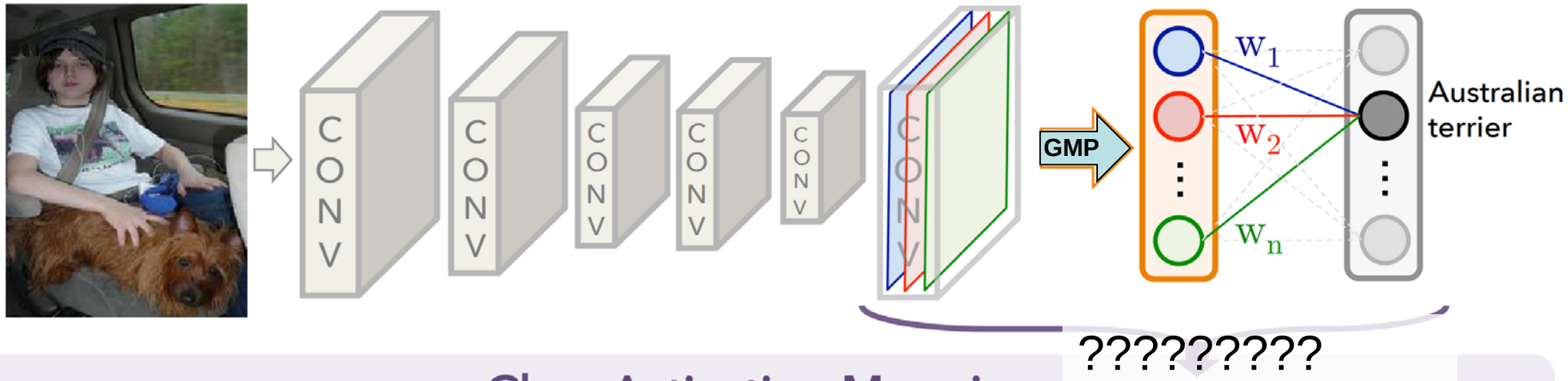
(b) high-resolution (i.e. capture fine-grained detail)

LRP has high-resolution but is not class-discriminative

CAM and grad-CAM are also class discriminative!

CAM: Class Activation Mapping on ConvNets (with Global Max Pooling??)

Net composed by several conv feature maps -> GMP -> fully connected layer (softmax activation)



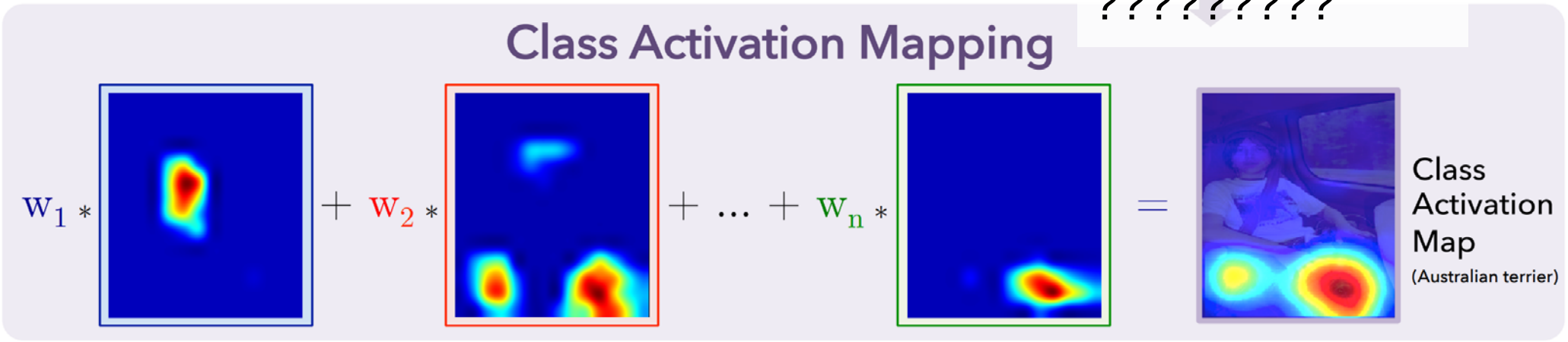
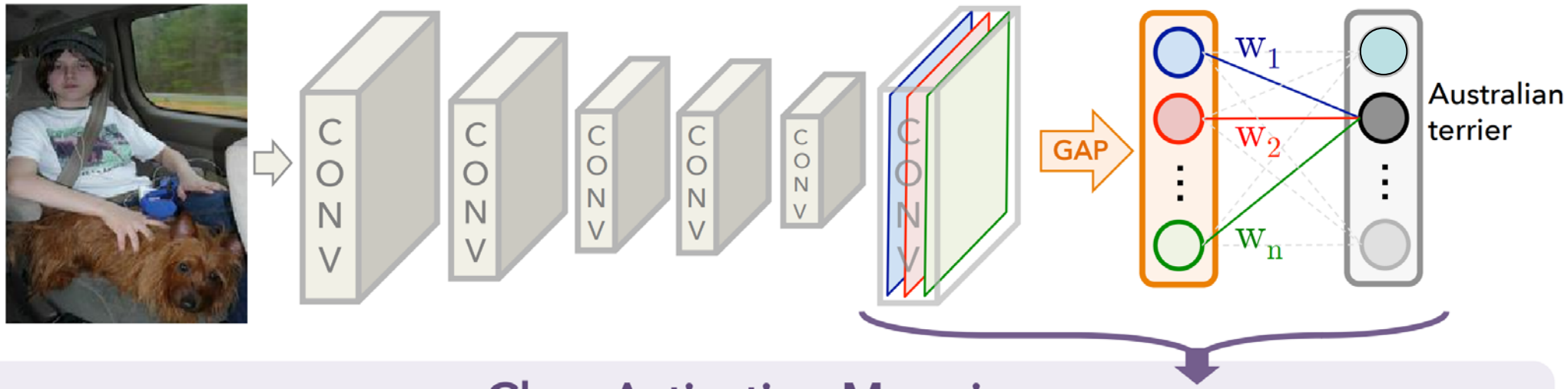


Figure 2. Class Activation Mapping: the predicted class score is mapped back to the previous convolutional layer to generate the class activation maps (CAMs). The CAM highlights the class-specific discriminative regions.

YES with Global Average Pooling (GAP)

Net composed by several conv feature maps -> **GAP** -> fully connected layer (softmax activation)



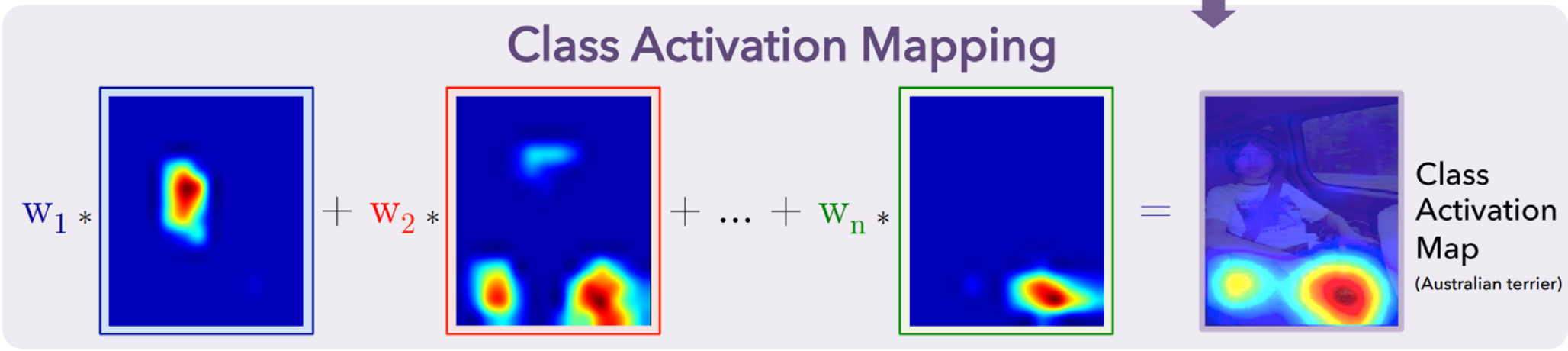
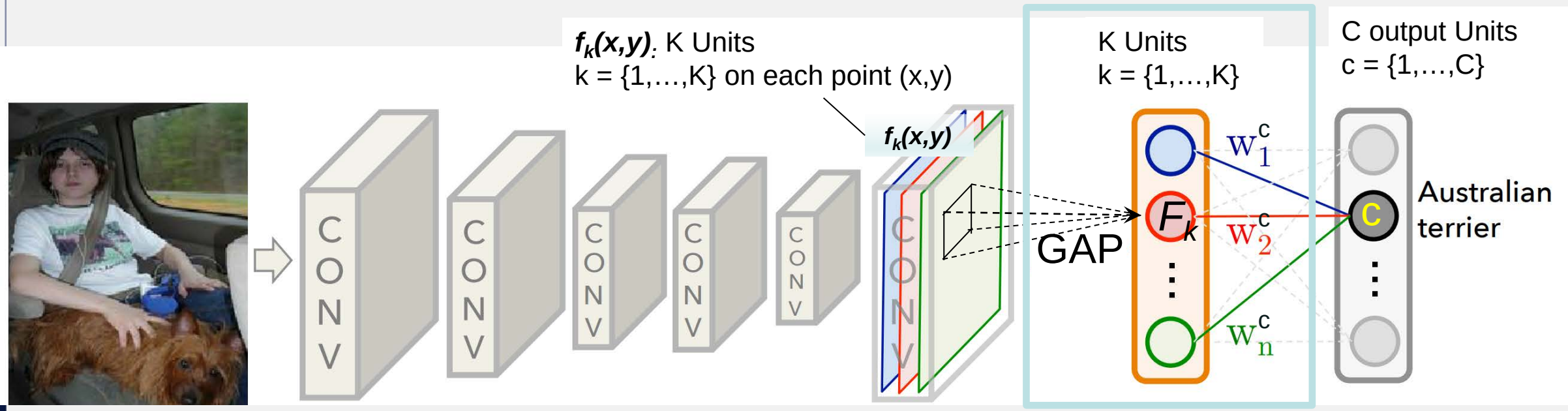


Figure 2. Class Activation Mapping: the predicted class score is mapped back to the previous convolutional layer to generate the class activation maps (CAMs). The CAM highlights the class-specific discriminative regions.



$f_k(x,y)$ has a value depending on some visual pattern within its receptive field

After training w_k^c is fixed

Substituting max pooling with average pooling you can:

For a given image, let $f_k(x, y)$ represent the activation of unit k in the last convolutional layer at spatial location (x, y). Then, for unit k, the result of performing global average pooling, F^k is $\sum_{x,y} f_k(x, y)$. Thus, for a given class c, the input to the softmax, S_c, is $\sum_k w_k^c F_k$ where w_k^c is the weight corresponding to class c for unit k. Essentially, w_k^c indicates the *importance* of F_k for class c. Finally the output of the softmax for class c, P_c is given by $\frac{\exp(S_c)}{\sum_c \exp(S_c)}$. Here we ignore the bias term: we explicitly set the input bias of the softmax to 0 as it has little to no impact on the classification performance.

By plugging $F_k = \sum_{x,y} f_k(x, y)$ into the class score, S_c, we obtain

$$S_c = \sum_k w_k^c \sum_{x,y} f_k(x, y)$$
$$= \sum_{x,y} \sum_k w_k^c f_k(x, y). \quad (1)$$

We define M_c as the class activation map for class c, where each spatial element is given by

$$M_c(x, y) = \sum_k w_k^c f_k(x, y). \quad (2)$$

Thus, $S_c = \sum_{x,y} M_c(x, y)$, and hence $M_c(x, y)$ directly indicates the importance of the activation at spatial grid (x, y) leading to the classification of an image to class c.

A drawback of CAM is that it requires feature maps to directly precede softmax layers, so it is only applicable to a particular kind of CNN architectures performing global average pooling over convolutional maps immediately prior to prediction (conv feature maps -> global average pooling -> softmax layer)

gradCAM generalized to different networks

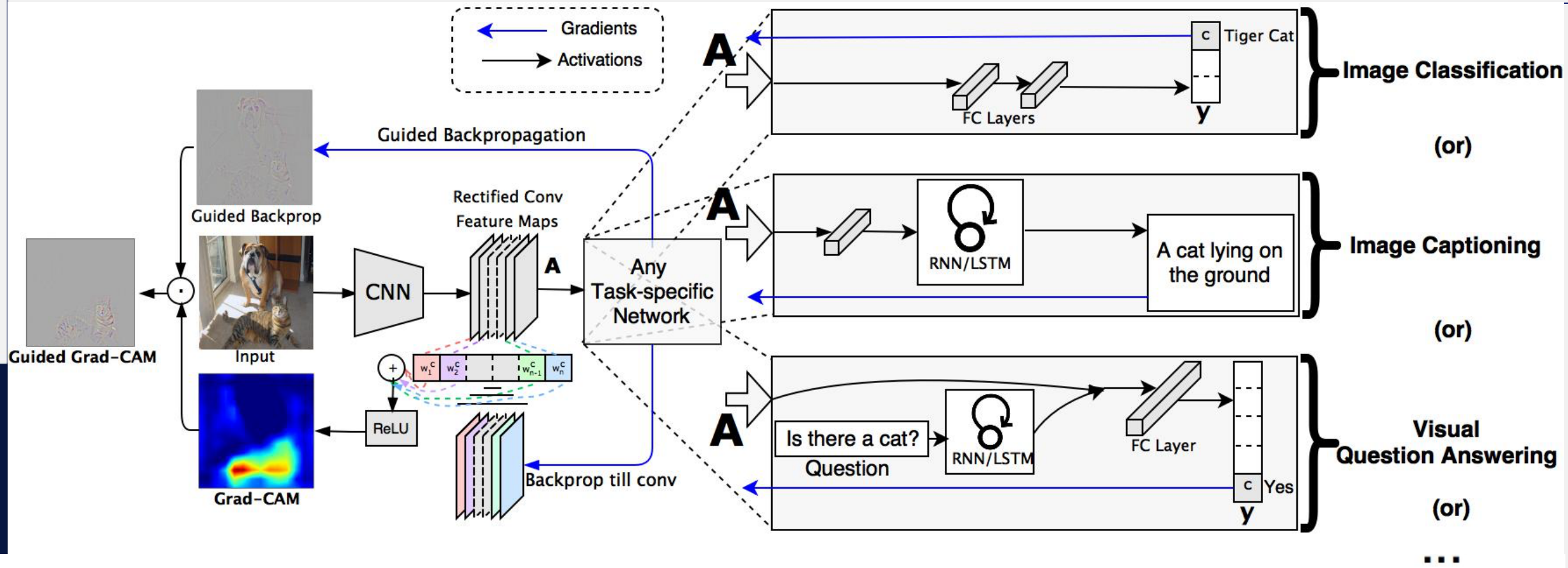
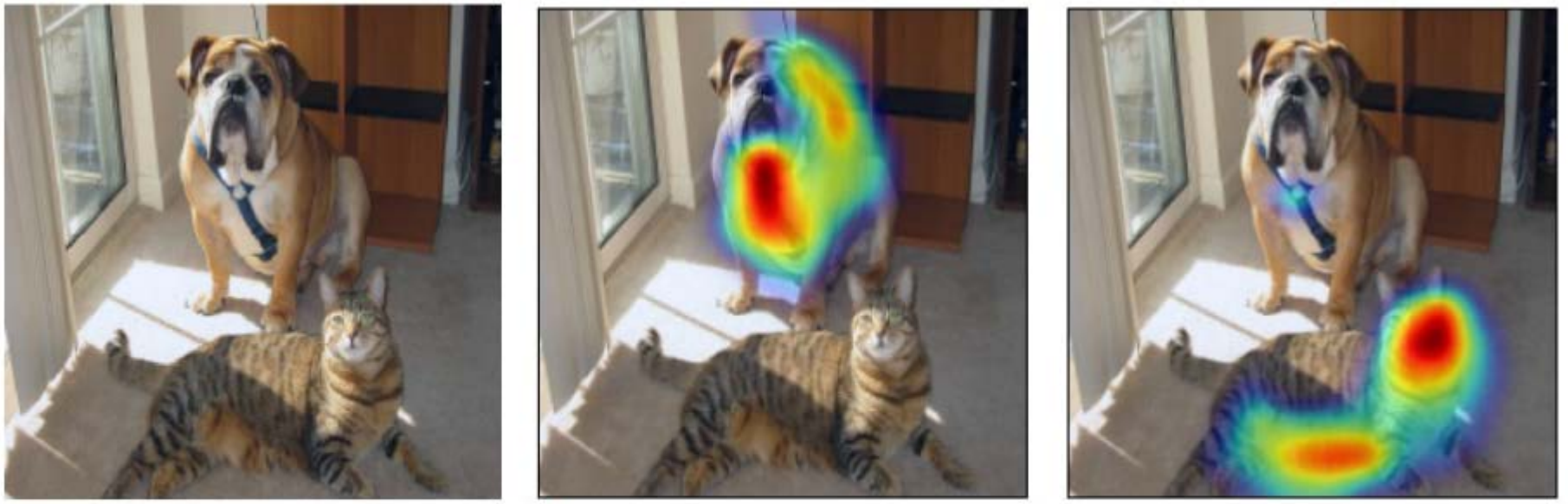


Fig. 2: Grad-CAM overview: Given an image and a class of interest (*e.g.*, 'tiger cat' or any other type of differentiable output) as input, we forward propagate the image through the CNN part of the model and then through task-specific computations to obtain a raw score for the category. The gradients are set to zero for all classes except the desired class (tiger cat), which is set to 1. This signal is then backpropagated to the rectified convolutional feature maps of interest, which we combine to compute the coarse Grad-CAM localization (blue heatmap) which represents where the model has to look to make the particular decision. Finally, we pointwise multiply the heatmap with guided backpropagation to get Guided Grad-CAM visualizations which are both high-resolution and concept-specific.

By choosing the score that is propagated back you compute counterfactuals (???)



(a) Original Image (b) Cat Counterfactual exp (c) Dog Counterfactual exp

Fig. 3: Counterfactual Explanations with Grad-CAM

Counterfactuals?

A counterfactual explanation describes a causal situation in the form:

“If X had not occurred, Y would not have occurred”.

For example: “If I hadn’t taken a sip of this hot coffee, I wouldn’t have burned my tongue”.

Event Y is that I burned my tongue; cause X is that I had a hot coffee.

Thinking in counterfactuals requires imagining a hypothetical reality that contradicts the observed facts (e.g. a world in which I have not drunk the hot coffee), hence the name “counterfactual”.

The ability to think in counterfactuals makes us humans so smart compared to other animals.

https://christophm.github.io/interpretable-ml-book/

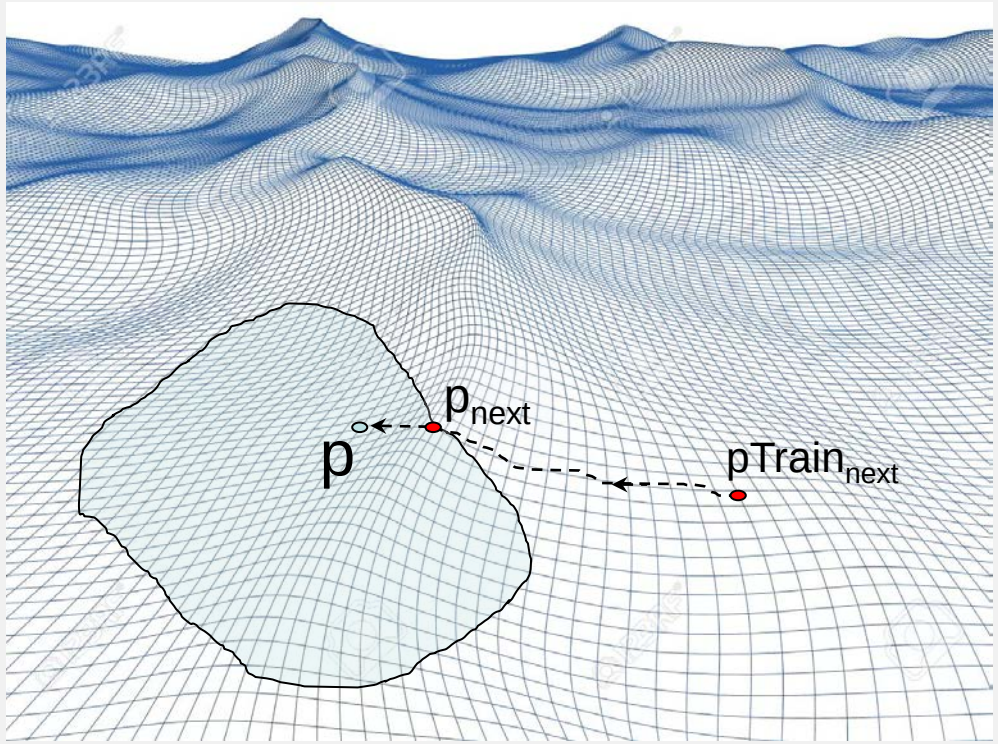
How to compute couterfactuals for a point p:

- Find the nearest point to p (p_{next}) that gets a different prediction from your black box.
- Find the features (f_{diff}) of p_{next} differing from those of p
- Event Y is the classification score for p. Cause X is that features f_{diff} of X have values different than the f_{diff} of p.

Counterfactuals allow computing suggestions!!

To compute p_{next}:

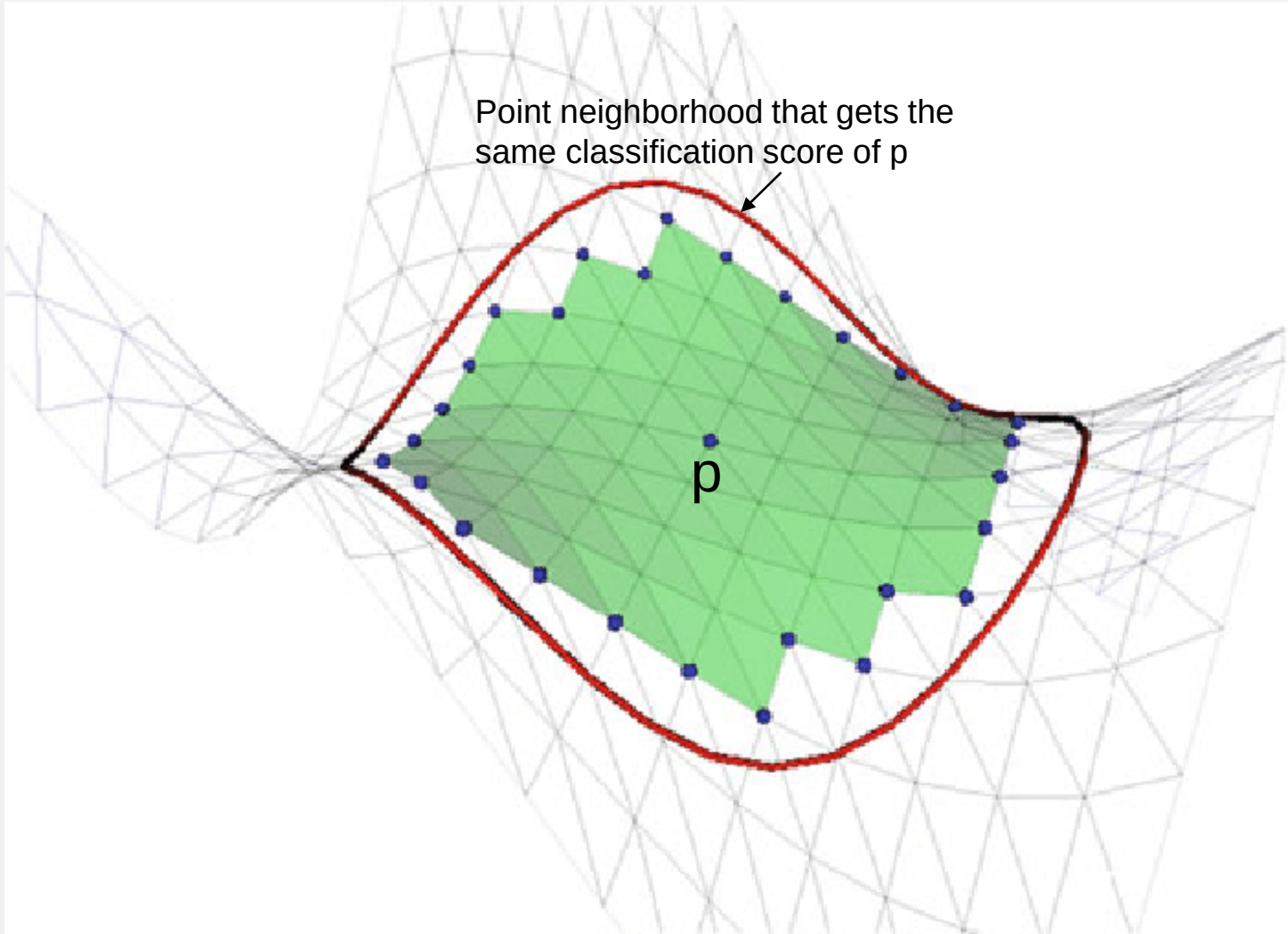
- find the nearest training point $pTrain_{next}$ which belongs to the other class
- Move from $pTrain_{next}$ to p until there a change of classification



Anchors

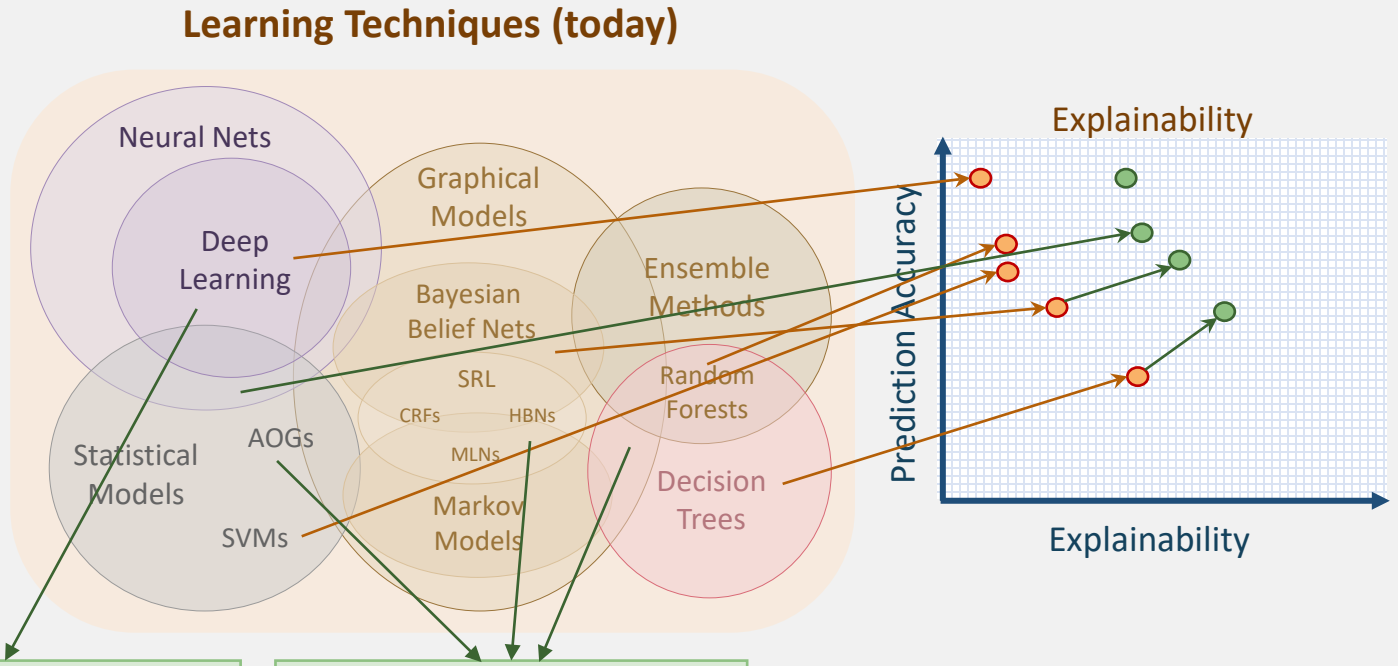
Anchor values are those values that anchor the prediction.

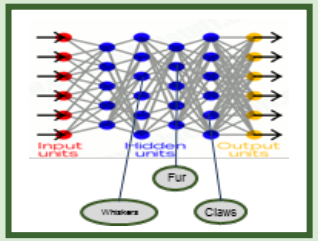
Based on the assumption that a model is linear in its neighborhood.



Learning an interpretable surrogate (model induction)

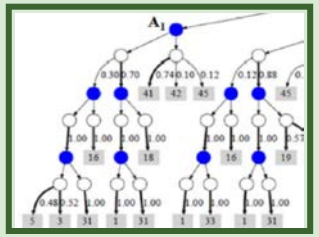
Create a suite of machine learning techniques that produce more explainable models, while maintaining a high level of learning performance





Deep Explanation

Modified deep learning techniques to learn explainable features



Interpretable Models

Techniques to learn more structured, interpretable, causal models

AOG = And-Or graph
HBN = hierarchical Bayesian Networks
CRF = Conditional random fields
MLN = Markov Linear Networks

LIME for instance-level explanations

For each point to be explained, given a trained black-box, LIME trains local surrogate models:

- Perturb the dataset and get the black box predictions for these new points.
- Weight the new samples according to their proximity to the instance of interest.
- Train a weighted, interpretable model on the dataset with the variations.
- Explain the prediction by interpreting the local model.

The surrogate must have a good local accuracy