
Maybe ω_2
Perhaps ω_1
A clear example of ω_1
Why didn't we take the ML1 course?
It's ω_1!
No doubt : ω_2
Maybe ω_1, maybe ω_2
Definitely not ω_1

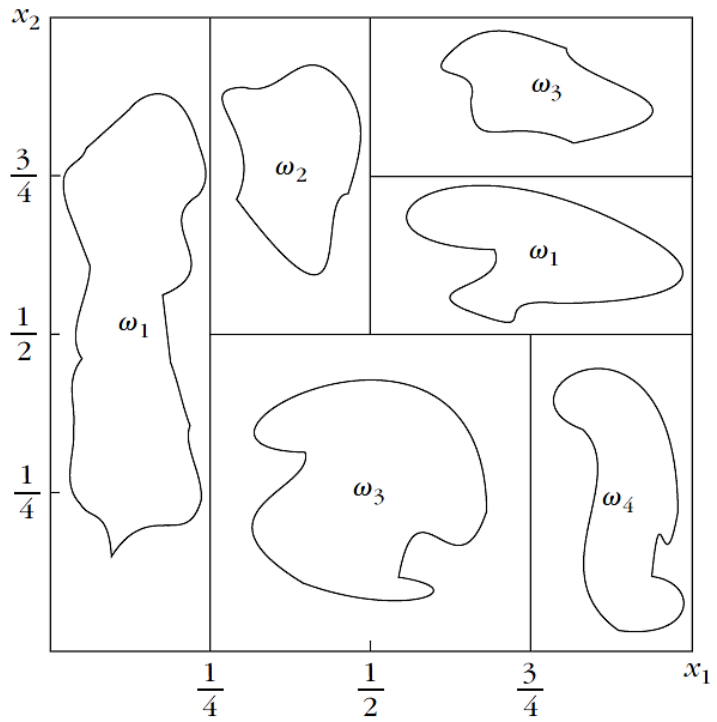
Decision Trees, Combining, and a [Whiff of] ANNs

› Marco Loog

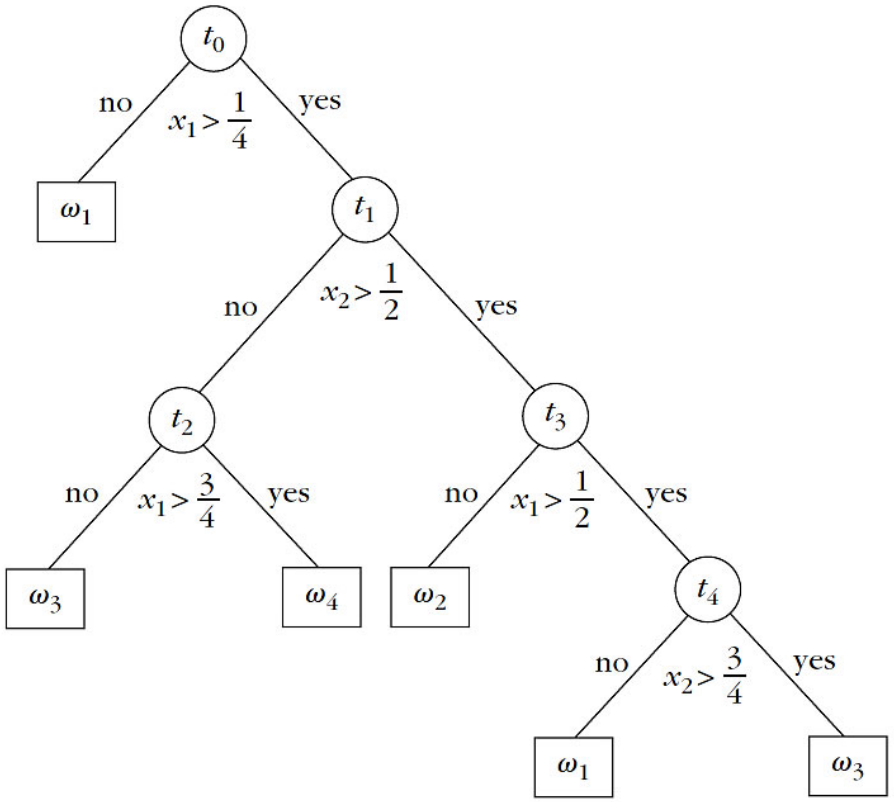
Three loosely connected topics…

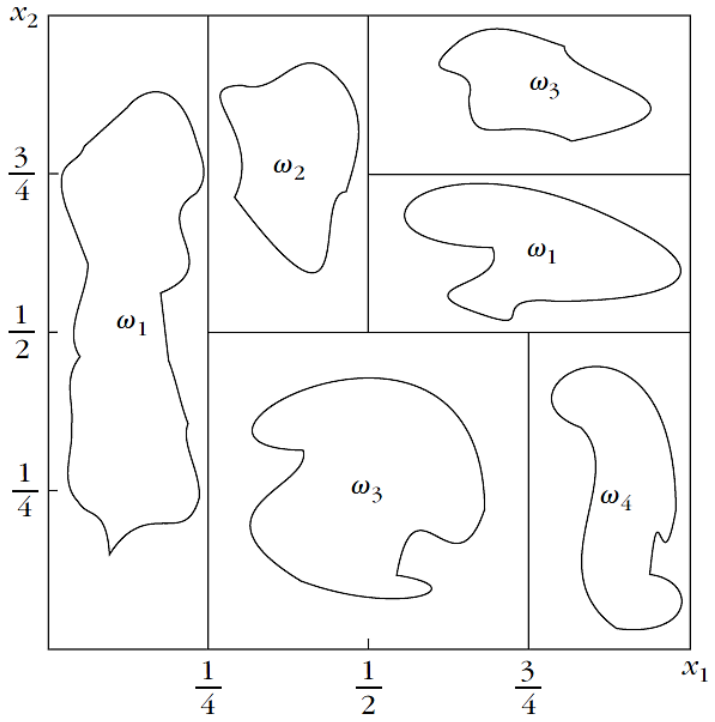
› Decision trees : yet another classifier
› Combining trees can fix high-variance problem
› On to general combining classifiers : how and why?
› Fixed vs. trained combiners [notably boosting]
› Neural networks : the ultimate trained combiner?
› Some further [and brief] observations on ANNs

A Decision Tree Example



A Decision Tree Example





Decision Tree Ingredients

› Set of questions to be asked

$x_k < \alpha?$

› Splitting criterion

E.g. entropy-based node impurity / Other options?

› Stop-splitting rule [or just : stopping rule]

› Class assignment rule in leafs

Decision Trees : Remarks

- Size is a critical factor : Not too small, not too large
- Variance of trees is a problem
  - Small changes in data may lead to large changes in tree
- Bagging is one way to reduce this variance…

Bagging

› Portmanteau of bootstrap aggregating

Bootstrap data sample to train a classifier

Average classifier outcomes [labels +1/−1 in two-class case]

› Bagging is general combining methodology

Currently still popular random forests combine bagging and random features

Combining Classifiers

- The idea of combining classifier outcomes has been widely studied and is still broadly applied
  - Synonyms : combining classifiers, multiple-classifier systems, ensemble methods, more?
- Combining creates wide range of possibilities to construct new classifiers from existing ones
  - By combining combiners, possibilities are endless…

Combining Classifiers

› Reasons for combining :

Even if one classifier performs better than other, former may still be better in particular part of feature space

[We already saw:] stabilizing variable classifiers, e.g., by means of bagging

› Two major classes : fixed and learned combiners

Fixed Combining Rules

Averaging rule
Max rule
Min rule
Product rule
Median rule
Voting rule
Etc.

› Things to be combined are class labels, posteriors, rankings, or maybe other classifier outputs

Fixed Combining Rules

› Rules often heuristic / justified conceptually

› One clear exception? : Product rule can be derived theoretically…

  Given the right assumptions…

› Generally, theoretical analysis is hard

  E.g. when and why does a combiner give improvements?

Condorcet Paradox

› Still, some interesting observations / theory can be found in areas like social choice theory, e.g. :

› Classifier 1 prefers class A to B to C
Classifier 2 prefers class B to C to A
Classifier 3 prefers class C to A to B

Of course, every choice is as good as every other

But it is worse : choose one preferred class, say A, this leaves us with C, that is preferred by the two other classifiers…

Major Issue to Realize…

- How to generate different classification outcomes to combine?
  - Different classifiers
  - Different object samplings
  - Different feature samplings
  - More?

- These procedures generate so-called base classifiers

Trained Combiners

› Instead of simple combining rule
  - One can take outputs of base classifiers as features
  - Classifier trained on to of these is the combiner

› An elaborate type of trained combiner is stacking
  - Tries to avoid “testing on training set”

› Even more advanced [or complicated?] is boosting

Boosting

› Greedily constructs two-class classifier of the form

$$f(x) = \text{sign}[\sum \alpha_k \, \phi(x; \theta_k)]$$

with base classifiers ϕ under exponential loss

$$\sum \exp[-y_i \sum \alpha_k \, \phi(x_i; \theta_k)]$$

Boosting

› Greedily constructs two-class classifier of the form

$$f(x) = \text{sign}[\sum \alpha_k \, \phi(x; \theta_k)]$$

› Direct optimization of α and θ is difficult, however

› Boosting suggests a stage-wise approach

First, say, k terms of $\sum \alpha_k \, \phi(x; \theta_k)$ are kept fixed
and term $k + 1$ is being optimized

AdaBoost

- Initialize: $w_i^{(1)} = \frac{1}{N},\ i = 1, 2 \ldots, N$
- Initialize: $m = 1$
- Repeat
  - Compute optimum $\boldsymbol{\theta}_m$ in $\phi(\cdot; \boldsymbol{\theta}_m)$ by minimizing P_m; (4.135)
  - Compute the optimum P_m; (4.135)
  - $\alpha_m = \frac{1}{2} \ln \frac{1-P_m}{P_m}$
  - $Z_m = 0.0$
  - For $i = 1$ to N
    - $w_i^{(m+1)} = w_i^{(m)} \exp\left(-y_i \alpha_m \phi(\boldsymbol{x}_i; \boldsymbol{\theta}_m)\right)$
    - $Z_m = Z_m + w_i^{(m+1)}$
  - End{For}
  - For $i = 1$ to N
    - $w_i^{(m+1)} = w_i^{(m+1)} / Z_m$
  - End {For}
  - $K = m$
  - $m = m + 1$
- Until a termination criterion is met.
- $f(\cdot) = \text{sign}(\sum_{k=1}^{K} \alpha_k \phi(\cdot, \boldsymbol{\theta}_k))$

Remark about (4.135)

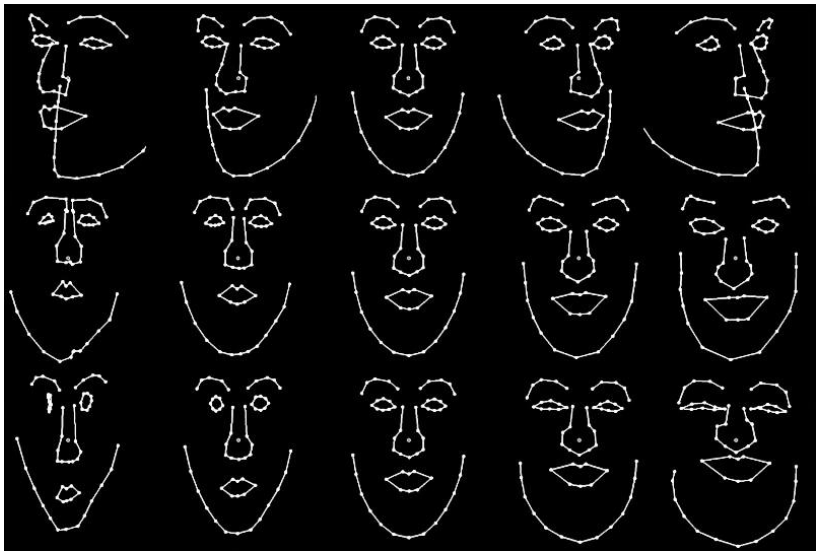
› It asks to minimize the 0-1 risk for $\phi(x; \theta_k)$…

But what is the problem with that generally?

Any classifier for which this may not be a problem?

› Ultimately, AdaBoost idea is applied more liberally…

Example Use of Combining?



On Combining Computer-Aided Detection Systems

Meindert Niemeijer*, Marco Loog, Michael David Abràmoff, *Senior Member, IEEE*, Max A. Viergever, *Fellow, IEEE*, Mathias Prokop, and Bram van Ginneken, *Member, IEEE*

***Abstract*—Computer-aided detection (CAD) is increasingly used in clinical practice and for many applications a multitude of CAD systems have been developed. In practice, CAD systems have different strengths and weaknesses and it is therefore interesting to consider their combination. In this paper, we present generic methods to combine multiple CAD systems and investigate what kind of performance increase can be expected. Experimental results are presented using data from the ANODE09 and ROC09 online CAD challenges for the detection of pulmonary nodules in computed tomography scans and red lesions in retinal images, respectively. For both applications, combination results in a large and significant increase in performance when compared to the best individual CAD system.**

***Index Terms*—Combination, computer-aided detection (CAD), lung, lung nodule, red lesion, retina.**

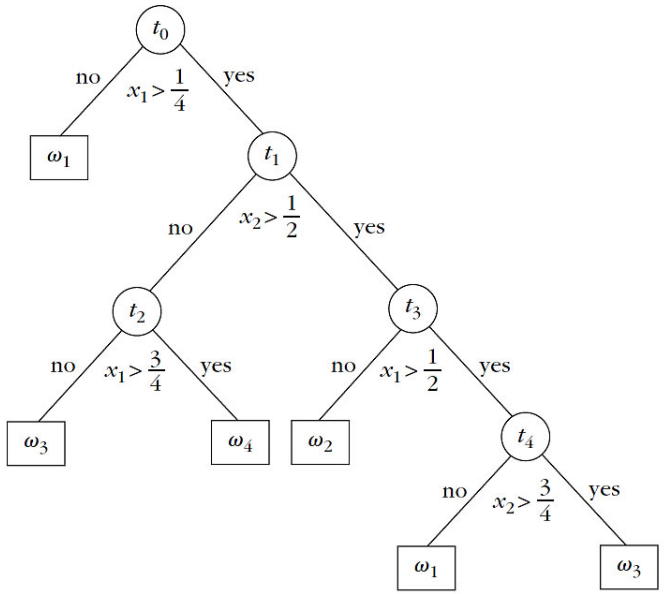
I. Introduction

COMPUTER-AIDED detection (CAD) is increasingly used in clinical practice to assist physicians in the detection of abnormalities. The abnormalities to be detected can be in different stages of development or show morphologically different manifestations of a disease. These factors lead to some CAD systems being able to better detect certain types of abnormalities. As a result, it is reasonable to expect that it can be beneficial to combine multiple CAD systems for a given task. In this work we present generic techniques to combine CAD systems and investigate what kind of performance increase can be expected.

The combination of multiple classification methods to increase performance has been an accepted practice within the area of pattern recognition [2], [3] and forms the basis of established strategies such as boosting [4]. In medicine, the combination of multiple readers is standard practice. Several studies have shown that double reading can improve abnormality detection rates. Two prominent examples are lung cancer detection in chest radiographs and CT scans [5], [6] and cancer detection in mammography [7], [8]. Double reading by technologists has been shown to improve cancer detection rates while maintaining a low recall rate in mammography screening [9]. This shows that even the addition of nonexpert readers can have a

Some Remarks

› Decision trees as classifier combiners…

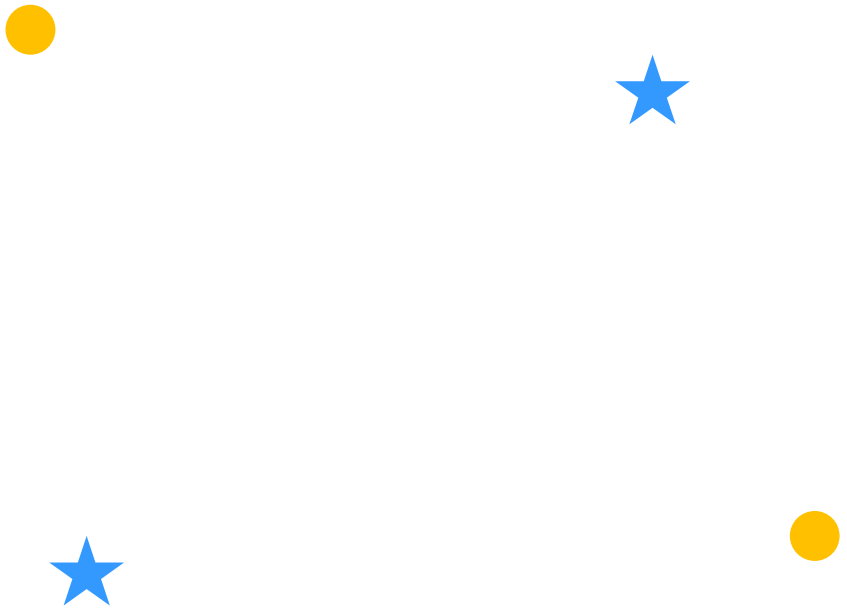


Some Remarks

› Link between classifier combiners and neural networks?

Let us briefly discuss the latter class of learning machines

Archetypical Problem…

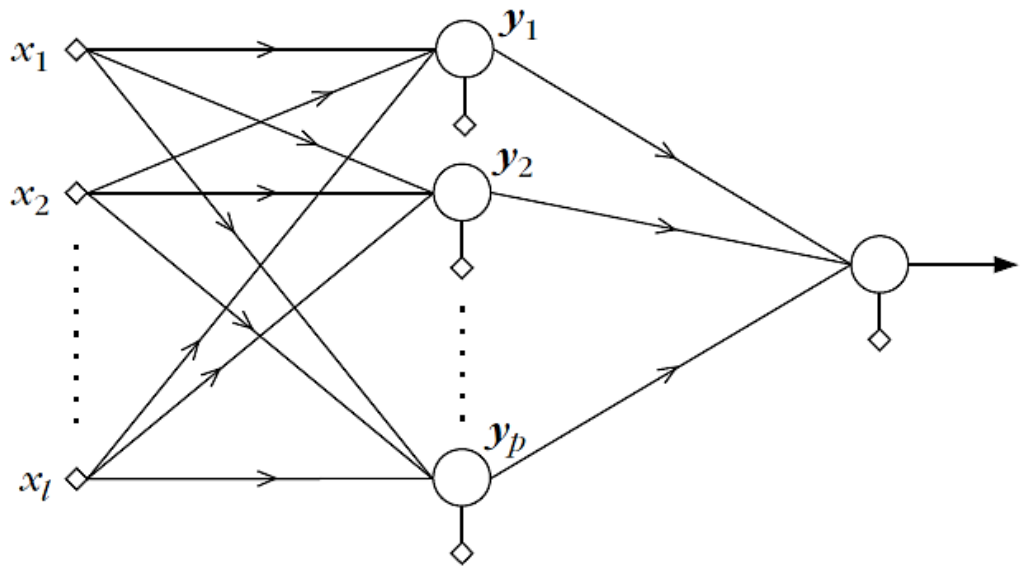


› Will a perceptron solve this problem?

Two-Layer Perceptron / NN

› Ingredients : nodes, layers, activation functions

› How to read the figure?

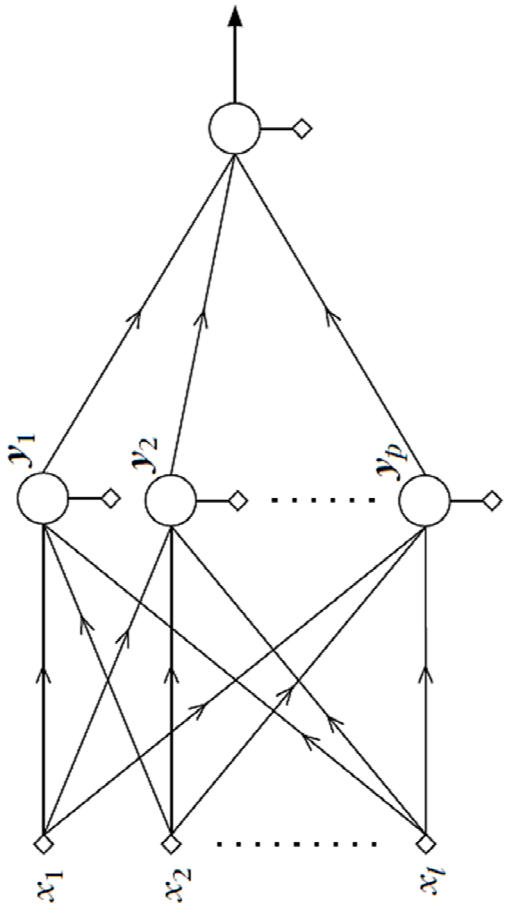


NN Ingredients…

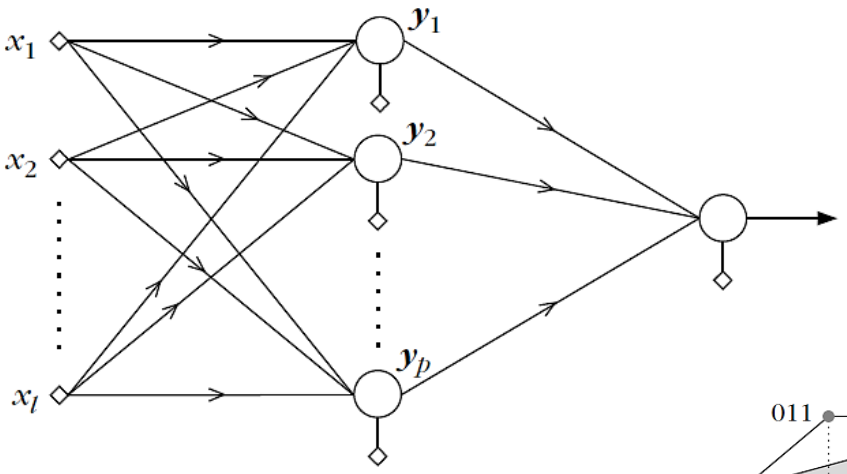
How to Solve XOR?

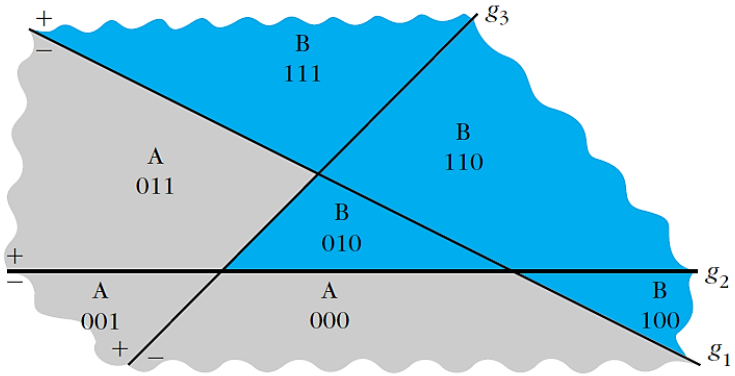
[Consider hard activation / transfer functions, i.e., a step function]

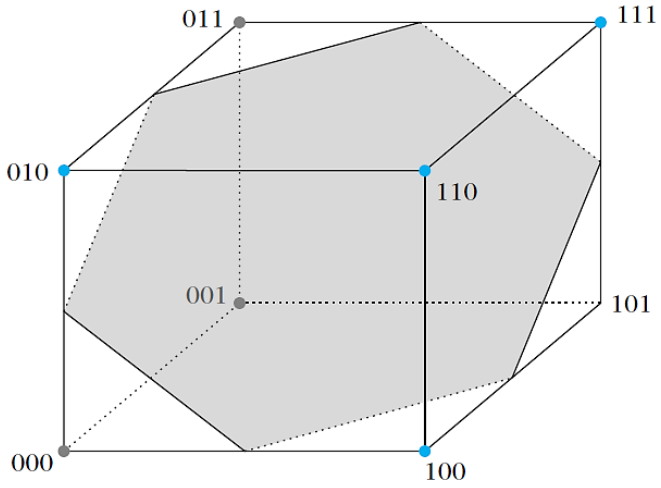
- How many input nodes?
- How many hidden nodes?
- How many output nodes?
- How to set the weights?
- Anything else?



Limitation of Two Layers

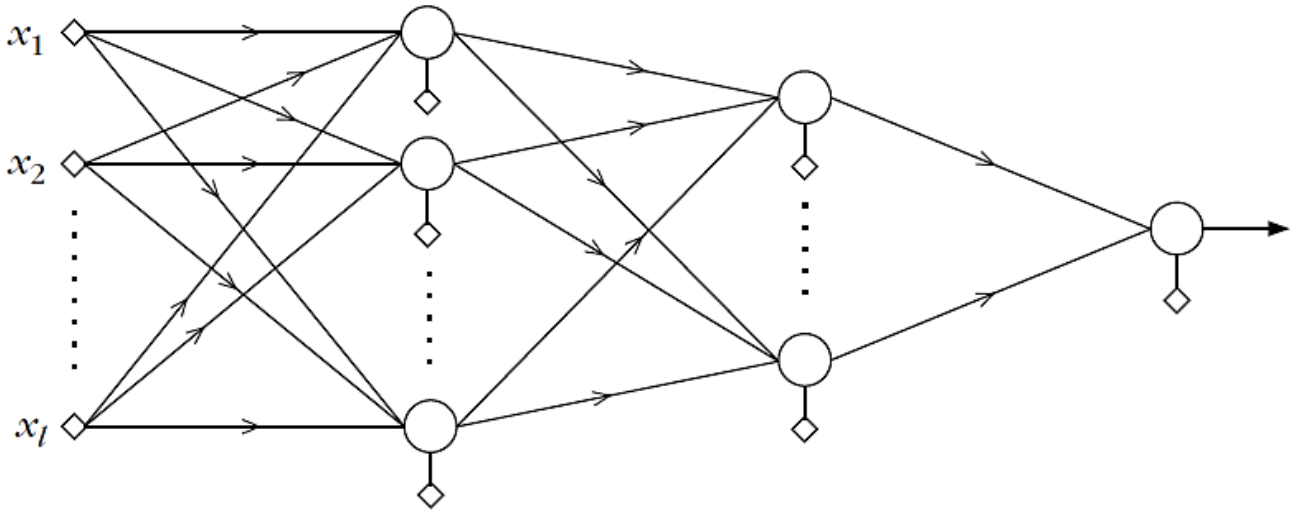






Multilayer Perceptron / NN

› Complex enough network can separate any training set…



How to Train?

› Seen what NNs are capable of

› Q : But how do they learn?

subQ : How does the objective function look?

subQ : Standard way of optimizing a function?

subQ : Problem with hard transfer?

How to Train?

› Seen what NNs are capable of

› Q : But how do they learn?

subQ : How does the objective function look?

subQ : Standard way of optimizing a function?

subQ : Problem with hard transfer?

› A : Backpropagation

[= just clever gradient descent]

Remarks

› Many[!] variations to NNs and their training

› More general cost functions can be optimized

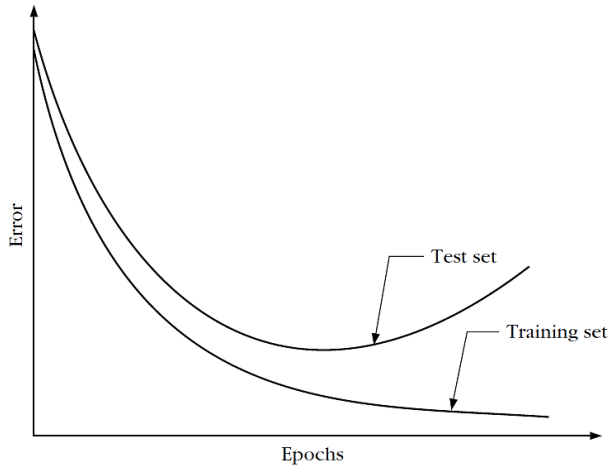
› NN as ultimate trained combiner?

› Some of main NN problems

Getting stuck in local optima

Choice of network topology

Danger of overtraining
[a problem in general!!!]

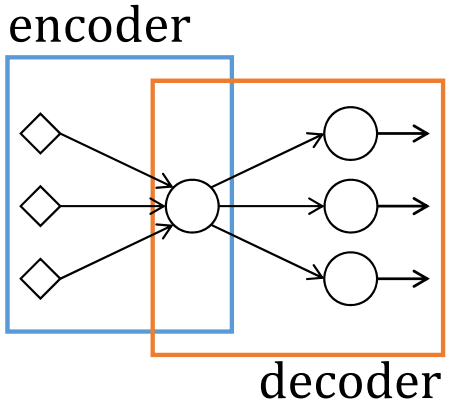


One more point :

Back to feature extraction

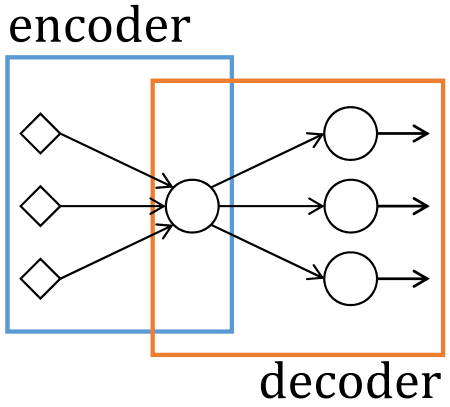
Autoencoding Neural Network

- Feedforward neural networks predicting their input, i.e., input = output



- Elementary form
  - Linear transfer functions; single bottleneck layer
  - Minimize squared Euclidean distance in and out
  - Trained like standard NNs
  - What mapping does encode give?

Autoencoding Neural Network



› Feedforward neural networks predicting their input, i.e., input = output

› More complex form
⇒ more complex dimensionality reducer
  - Nonlinear transfer functions
  - Possible other optimization criterion
  - More layers

Wrap Up

- DTs and ANNs in context of combining
  - Both also provide additional nonlinear classifiers
- Large number of combiners, fixed and trained
  - …leading to immense possibility of "new" learners
  - Well-known are decision forests, random forests, boosting
- Boosting idea of adaptation generally appealing
- Autoencoding ANNs for dimensionality reduction

⟩ Q?