Jaime S. Cardoso
Associate Professor
jaime.cardoso@inesctec.pt
http://www.inescporto.pt/~jsc/

http://medicalresearch.inescporto.pt/breastresearch/

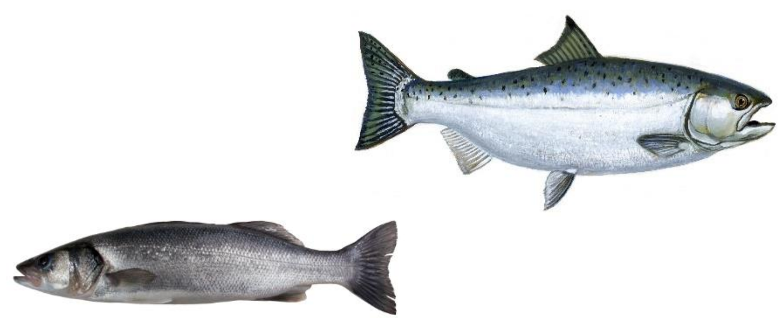
INESC TEC and Faculdade de Engenharia, Universidade do Porto, Portugal

# Deep Learning: Variations on the Theme and Current Trends

**Data Science Portugal**
**Feb 21st, 2017, Portugal**

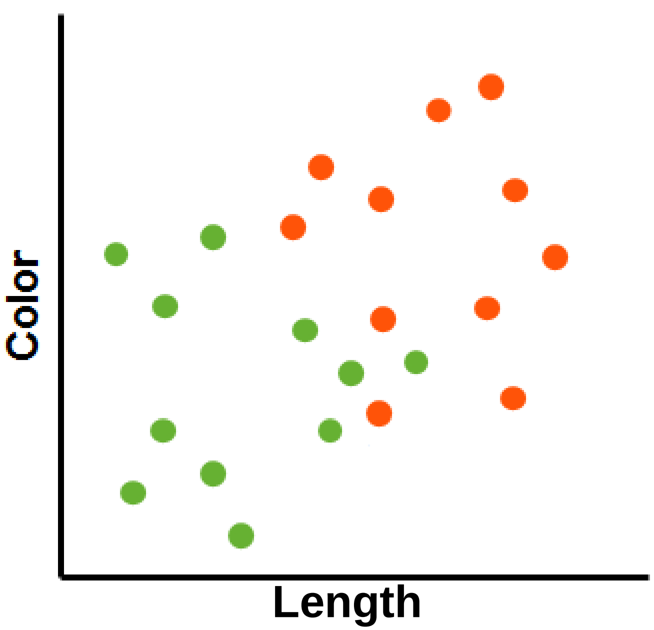
# Data Driven Design

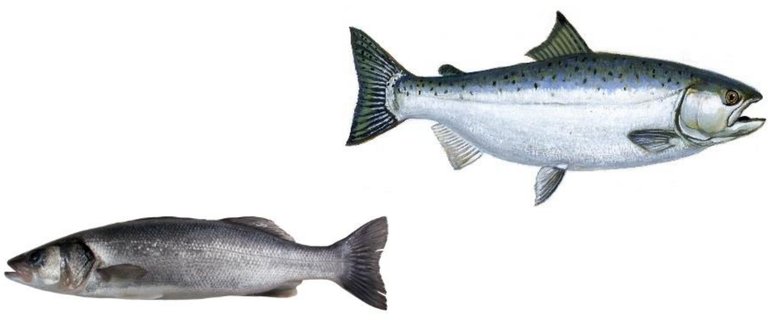
- When to use?
  - Difficult to reason about a generic rule that solves the problem
  - Easy to collect examples (with the solution)



# Data Driven Design

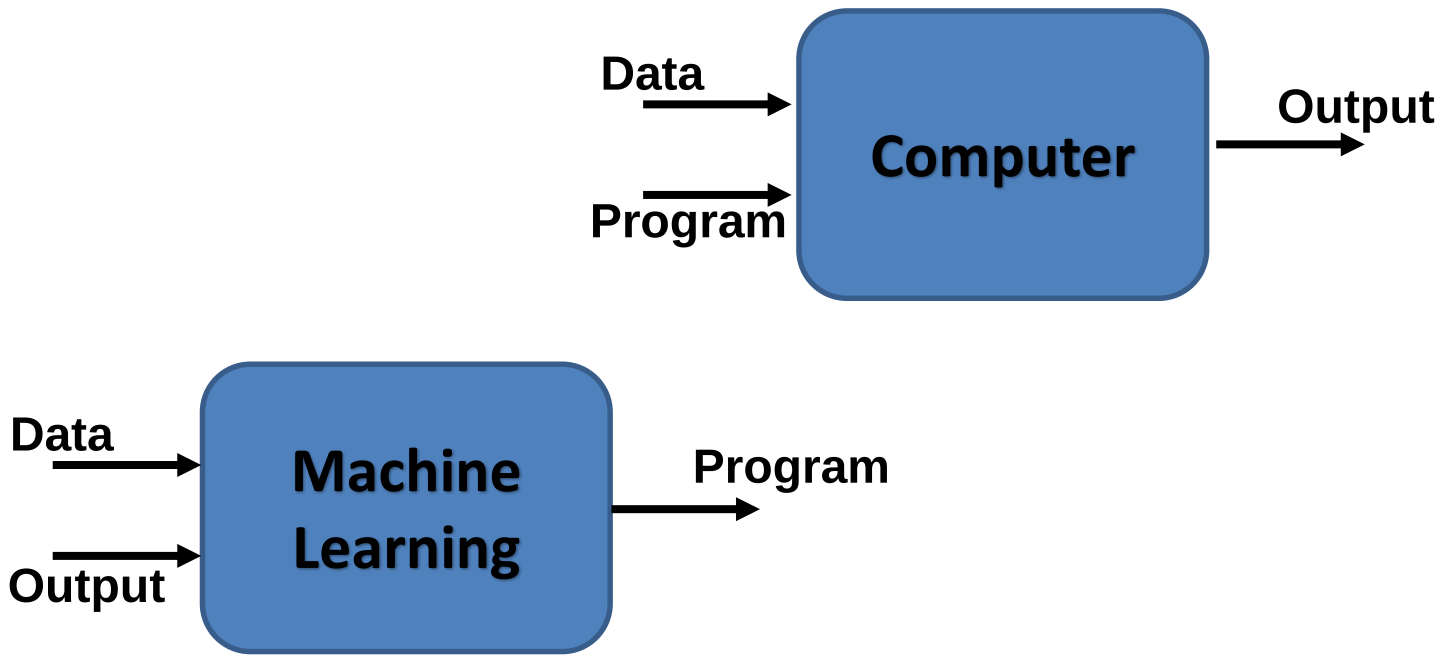
- When to use?
  - Difficult to reason about a generic rule that solves the problem
  - Easy to collect examples (with the solution)



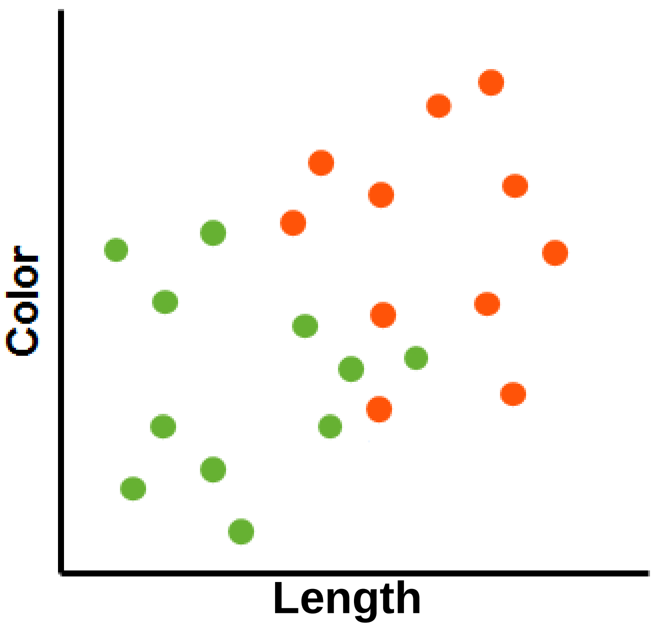
# What is Machine Learning?

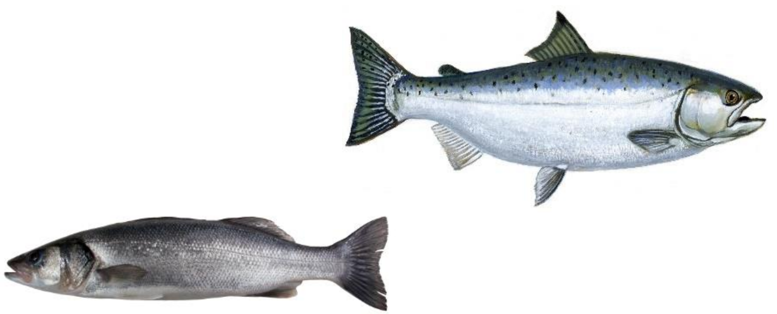
- Automating the Automation



# Data Driven Design

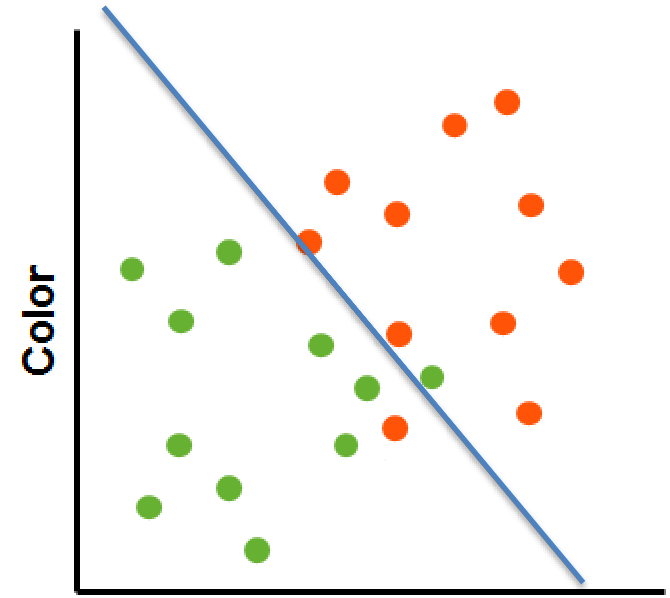
- When to use?
  - Difficult to reason about a generic rule that solves the problem
  - Easy to collect examples (with the solution)

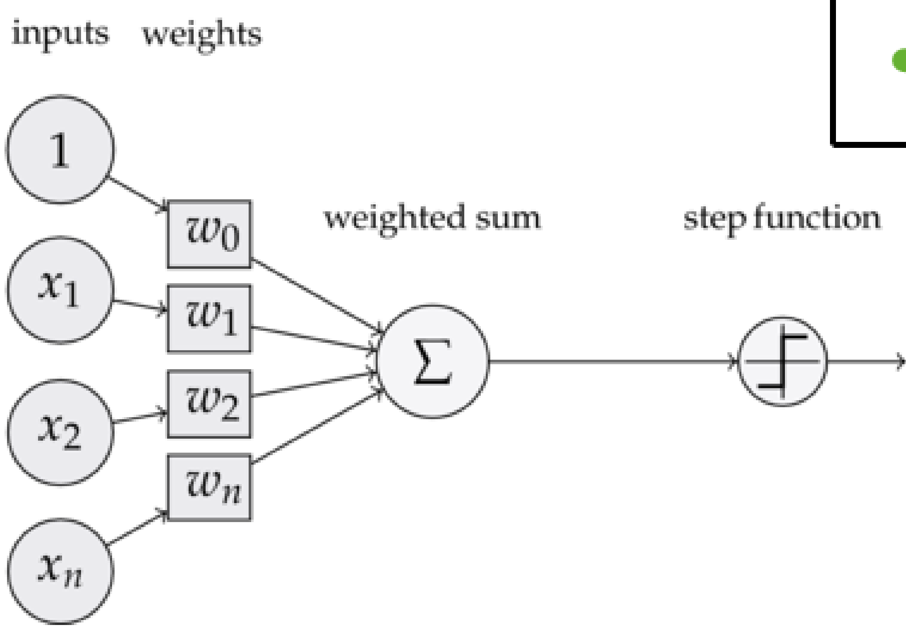


# Do we need deep learning?

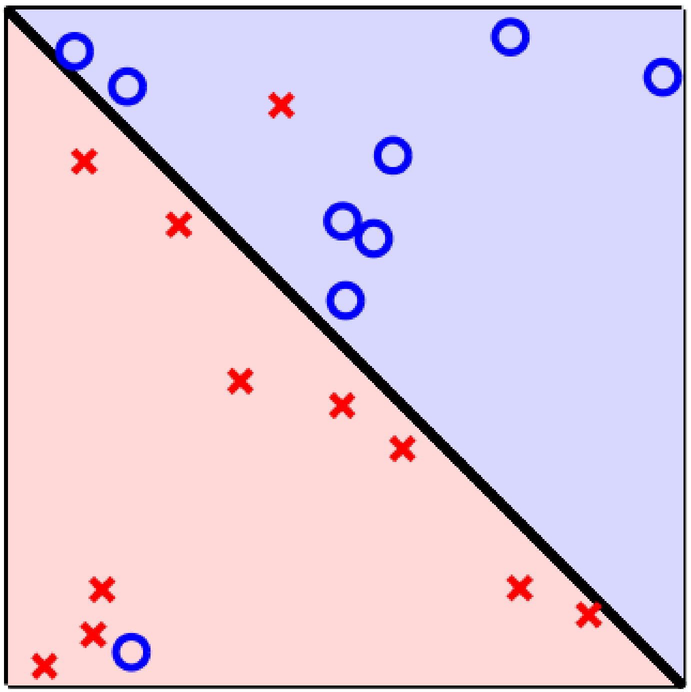
- For simple (linear) classification problems?



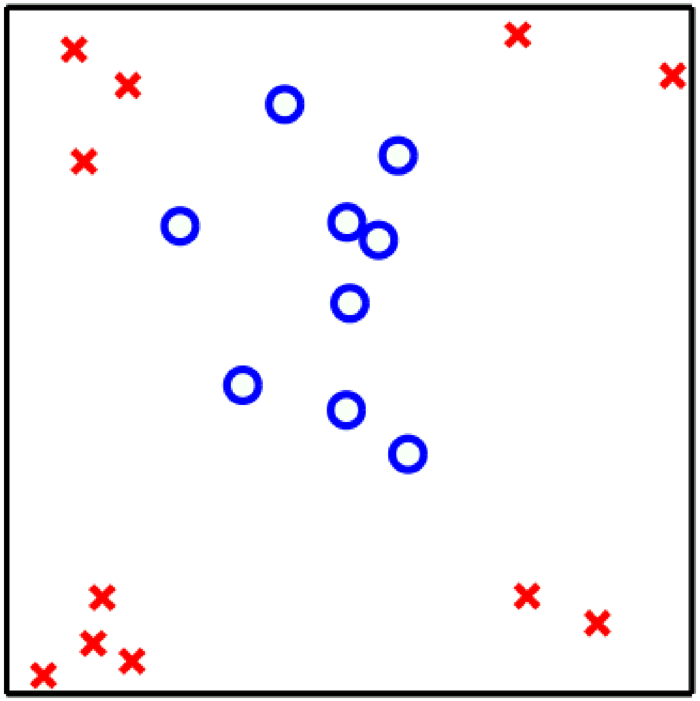



# Do we need deep learning?

- The Linear Model has its Limits



(a) Linear with outliers

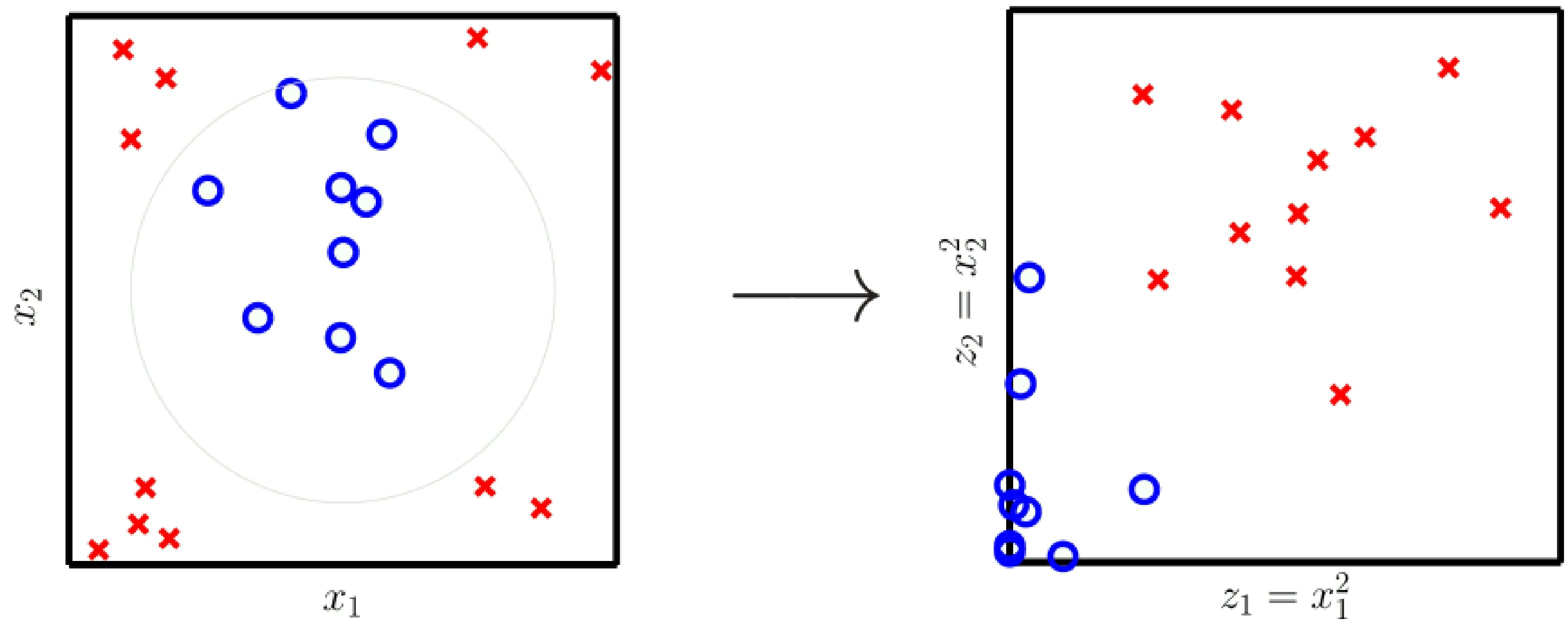


(b) Essentially nonlinear

To address (b) we need something more than linear.

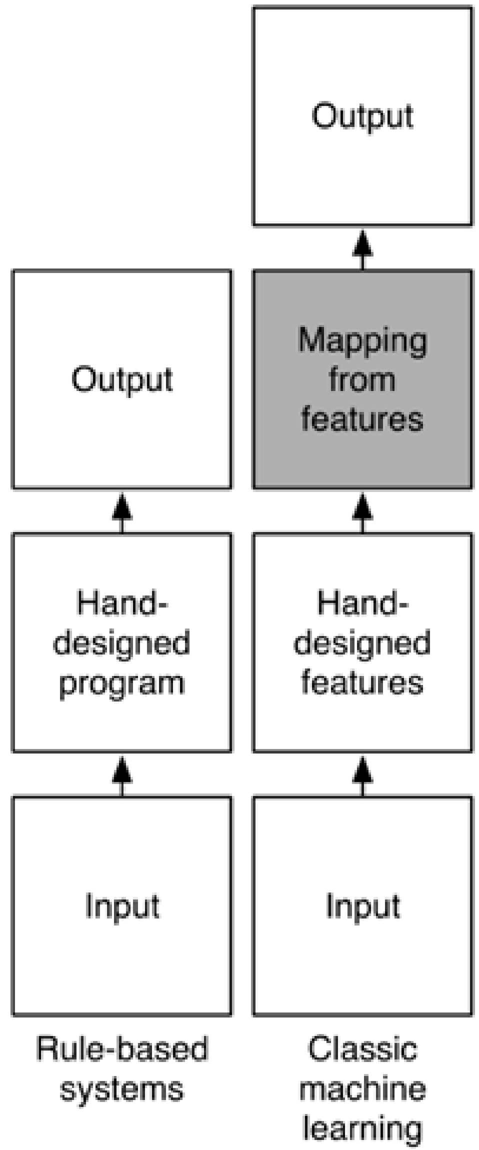
# Do we need deep learning?

- Change Your Features Using a Transform




$$\mathbf{x} = \begin{bmatrix} 1 \\ x_1 \\ x_2 \end{bmatrix} \longrightarrow \mathbf{z} = \mathbf{\Phi}(\mathbf{x}) = \begin{bmatrix} 1 \\ x_1^2 \\ x_2^2 \end{bmatrix} = \begin{bmatrix} 1 \\ \Phi_1(\mathbf{x}) \\ \Phi_2(\mathbf{x}) \end{bmatrix}$$

# Classic Machine Learning

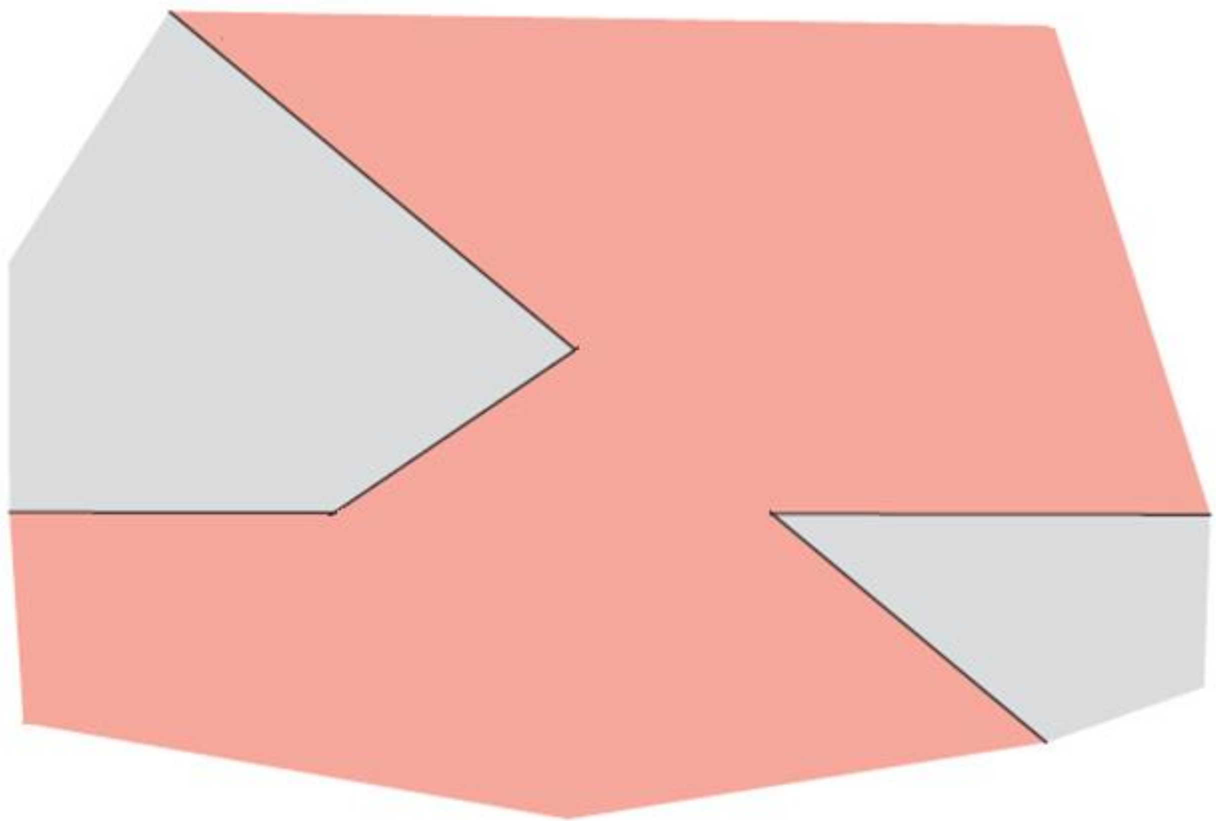


# Use the Linear Model?

- First try a linear model – simple, robust and works.
- Algorithms can tolerate error plus you have nonlinear feature transforms.
- Choose a feature transform before seeing the data. Stay simple.
- **Data snooping** is hazardous to your generalization.
- Linear models are fundamental in their own right; they are also the building blocks of many more complex models like support vector machines.
- Nonlinear transforms also apply to regression and logistic regression and …

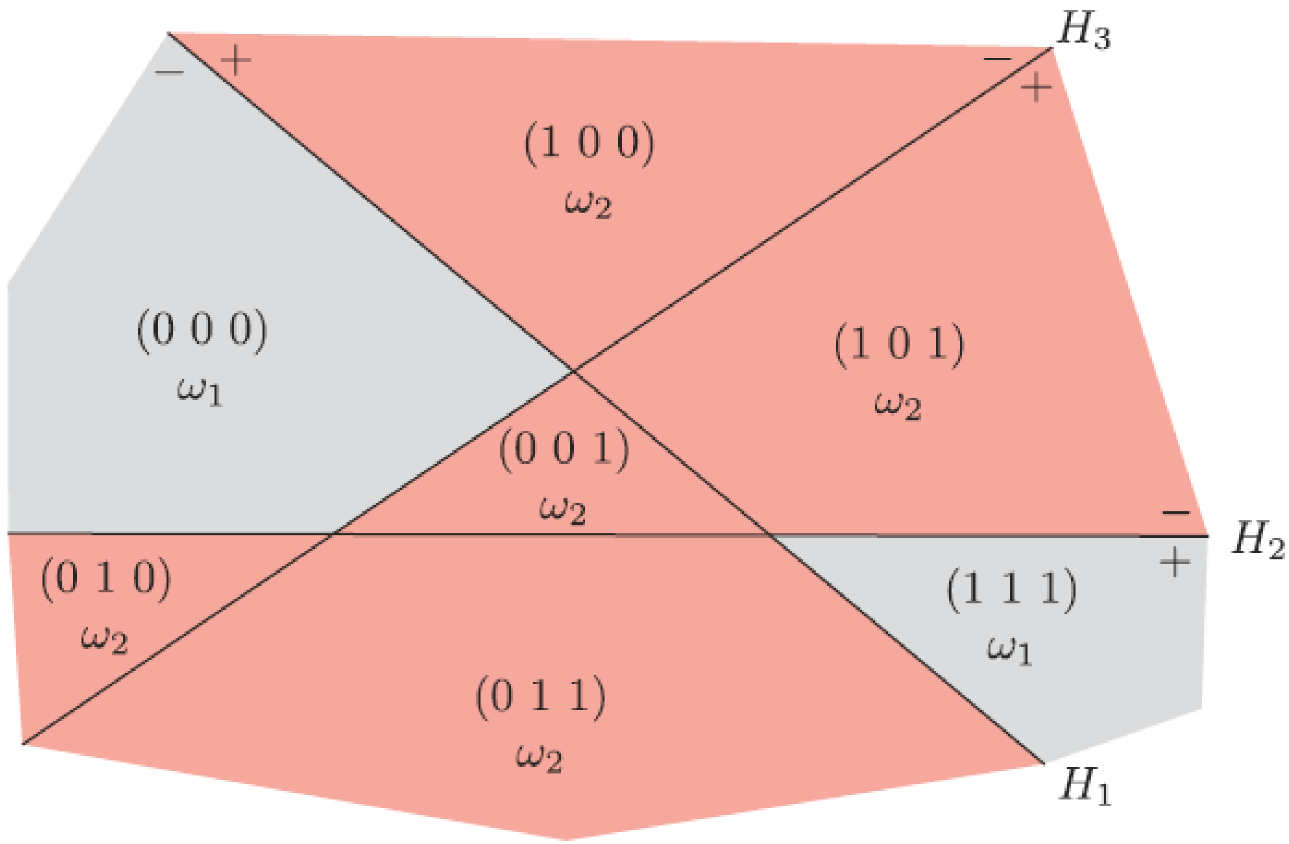
# Do we need deep learning?

- For complex binary classification problems?



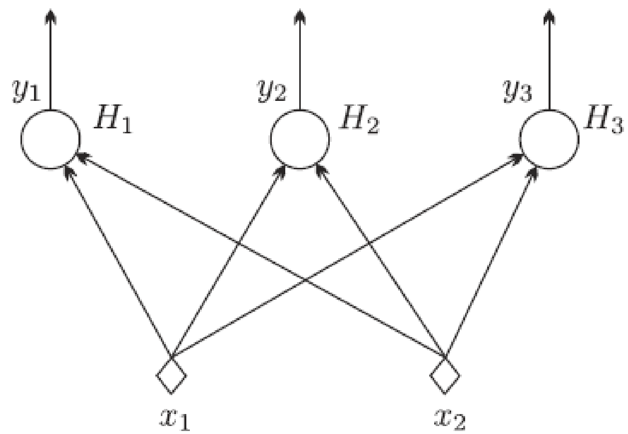
# Do we need deep learning?

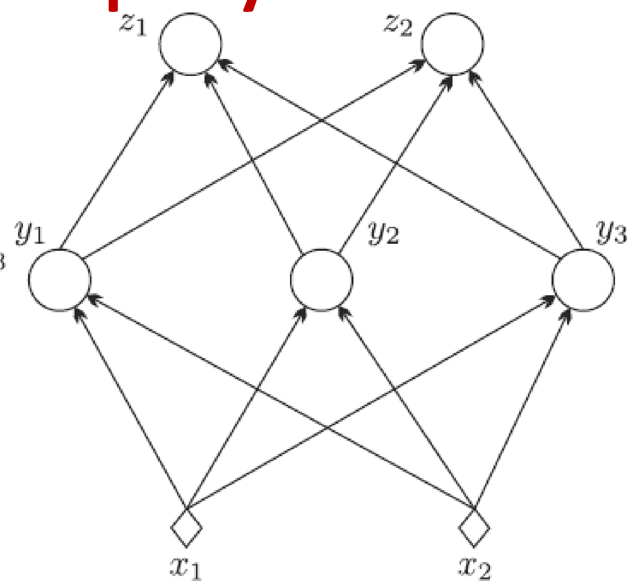
- For complex binary classification problems?

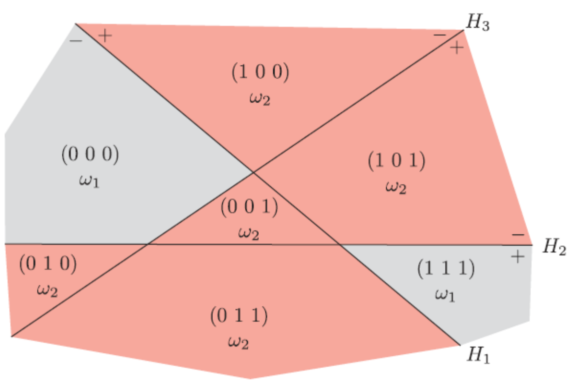


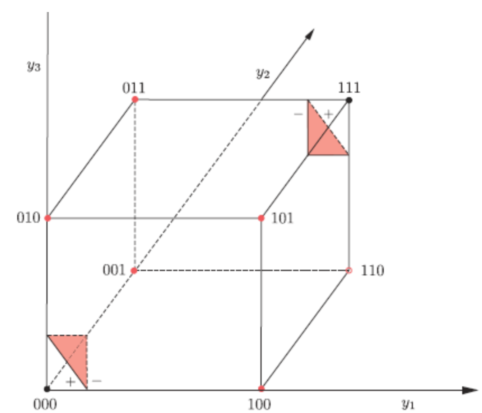
# Do we need deep learning?

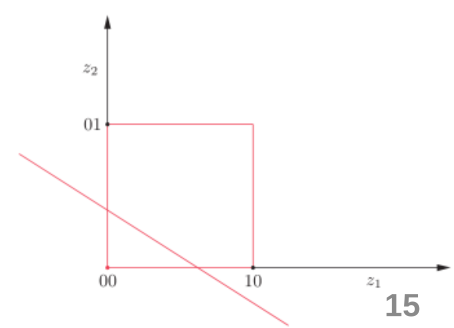
- For complex binary classification problems?
  - No: **Universal Approximation Property of Feed-forward Networks**



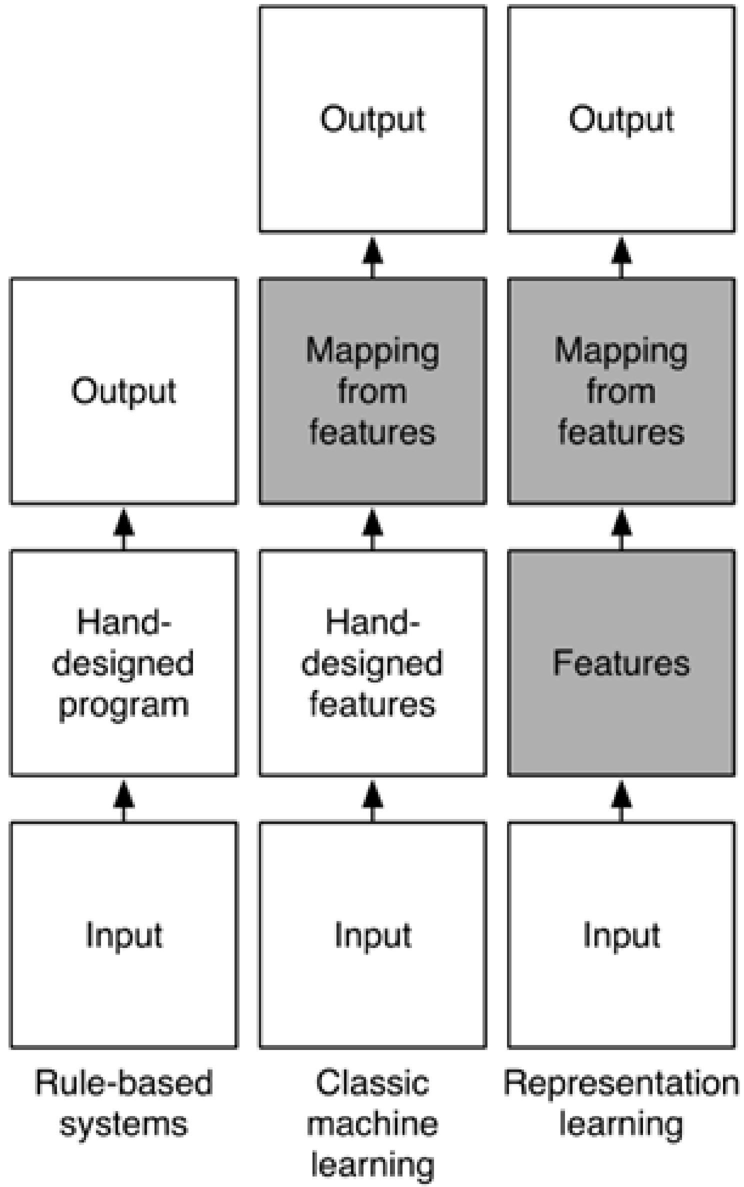



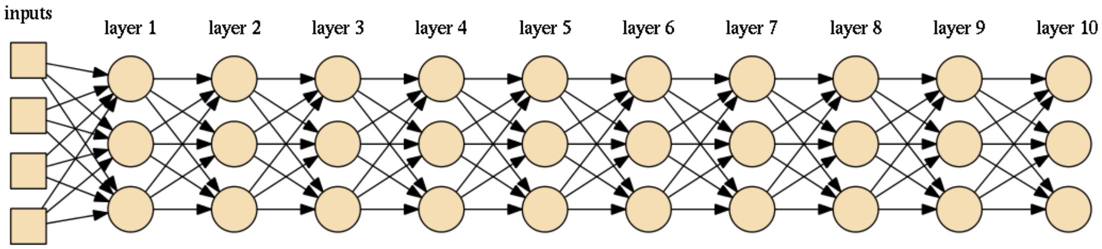



# Representation Learning



# Do we need deep learning?

- For complex binary classification problems?
  **Yes**
  - In any learning task, we have to be concerned with what is feasibly "learnable" in a given representation.
  - Using networks with more layers, one can obtain more **compact representations** of the input-output relation.
    - We say that a network is **compact** if it consists of relatively few free parameters (few computational elements) to be learned/tuned during the training phase.
    - For a given number of training points, we expect compact representations to result in better generalization performance.
  - For complex tasks, where more complex concepts have to be learned, for example, recognition of a scene in a video recording, language and speech recognition, the underlying functional dependence is of a very complex nature so that we are unable to express it analytically in a simple way.

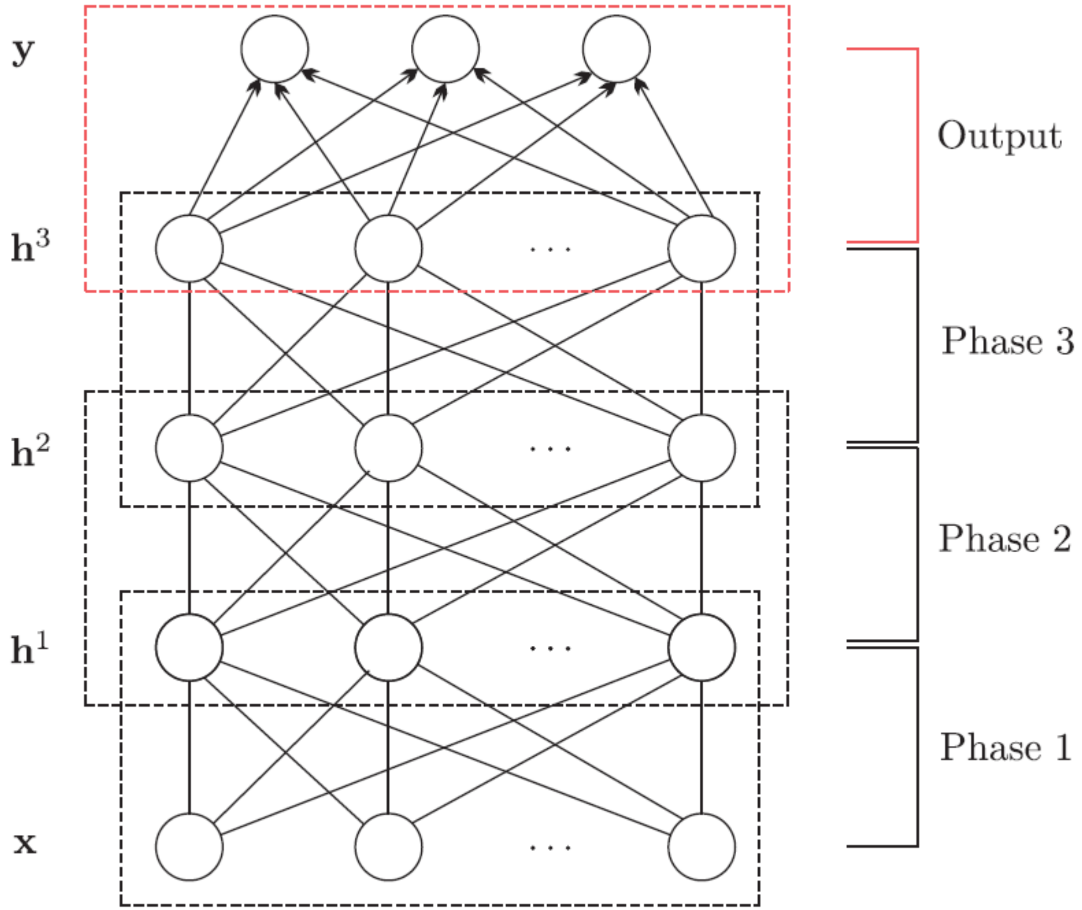


# Training Deep Neural Networks

- Top-down Training
  - Training with backpropagation can become difficult and often algorithms are stuck in local minima.
    - Is there a training scheme, beyond or complementary to the backpropagation algorithm, to assist the optimization process to settle in a "good" local minimum, by extracting and exploiting more information from the input data?
- Can we **train the representation** without using top-down supervision?

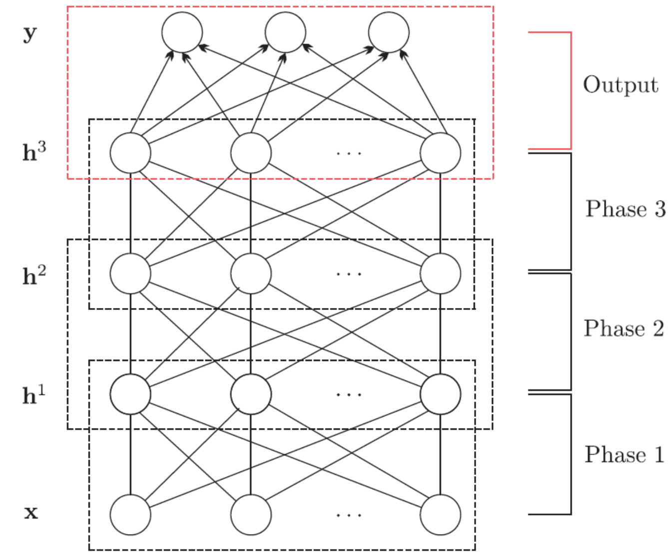
# Training Deep Neural Networks

- The main idea is to pre-train each layer, via an **unsupervised** learning algorithm, **one layer at a time**, in a greedy-like rationale.

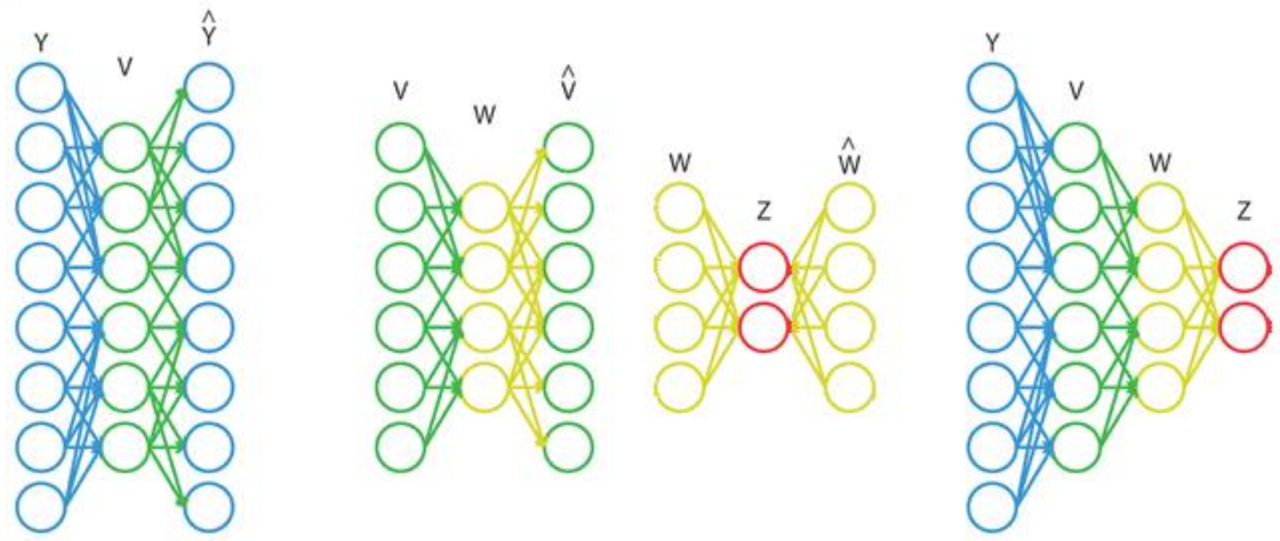


# Training Deep Neural Networks: **Different flavours**

- Using Stacked AutoEncoders (SAE)

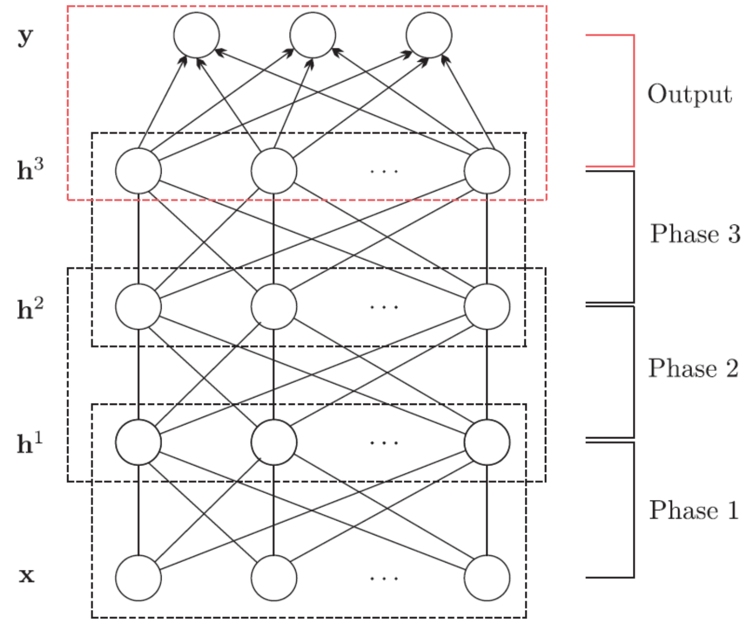



**Training a SAE**

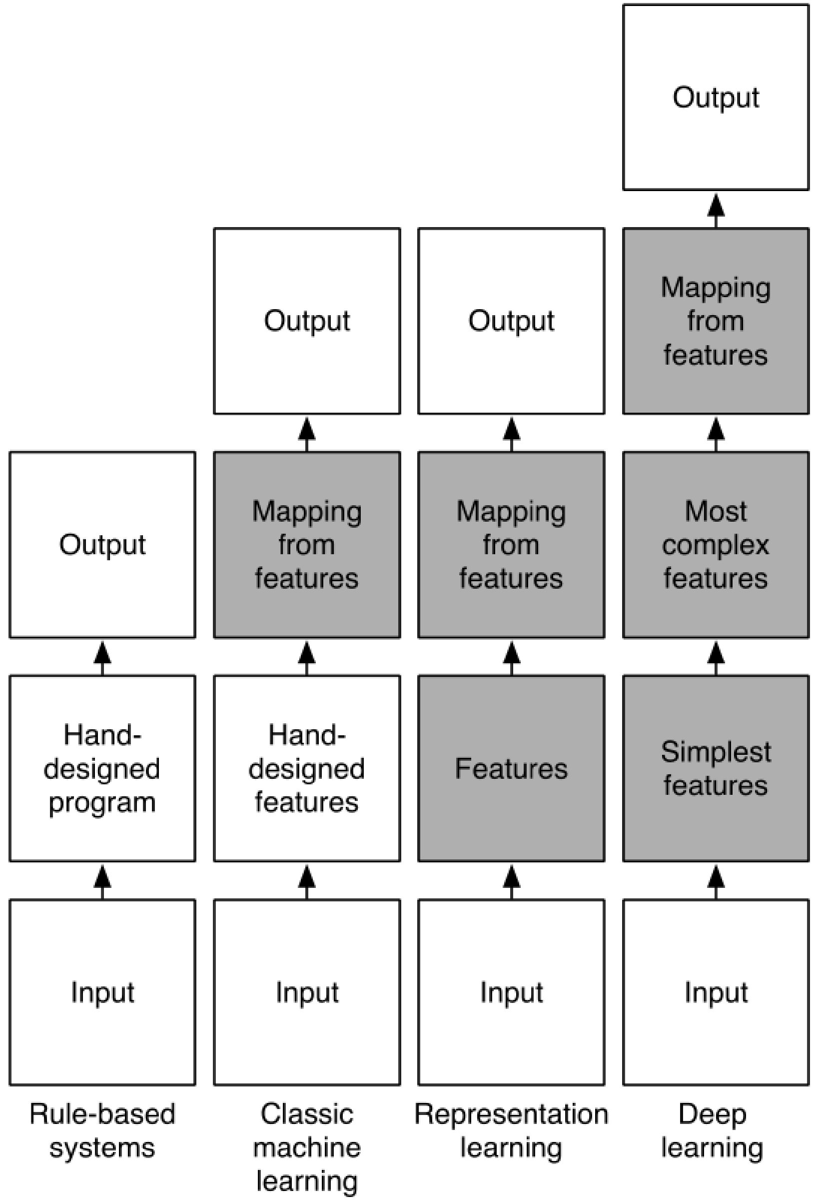


# Training Deep Neural Networks: **Different flavours**

- Training Deep Feed-Forward Networks
  - Step I: Unsupervised Pre-training of Hidden Units
    - Stacked Autoencoders:
      - Phase 1
      - Phase 2
      - Phase 3
  - Step II: Supervised Pre-training of Output Nodes
  - Step III: Fine-Tuning of All Nodes via Supervised Training

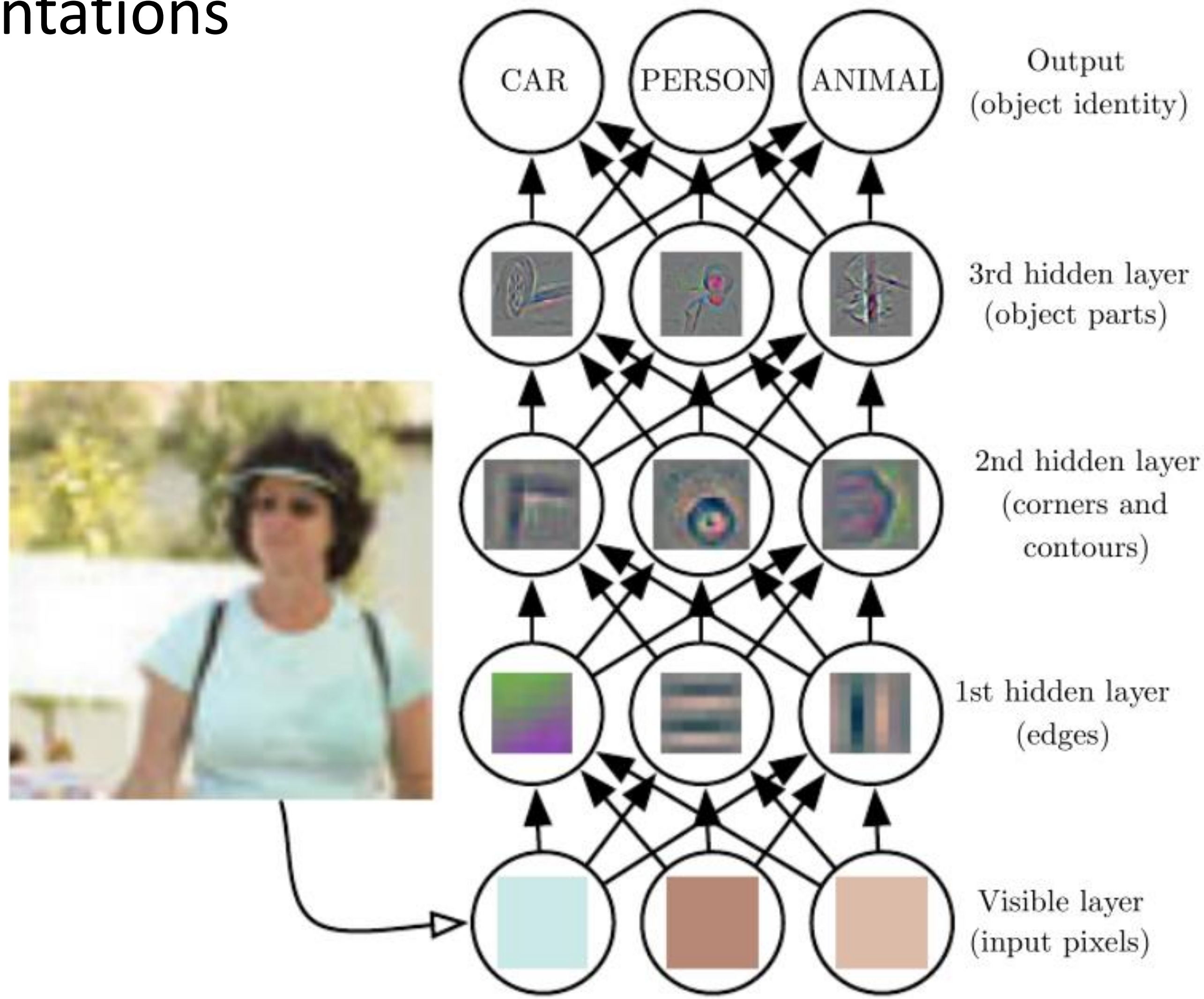


# Why is Deep Learning working so well?



# Why is Deep Learning working so well?

- Successive model layers learn deeper intermediate representations



# Why is Deep Learning working so well?

Four key ingredients

1. Lots & lots of data
2. Very flexible models
3. Enough **computing power**
4. Powerful priors that can defeat the curse of dimensionality

# Why Unsupervised (Deep) Representation?

- Potential benefits:
  - Exploit tons of unlabeled data
  - Answer new questions about the variables observed
  - Regularizer – transfer learning – domain adaptation
  - Easier optimization (local training signal)
  - Structured outputs

How do humans generalize from very few examples?

- They transfer knowledge from previous learning:
  - Representations
  - Explanatory factors
- Previous learning from:
  unlabeled data + labels for other tasks

# Transfer Learning

- Transfer learning (TL) aims to extract knowledge from at least one source task and use it when learning a predictive model for a new target task.
    - The intuition behind this idea is that learning a new task from related tasks should be easier (faster, with better solutions or with less amount of labelled data) than learning the target task in isolation.
  - transferring data, namely, strategically including data from the source task in the target dataset
  - adapting knowledge instead of data. This idea is handled by parameter transfer approaches, which rely on the idea that individual models for related tasks should share some structure (parameters or hyper-parameters)

# Transfer Learning

- In practice, very few people train an entire Convolutional Network from scratch (with random initialization). Instead, it is common to pretrain a ConvNet on a very large dataset
  – ConvNet as fixed feature extractor
  – Fine-tuning the ConvNet

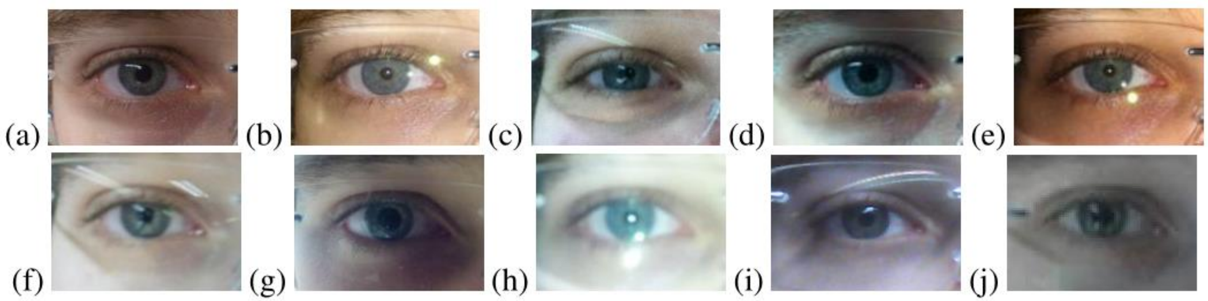
# Transfer Learning

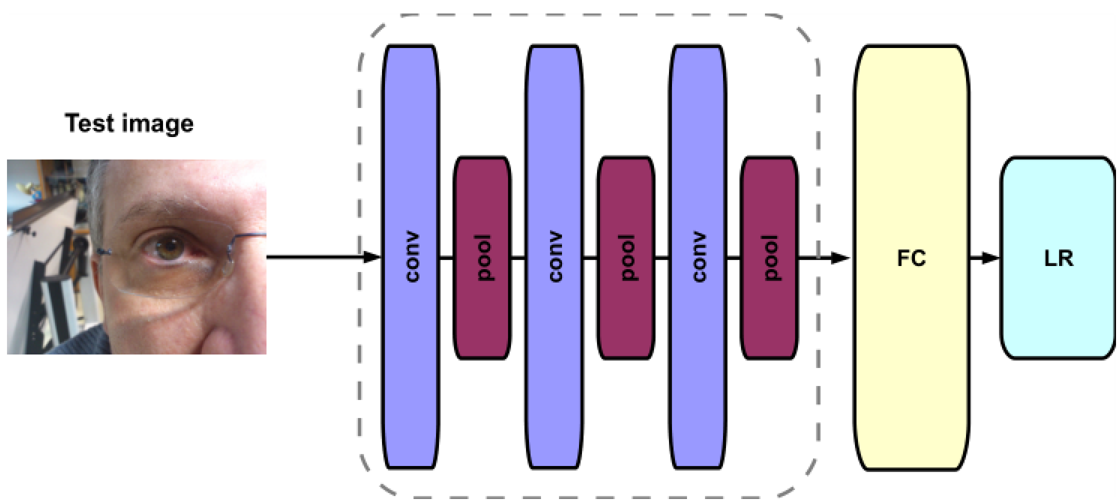
- In practice, very few people train an entire Convolutional Network from scratch
  - Example: Face Recognition

  - Use VGG Deep Net (trained with millions of faces – but for a different task) to extract a feature vector by removing the output layer
  - Use euclidean distance (or variations) for matching
    - Improves over state of the art approaches in several settings.

# Transfer Learning

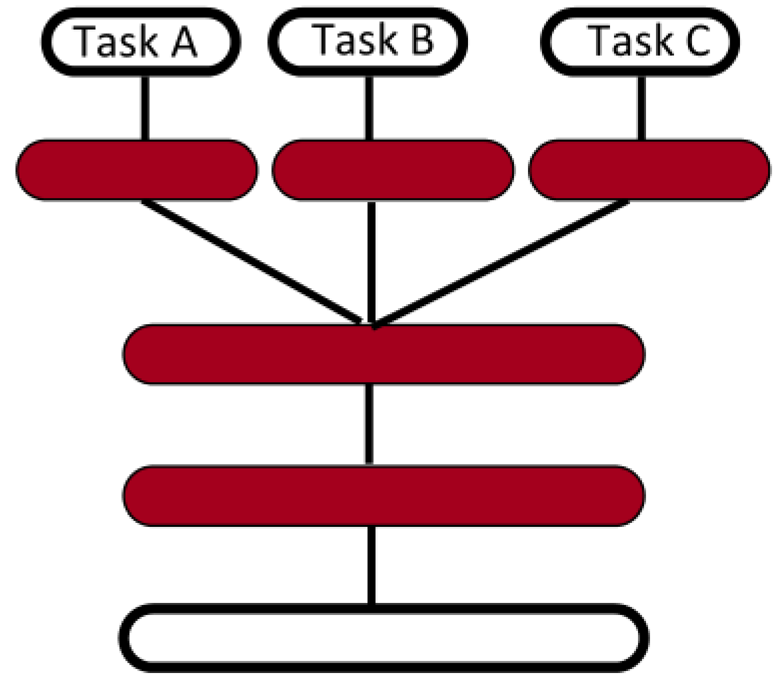
- Cross Sensor Adaptation for periocular recognition



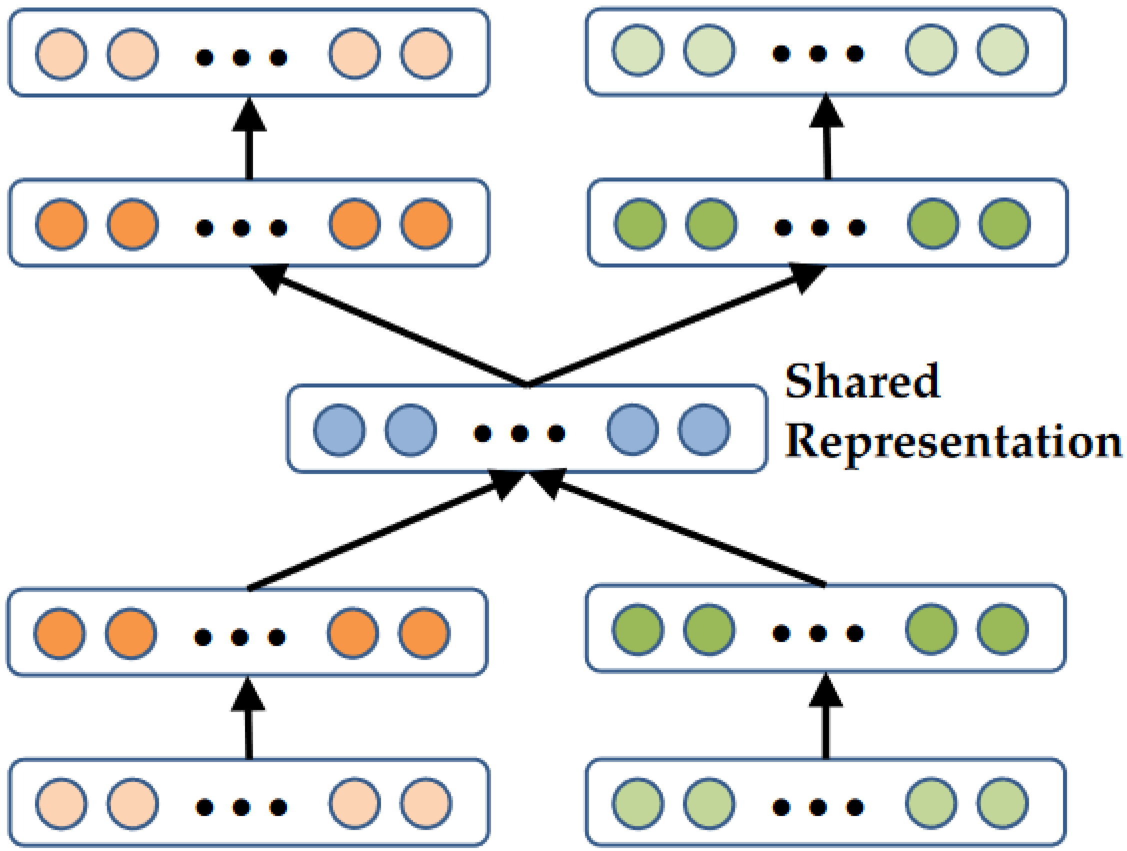
# Multi-Task Learning

- Generalizing better to new tasks (tens of thousands!) is crucial to approach AI
- Example: speech recognition, sharing across multiple languages
- Deep architectures learn good intermediate representations that can be shared across tasks

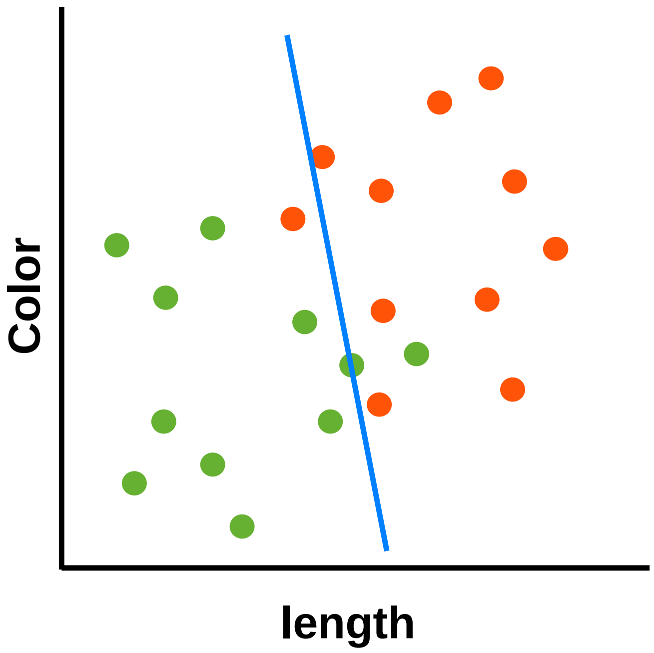



**E.g. dictionary, with intermediate concepts re-used across many definitions**

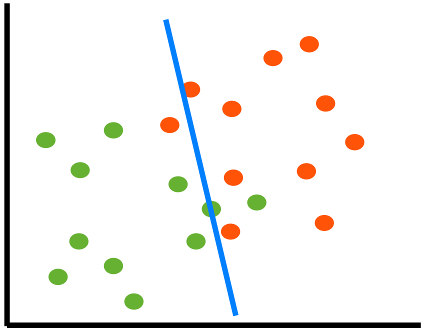
# Multimodal Deep Learning

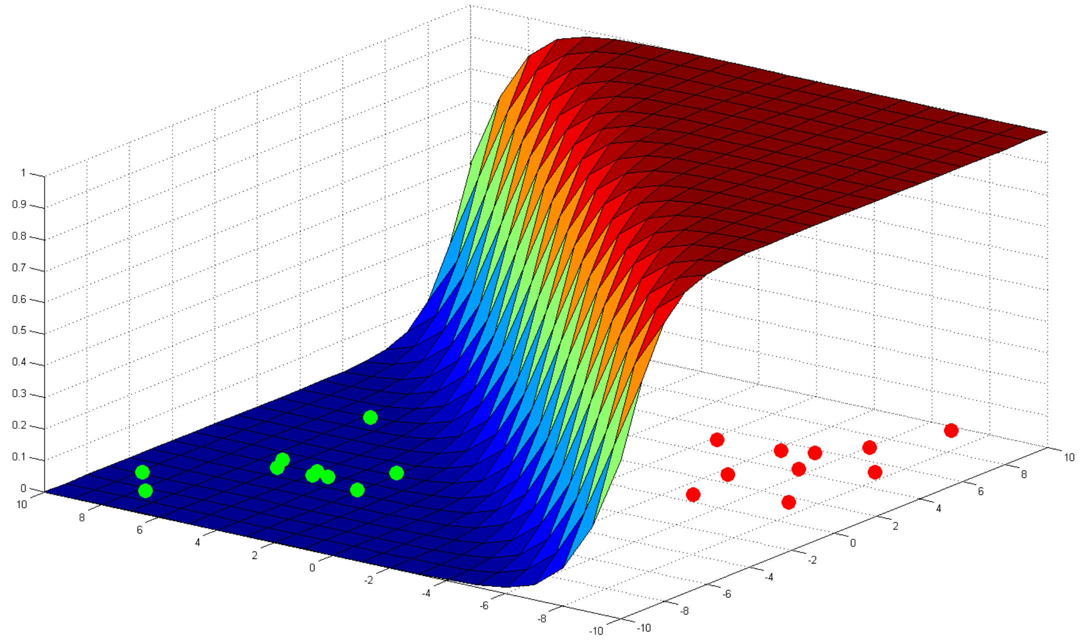


# Design of a Classifier

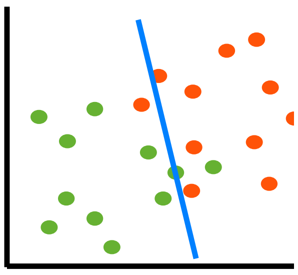


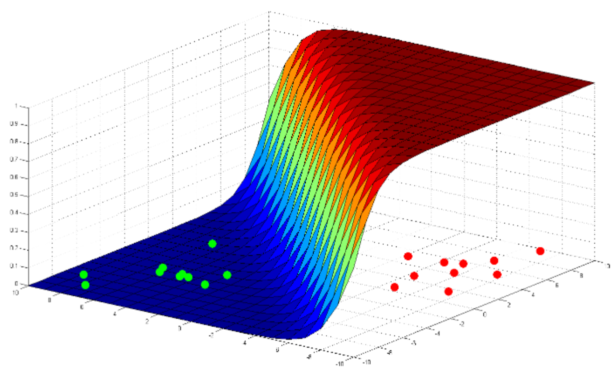
# Design of a Classifier

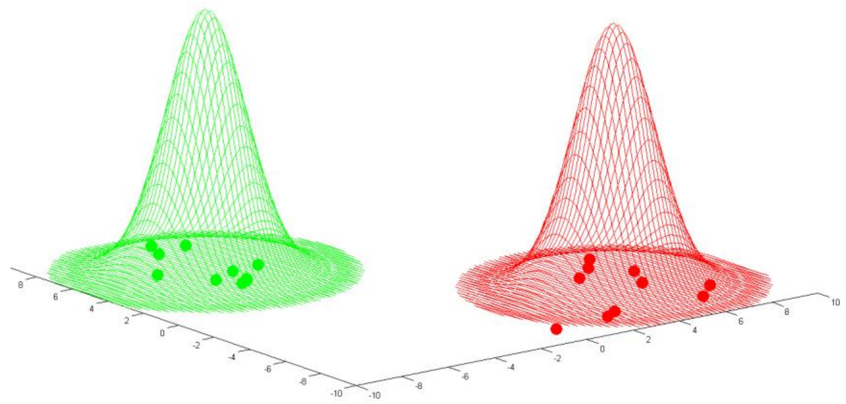




# Design of a Classifier

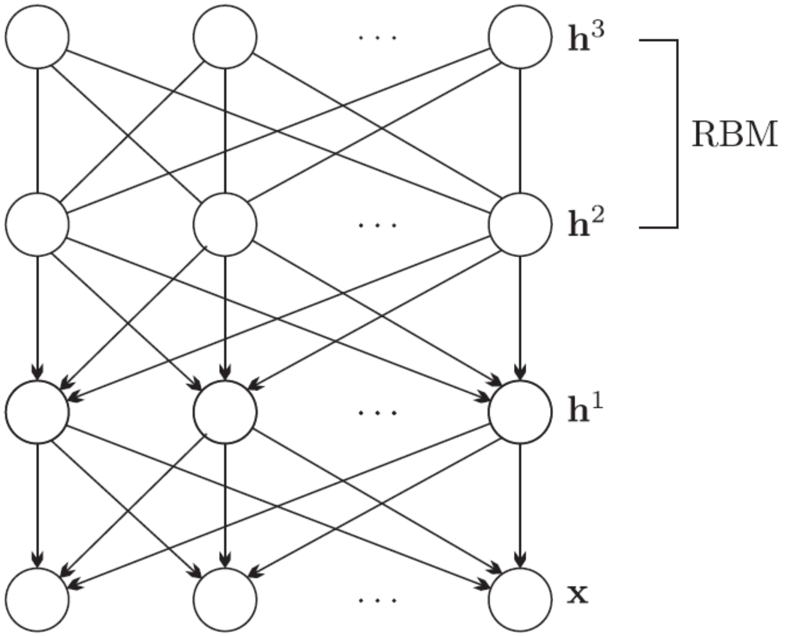




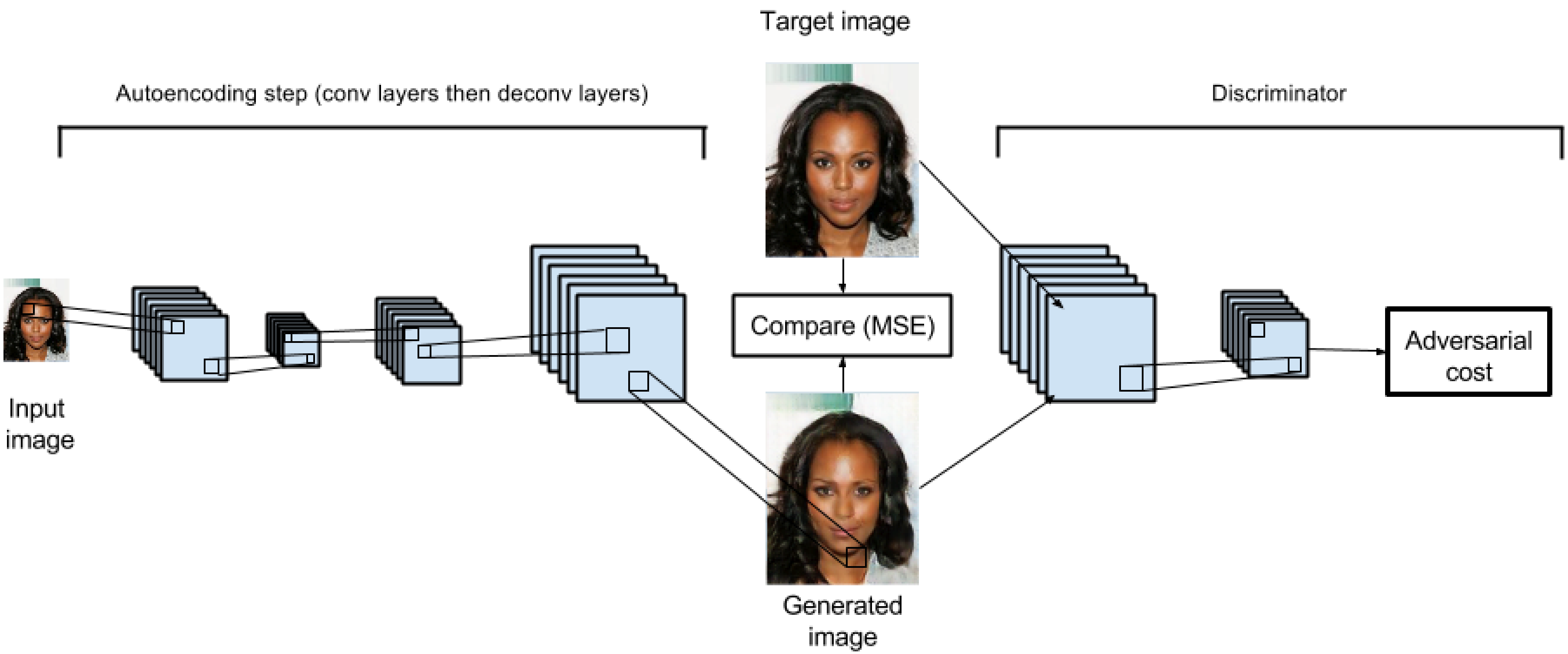


# Deep Belief Networks

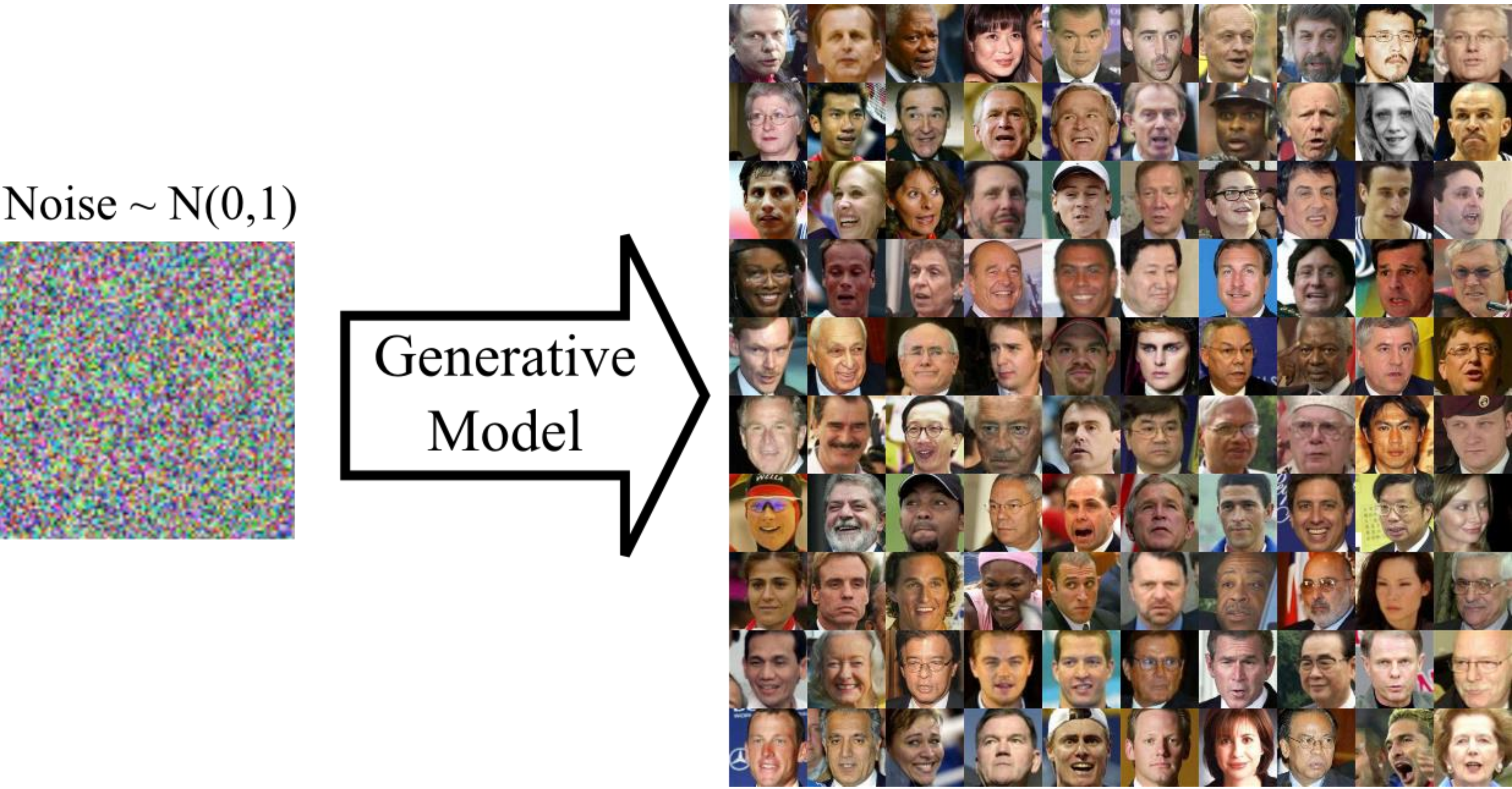
- We focused our discussion on deep learning on multilayer perceptrons for supervised learning.
  - Focus on the information flow in the feed-forward or bottom-up direction.
- **Can we start from the last layer, corresponding to the most abstract representation, and follow a top-down path with the new goal of generating data?**
  - **Studies suggest that such top-down connections exist in our visual system to generate lower level features of images starting from higher level representations.**
  - **Important for disambiguating effect on the interpretation of local image regions by providing contextual prior information from previous frames**



# Adversarial Neural Network



# Adversarial Neural Network



# Ideas

- Ideally: the Master Algorithm
  - Given
    - data
    - learning bias / prior
  - Does the job
- Deep Models: a step in the right direction?
  - A lot of parameters
    - Difficult to guide the learning in the 'right direction'
    - Difficult to avoid overfitting
      - Presently we are just overfitting to larger datasets
      - Vulnerable to **adversarial examples**

# References (slides/images/technical info)


- http://cs231n.stanford.edu/
  - http://cs231n.github.io/convolutional-networks/
  - http://cs231n.github.io/transfer-learning/
- Machine Learning A Bayesian and Optimization Perspective
- Bengio's webpage
  - http://www.iro.umontreal.ca/~bengioy/yoshua_en/index.html
- Deep Learning, Ian Goodfellow, Yoshua Bengio, Aaron Courville
- Jaime Cardoso's Webpage
  - http://www.inescporto.pt/~jsc/


**THANK YOU**