Multilayer Networks

Léon Bottou

COS 424 – 3/11/2010

Agenda

Goals	Classification, clustering, regression, other.
Representation	Parametric vs. kernels vs. nonparametric Probabilistic vs. nonprobabilistic Linear vs. nonlinear Deep vs. shallow
Capacity Control	Explicit: architecture, feature selection Explicit: regularization, priors Implicit: approximate optimization Implicit: bayesian averaging, ensembles
Operational Considerations	Loss functions Budget constraints Online vs. offline
Computational Considerations	Exact algorithms for small datasets. Stochastic algorithms for big datasets. Parallel algorithms.

Summary

1. Brains and machines.

2. Multilayer networks.

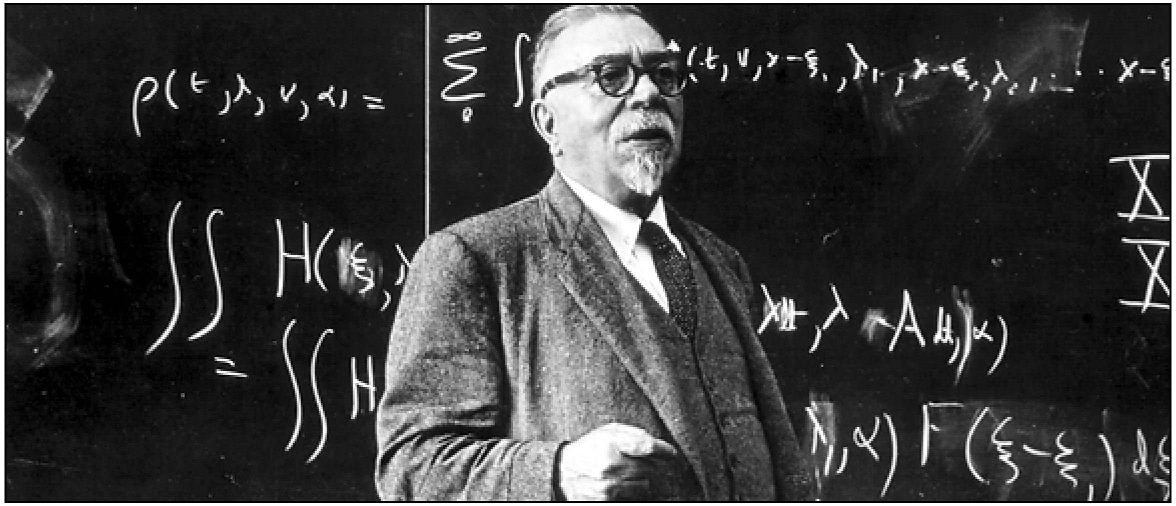
3. Modular back-propagation.

4. Examples

5. Tricks

Cybernetics

Mature communication technologies: telegraph, telephone, radio, . . .
Nascent computing technologies: Eniac (1946)



Norber Wiener (1948)
Cybernetics or Control and Communication in the Animal and the Machine.

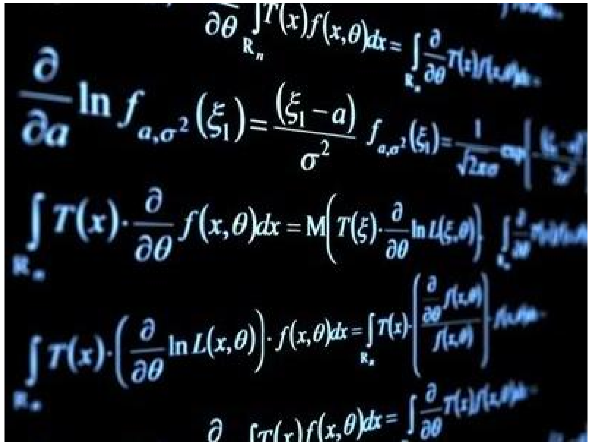
Redefining of the man–machine boundary.

What should a computer be?

A universal machine to process information.
– which structure? what building blocks?
– which model to emulate?

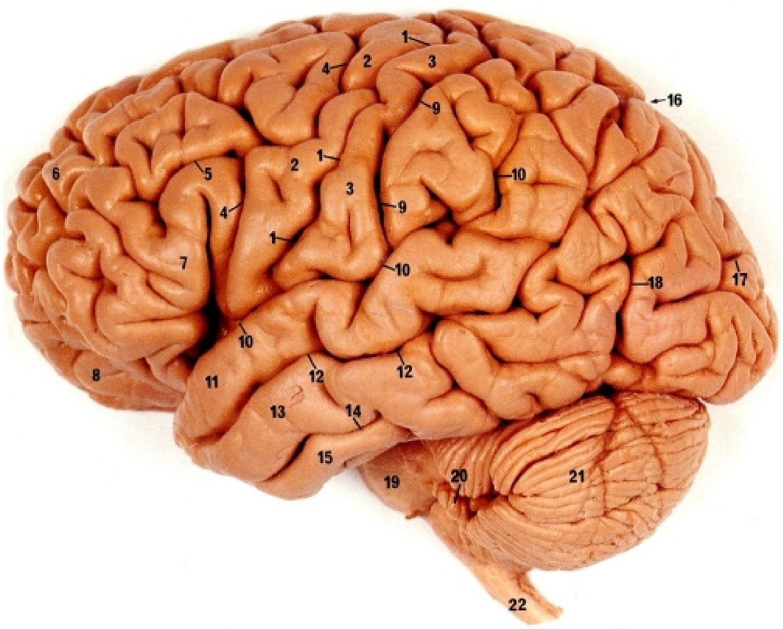
Biological computer

Mathematical computer



Mathematical logic offers a lot more guidance.
→ Turing machines.
→ Von Neumann architecture.
→ Software and hardware.
→ Today's computer science.

An engineering perspective on the brain



The brain as a computer

– Compact
– Energy efficient (20 Watts)
– Amazingly good for perception
  and informal reasoning.

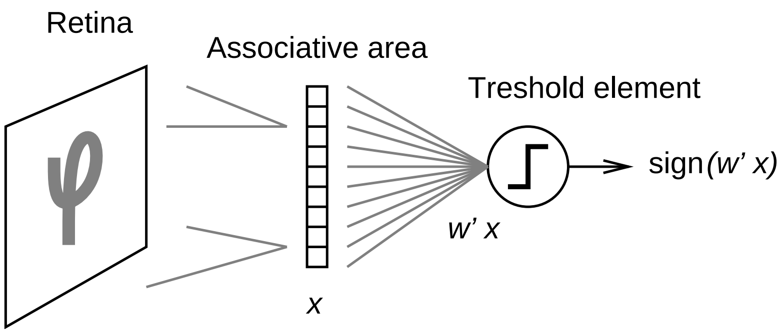
Bill of materials

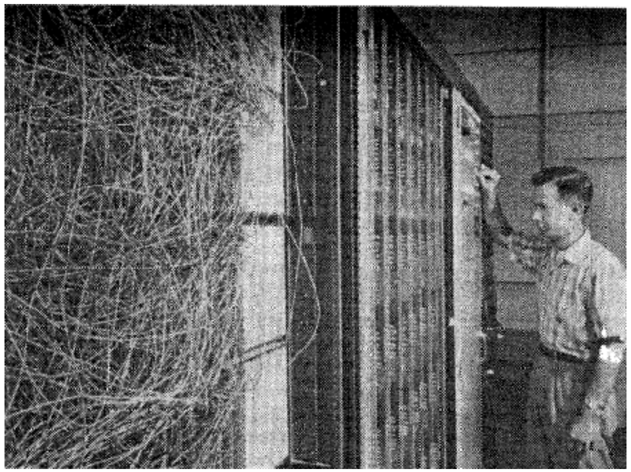
$\approx$ 90%: support, energy, cooling.
$\approx$ 10%: signalling wires.

A lot of wires in a small box

– Severe wiring constraints force a very specific architecture.
– Local connections (98%) vs. long distance connections (2%).
– Layered structure (at least in the visual system.)
– This is not a universal machine!
– But this machine defines what we belive is interesting!

Computing with artificial neurons?





McCulloch and Pitts (1943)

– Neurons as linear threshold units.

Perceptron (1957)
Adaline (1961)

– Training linear threshold units.
– A viable computing primitive?
⇐ People really tried things!

– Madaline, NeoCognitron.
– But how to train them?

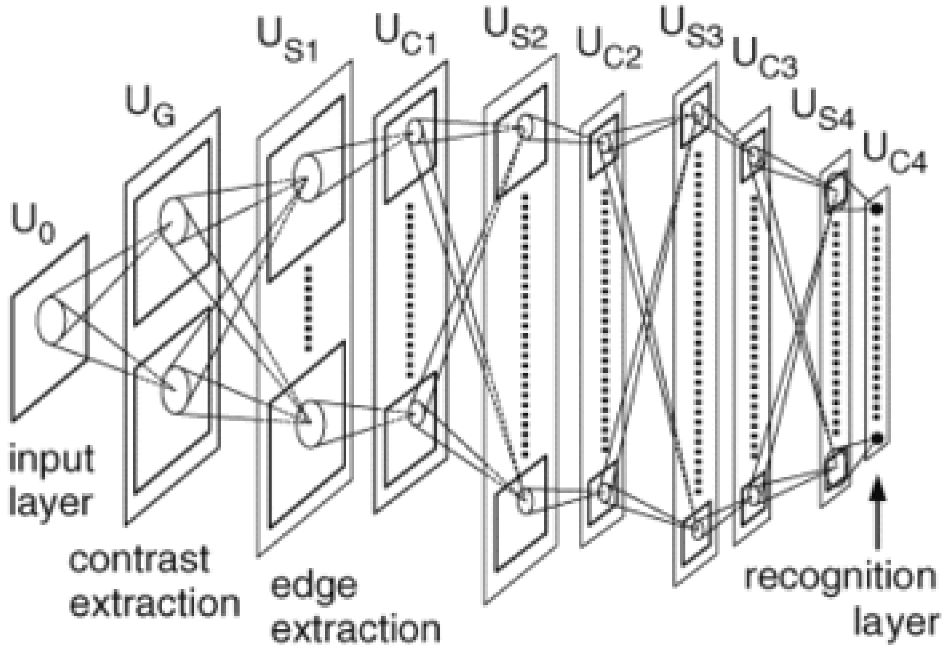
Computing with artificial neurons?

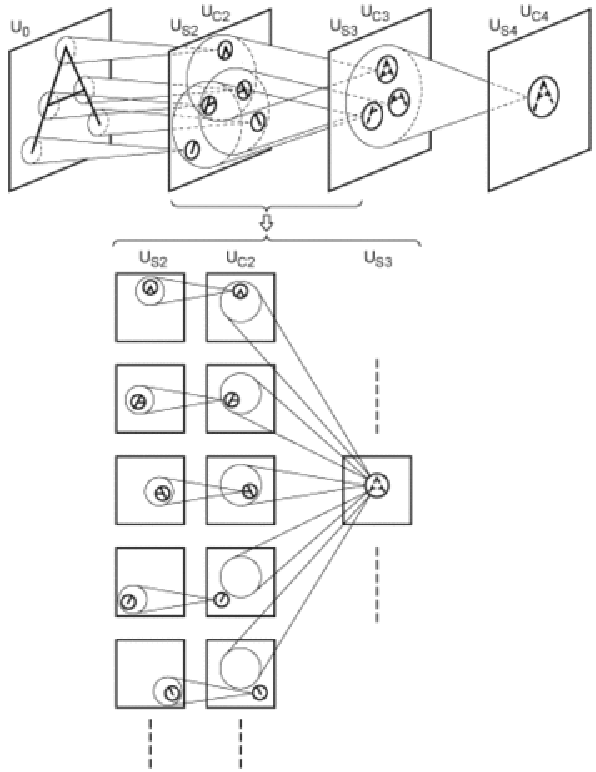
Circuits of linear threshold units?

– You can do complicated things that actually work. . .
– But how to train them?

Fukushima's NeoCognitron (1980)

– Leveraging symmetries and invariances.

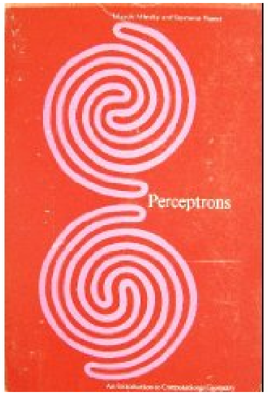




Minsky and Papert “Perceptrons” (1969)

Cicuits of logic gates

– Linear threshold unit ≈ logic gate.
– Computers ≈ lots of logic gates.
– Which functions require what kind of circuit?



Counter-examples

– Easily solvable on a general purpose computer.
– Demand deep circuits to solve effectively.
– Perceptron can train a single logic gate!
– Training deep circuits seem hopeless.

In the background

– Universal computers need a universal representation of knowledge.
– Mathematical logic is offering first order logic.
– First order logic can represent a lot more than perceptrons.
– This is absolutely correct.

Choose your Evil

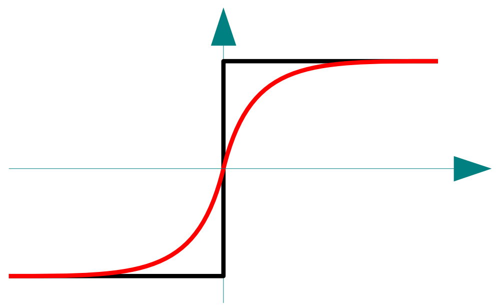
Training first order logic

Training deep circuits of logic gates

– Symbolic domains, discrete space,
– Combinatorial explosion,
– Non Polynomial

Continuous approximations

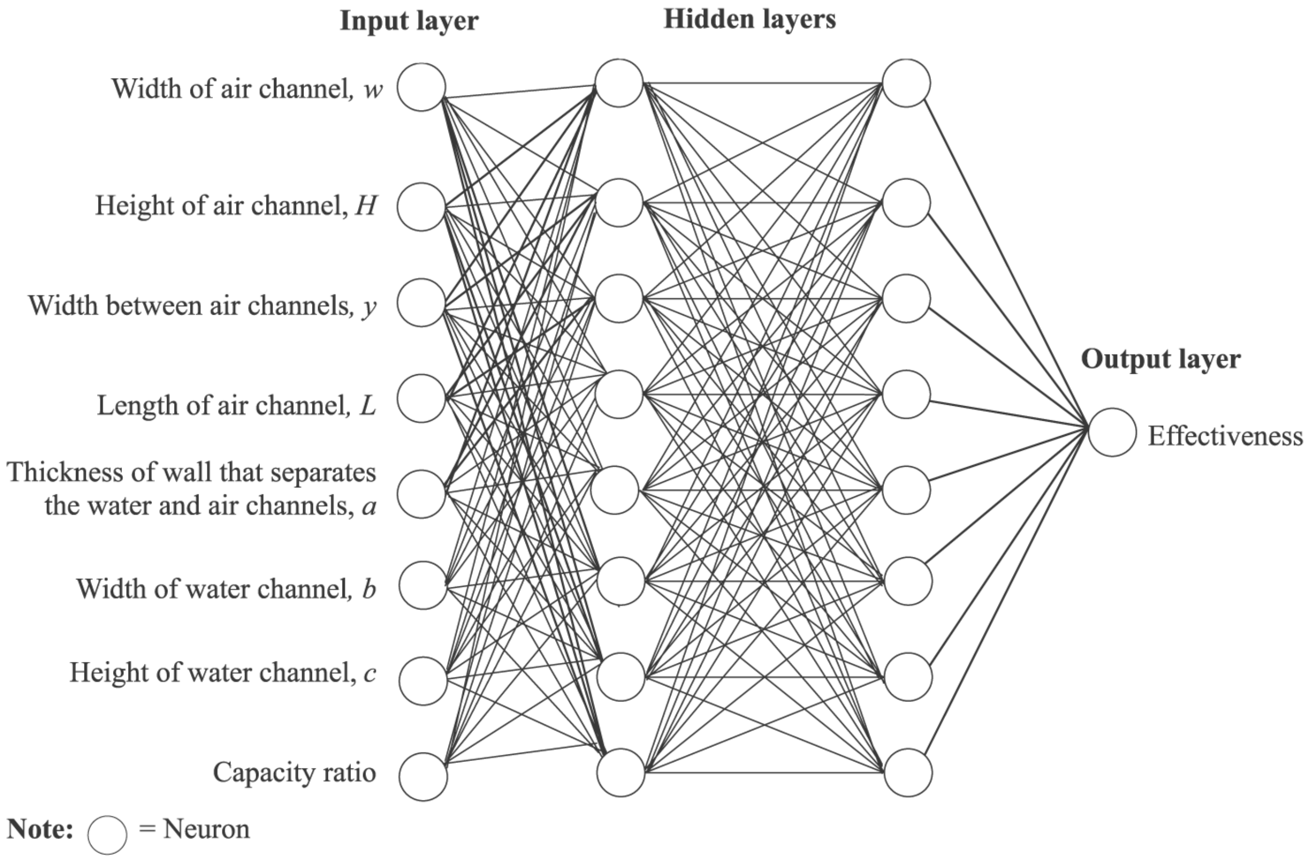
– Replace the threshold by a sigmoid function.
– Continuous and differentiable.
– Usually nonconvex.



Circuits of linear units	$\longrightarrow$	Multilayer networks (1985)
First order logic	$\longrightarrow$	Markov Logic networks (2010)
Human logic	$\longrightarrow$	?

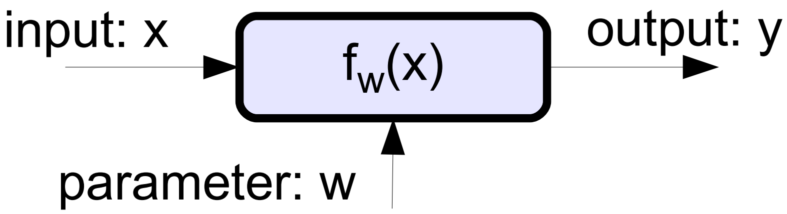
Multilayer networks, 1980s style

"ANN accurately predicts the effectiveness of the Micro-Compact Heat Exchanger and compares well with those obtained from the finite element simulation. [...] computational effort has been minimized and simulation time has been drastically reduced."



Multilayer networks, modularized

The generic brick

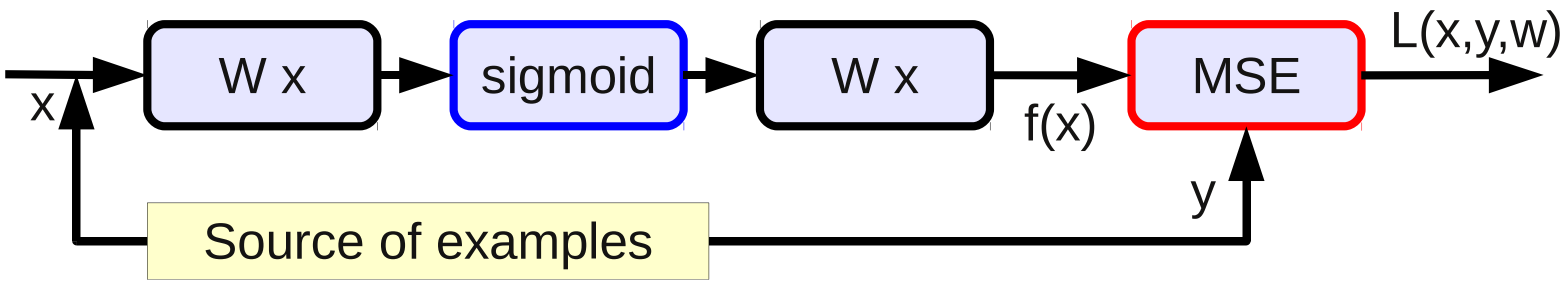


$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial y} \times \frac{\partial y}{\partial w}$$

$$\frac{\partial L}{\partial x} = \frac{\partial L}{\partial y} \times \frac{\partial y}{\partial x}$$

Forward pass in a two layer network

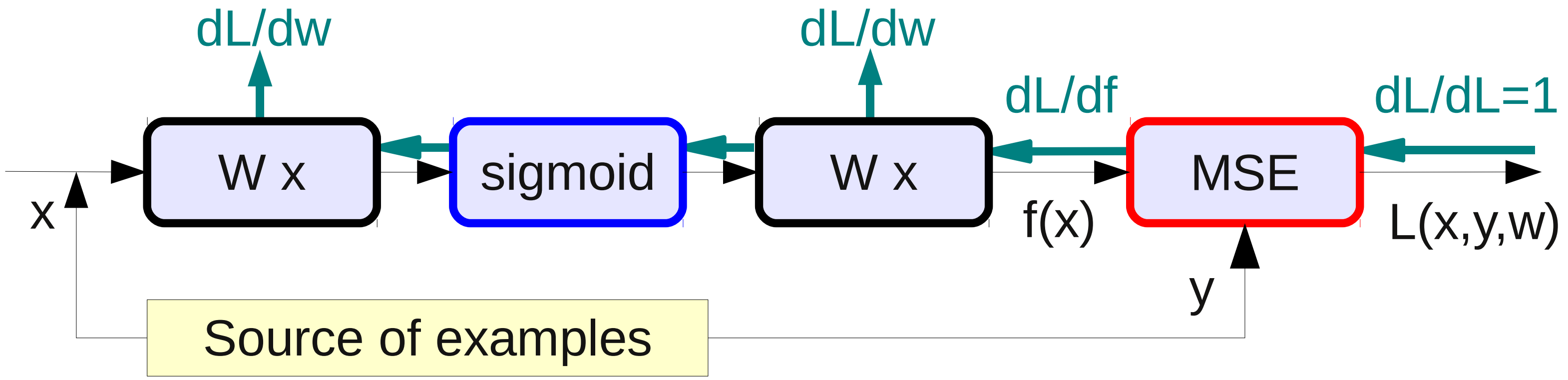
– Present example x, compute output $f(x)$, compute loss $L(x, y, w)$.



Back-propagation algorithm

Backward pass in the two layer network

– Set $dL/dL = 1$, compute gradients dL/dy and dL/dw for all boxes.



Update weights

– For instance with a stochastic gradient update.

$$\boldsymbol{w \leftarrow w - \gamma_t \frac{\partial L}{\partial w}(x, y, w)\,.}$$

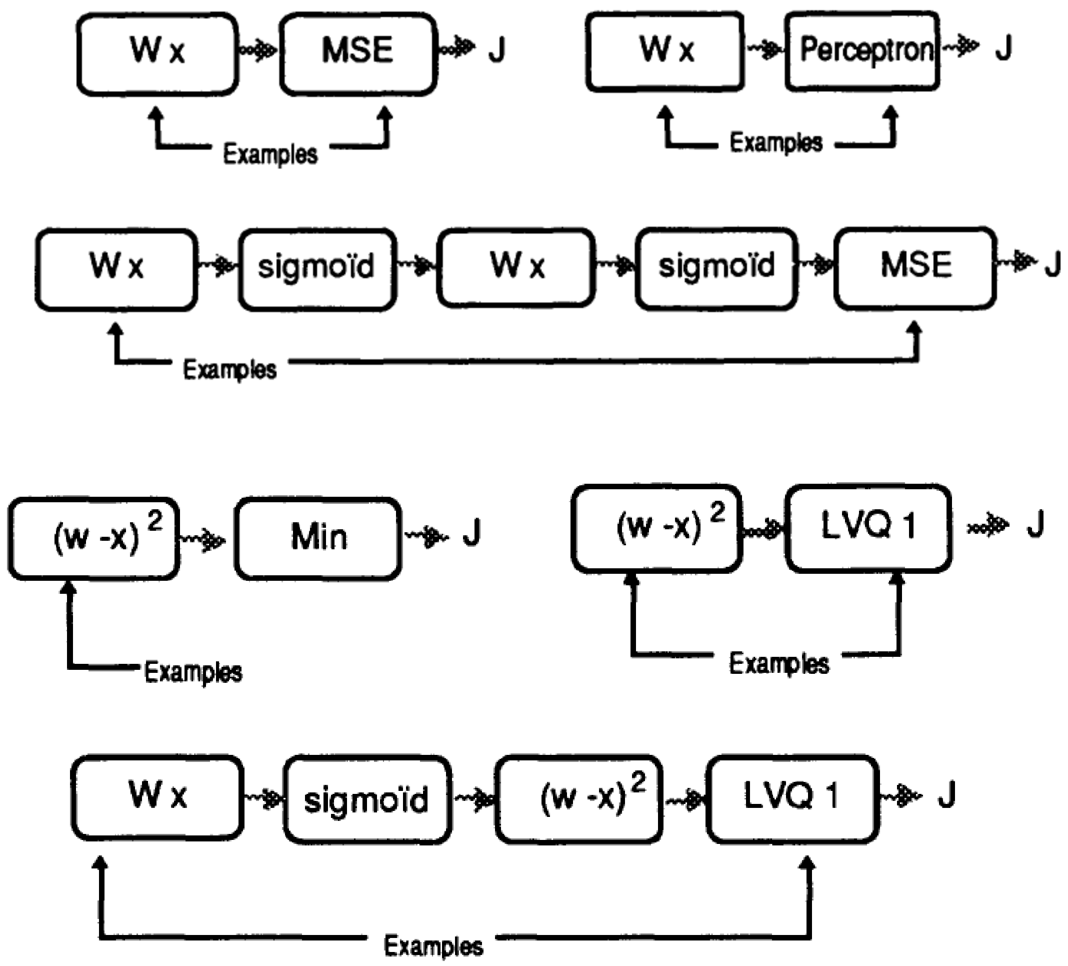
Modules

Build representations with any piece you need.

Module	**Symbol**	**Forward**	**Backward**	**Gradient**
Linear	Wx	$y = Wx$	$\check{x} = W^\top \check{y}$	$\check{w} = \check{y}\, x^\top$
Euclidian	(x-w)^2	$y_k = (x - w_k)^2$	$\check{x} = 2(x - w_k)\check{y}_k$	$\check{w}_k = 2(w_k - x)\check{y}_k$
Sigmoid	sigmoid	$y_i = \sigma(x_i)$	$\check{x}_i = \sigma'(x_i)\check{y}_i$	
MSE loss	MSE	$L = (x - y)^2$	$\check{x} = 2(x - y)\check{L}$	
Perceptron loss	Perceptron	$L = \max\{0, -yx\}$	$\check{x} = -\mathbb{1}(yx \leq 0)\check{L}$	
Log loss	LogLoss	$L = \log(1 + e^{-yx})$	$\check{x} = -(1 + e^{yx})^{-1}\check{L}$	

...

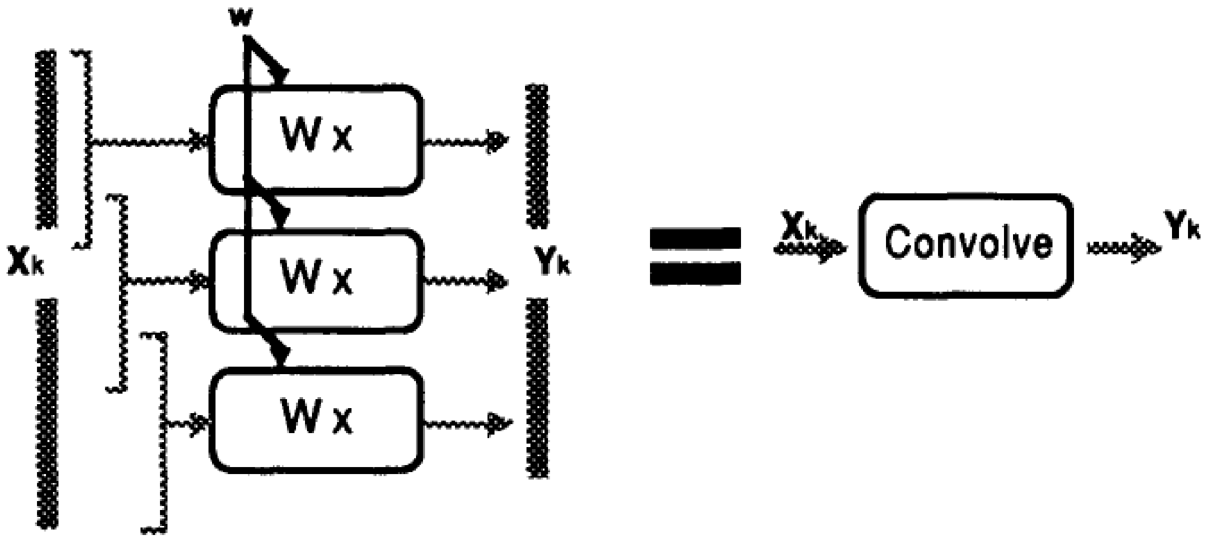
Combine modules



Composite modules

Convolutional module

– many linear modules with shared parameters.

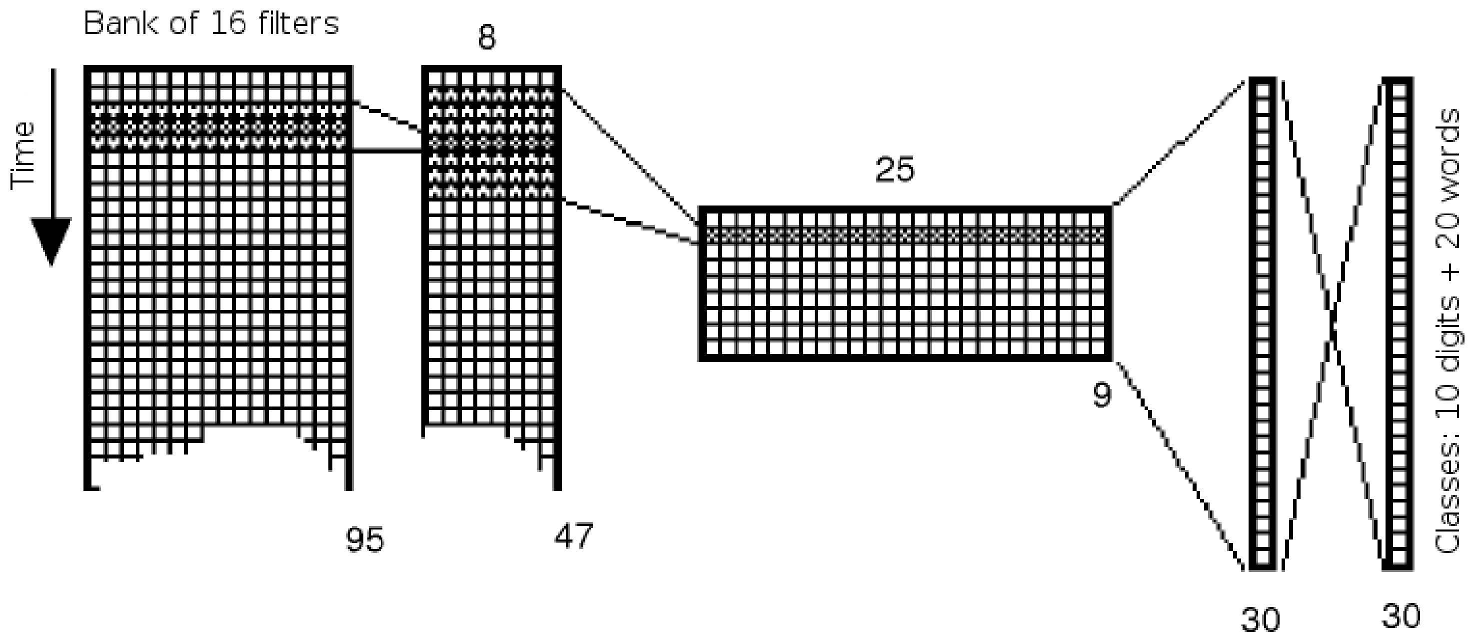


Remember the NeoCognitron?

CNNs for signal processing

Time-Delay Neural Networks

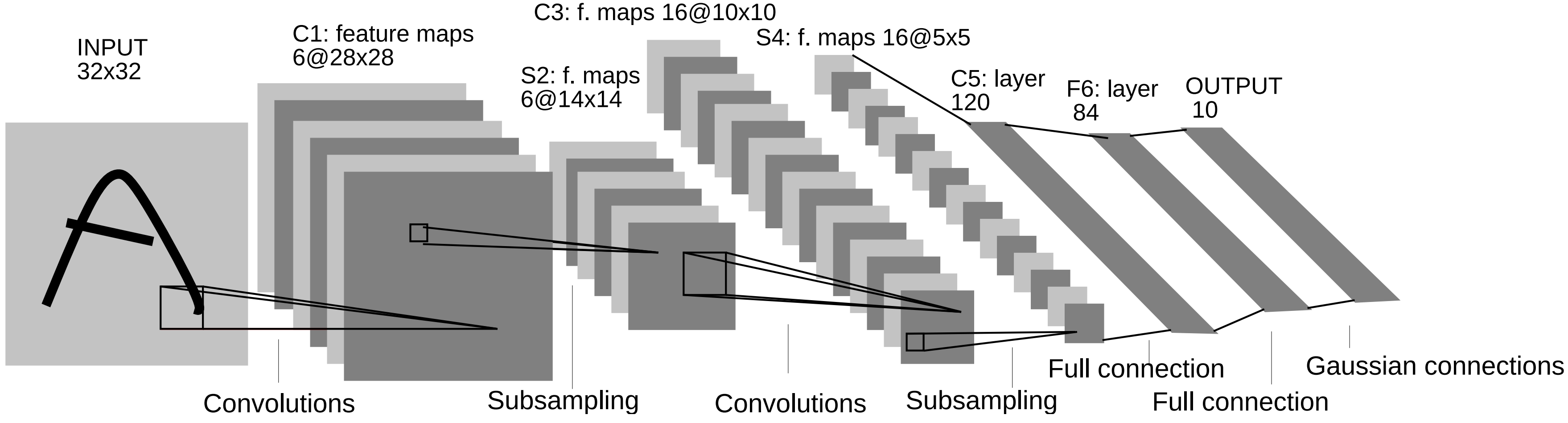
– 1990: speaker-independent phoneme recognition
– 1991: speaker-independent word recognition
– 1992: continuous speech recognition.



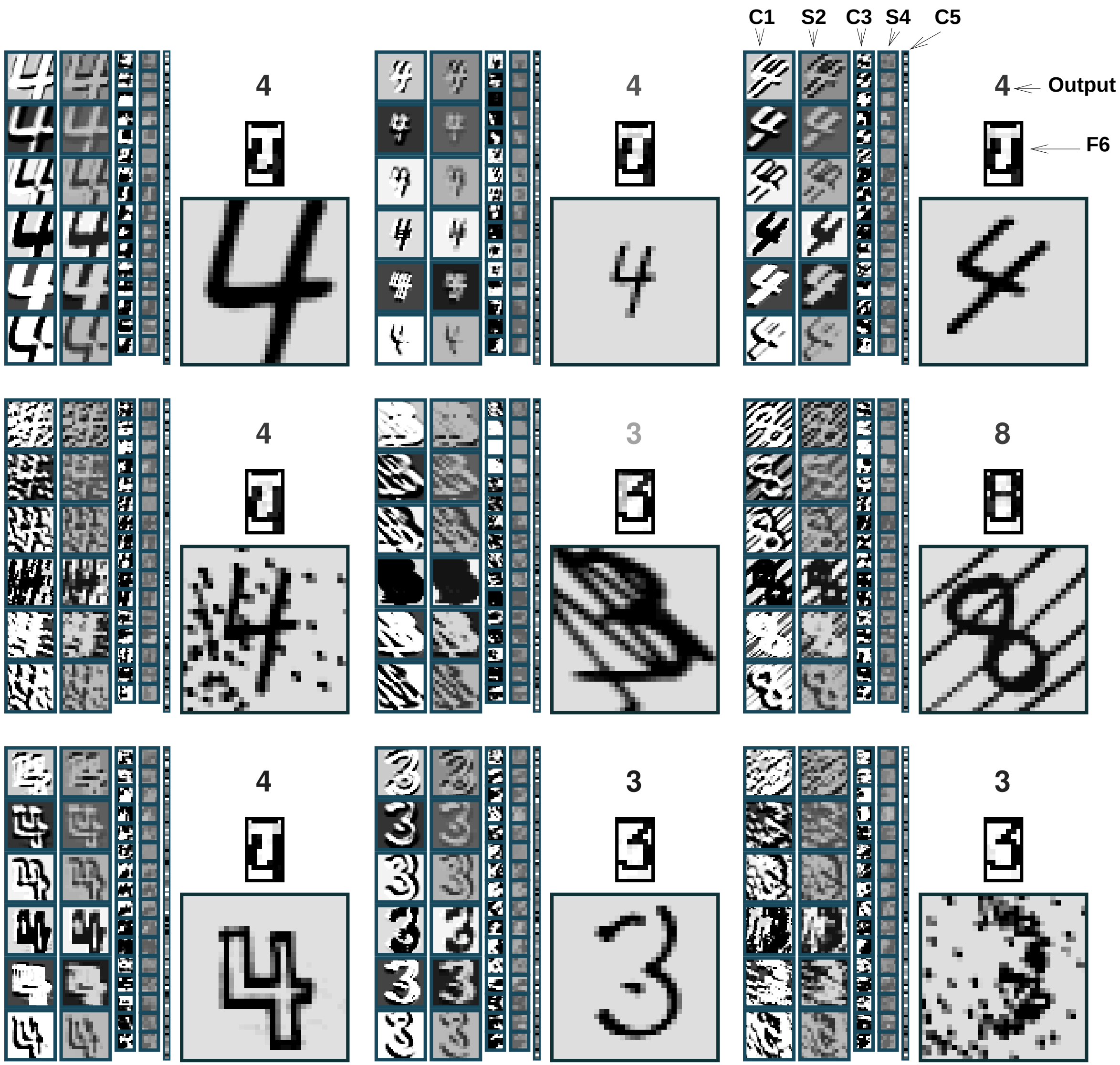
CNNs for image analysis

2D Convolutional Neural Networks

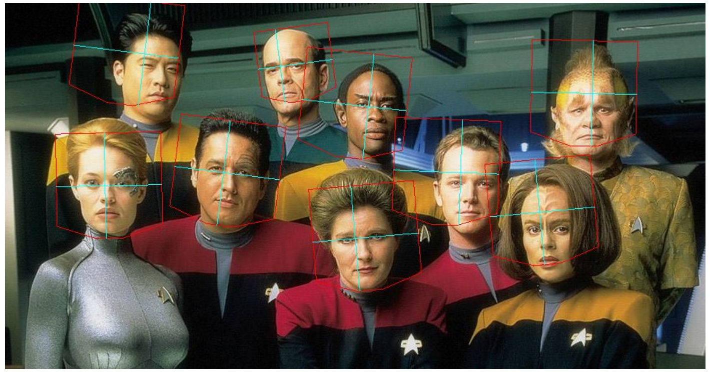
– 1989: isolated handwritten digit recognition
– 1991: face recognition, sonar image analysis
– 1993: vehicle recognition
– 1994: zip code recognition
– 1996: check reading

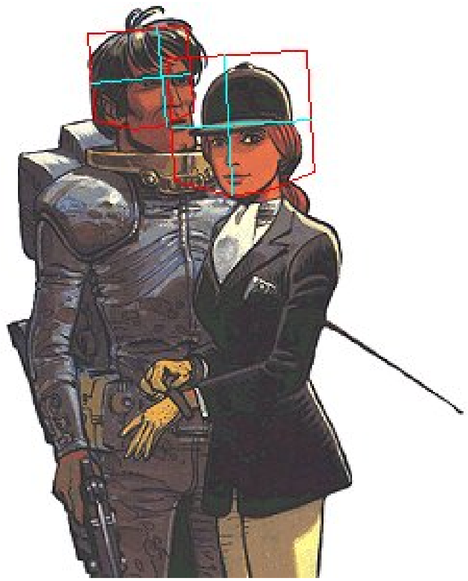


CNNs for character recognition



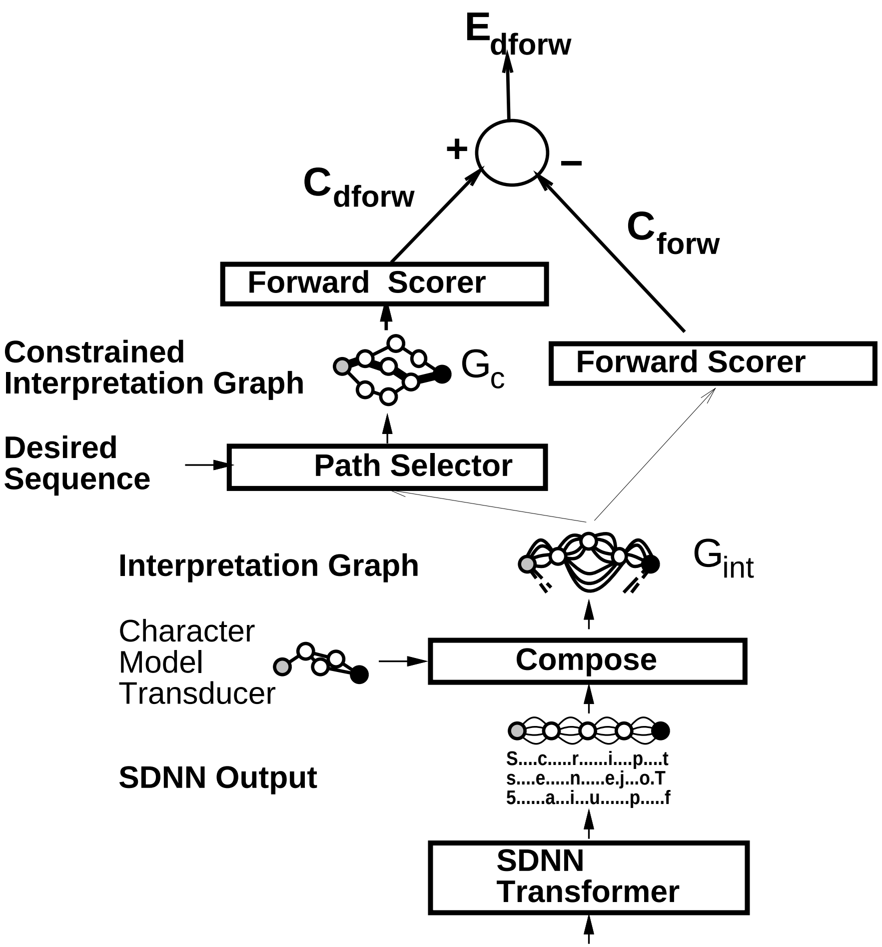
CNNs for face recognition



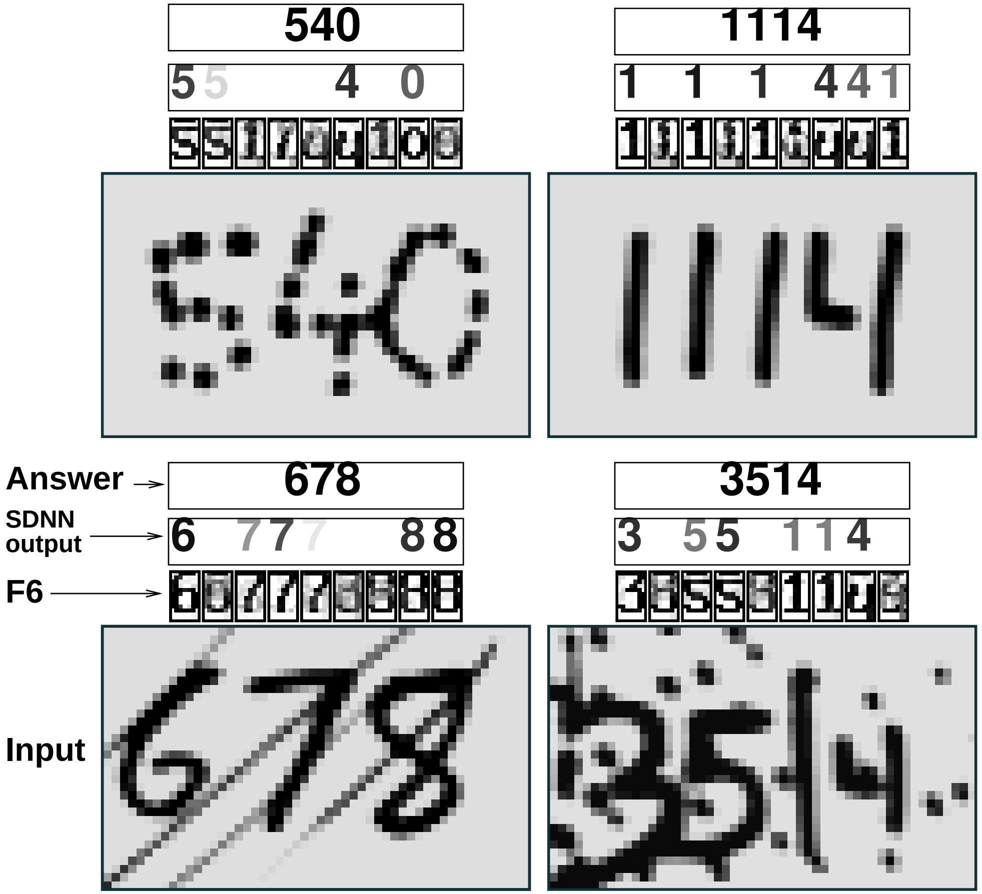


Note: same code as the digit recognizer.

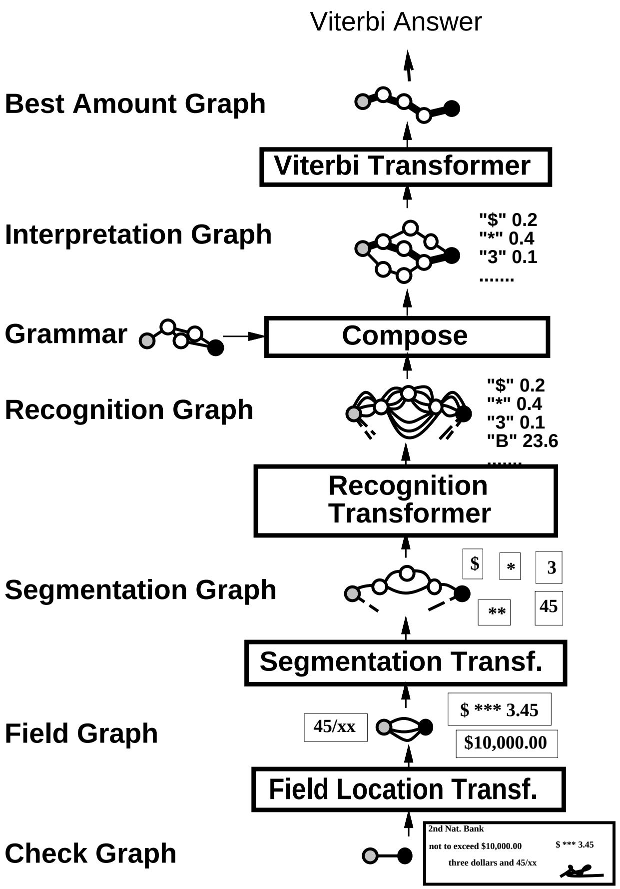
Combining CNNs and HMM



Combining CNNs and HMM



Combining CNNs and FSTs



Check reading involves

– locating the fields.

– segmenting the characters.

– recognizing the characters.

– making sense of the string.

Global training

– integrate all these modules into a single trainable system.

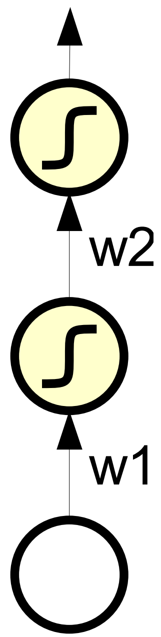
Deployment

– deployed in 1996-1997

– was still in use in 2007.

– processing ≈ 15% of the US checks.

Optimisation for multilayer network

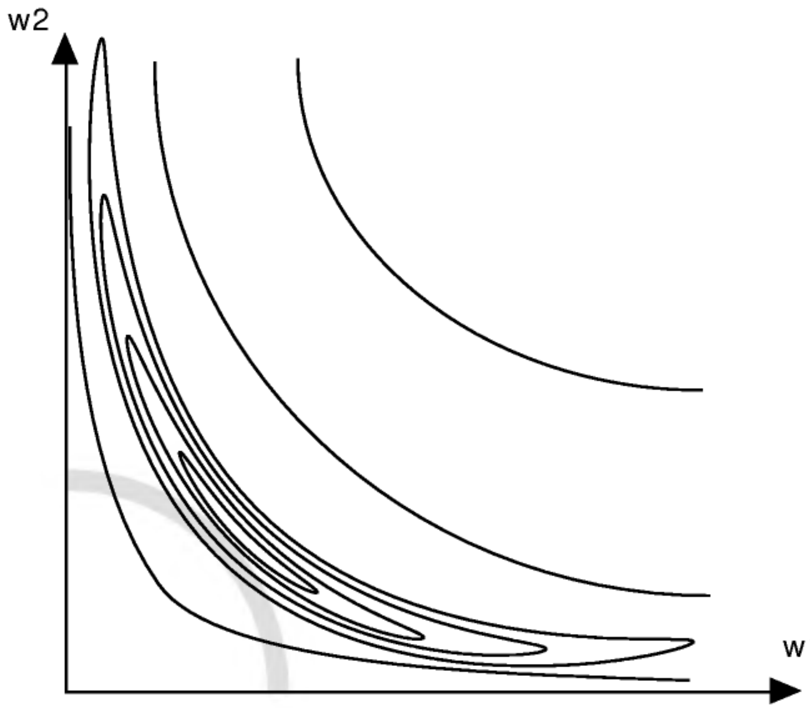


The simplest multilayer network

– Two weights w_1, w_2

– One example $\{(1,1)\}$

Optimisation for multilayer network



Landscape

– Ravine along $w_1\, w_2 = 1$.

– Massive saddle point near the origin.

– Mountains in the quadrants $w_1\, w_2 < 0$.

– Plateaux in the distance.

Tricks of the trade

– How to initialize the weights?

– How to avoid the great saddle point?

– etc.

Capacity control through optimization

Idea

– Initialize weights with quite small values (but not too small!)
  You are exercising the linear part of the sigmoid
  The whole network therefore implements a linear function.
– When learning progresses, weights increase.
  The function slowly becomes more and more nonlinear.

Early stopping

– Monitor both the training and validation errors during training.
– The training error illustrates the optimisation process.
– Stop training when the validation error stops improving.