Probabilistic Graphical Models & Probabilistic AI

Ben Lengerich

Lecture 15: Deep Learning from a GM Perspective

April 1, 2025

Reading: See course homepage

Entering Module 3: Modern Probabilistic AI

- Outstanding graded material:
  - Exam (20%, grades TBD)
  - Project midterm report (5%, 4/11)
  - Project presentation (5%, 4/31, 5/1)
    - Sign up here!
  - Project final report (15%, 5/5)
  - Extra credit (3%, sign-up)
- Module 3:
  - Papers > Textbooks

Weeks	Lecture Dates	Topic	Assignments
Module 1: Foundations of PGMs, Exact Inference			
1-4	Jan 21- Feb 13	Course Introduction, Foundations of PGMs, Exact Inference	HWs 1, 2
4	Feb 13	**Quiz**	
Module 2: Learning			
5-9	Feb 18 - Mar 18	Parameter Learning, Structure Learning, Approximate Inference	HWs 3,4,5
9	Mar 20	**Midterm Exam**	
10	Mar 21 - Mar 30	Spring Recess	
Module 3: Modern Probabilistic AI			
11-14	Apr 1 - Apr 24	Deep Learning, LLMs from a GM perspective	Project Midway Report
15	Apr 29 - May 1	Project Presentations	Project Final Report

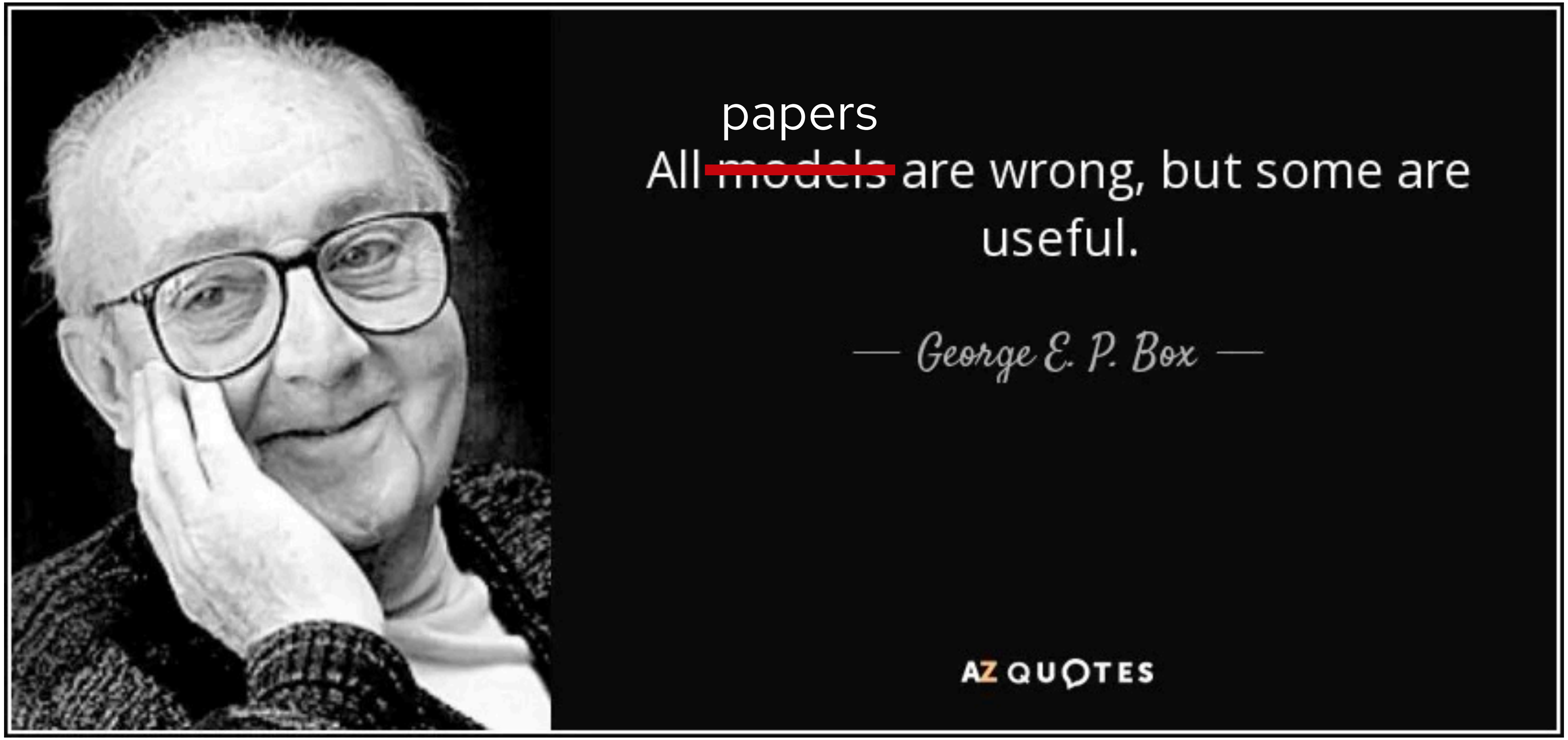
A note on research papers

How we imagine research papers:

How research papers **actually** are:

Holes big enough to drive a car through!

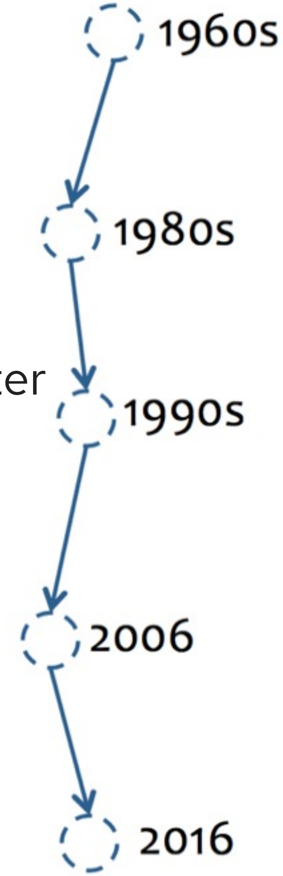
A note on research papers → let's be optimists.



Deep Learning from a GM Perspective

History – Motivation

1973 – Pres. Gerald Ford viewing computer translation



Deep learning:

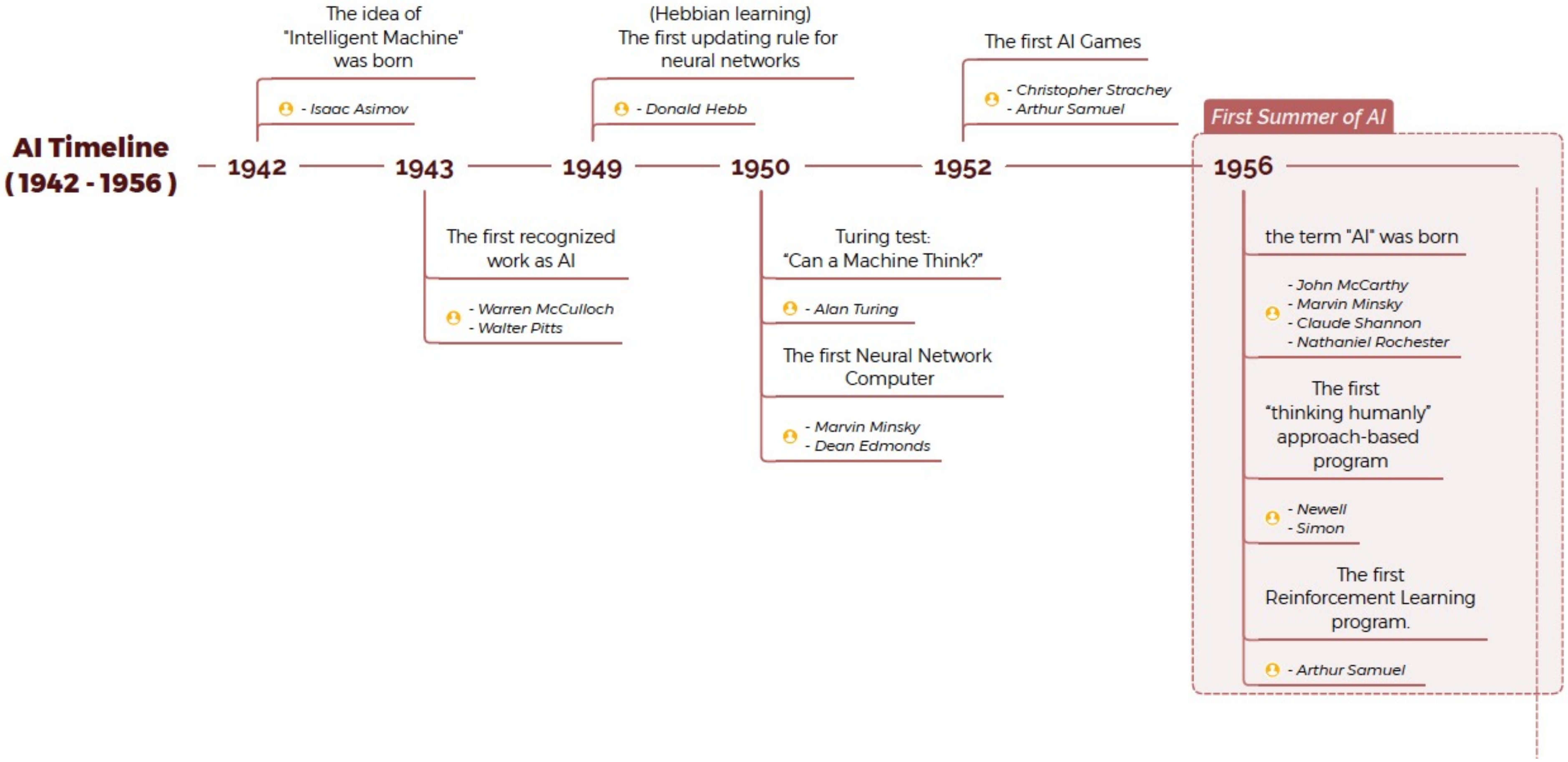
- Has won numerous pattern recognition competitions
- Does so with minimal feature engineering

This wasn't always the case!

Since 1980s: Form of models hasn't changed much, but lots of new tricks…

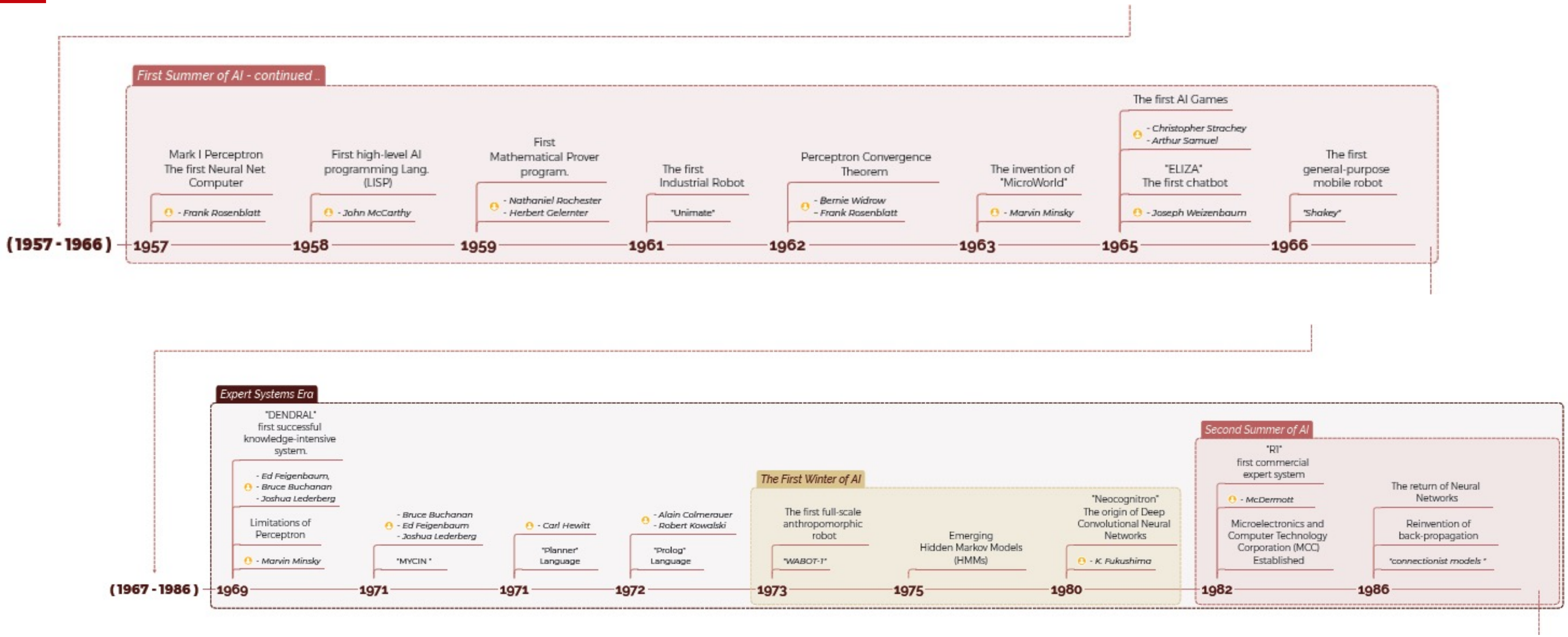
- More hidden units
- Better (online) optimization
- New nonlinear functions (ReLUs)
- Faster computers (CPUs and GPUs)

A brief history of AI



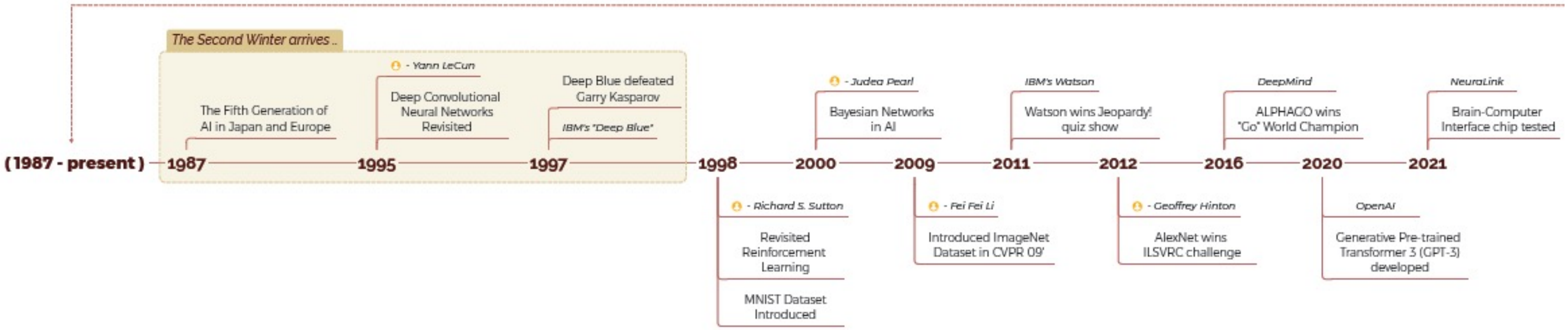
[Toosi et al 2021]

A brief history of AI



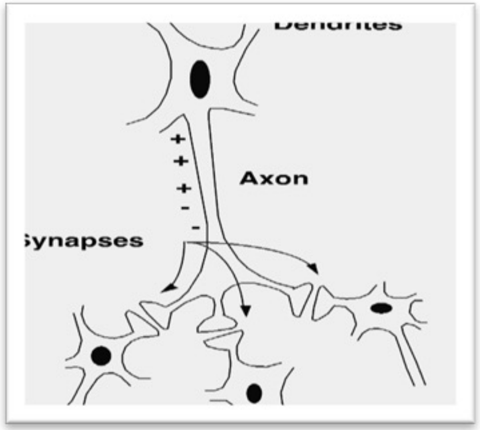
[Toosi et al 2021]

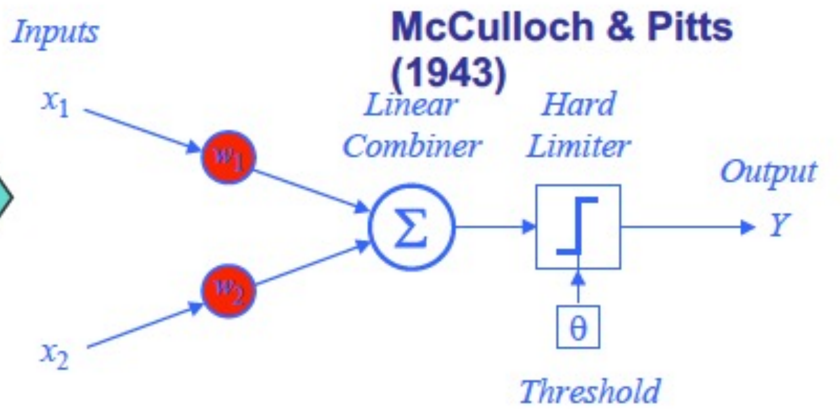
A brief history of AI



[Toosi et al 2021]

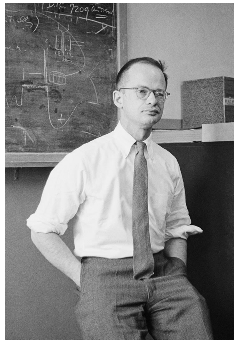
From biological neuron to artificial neuron





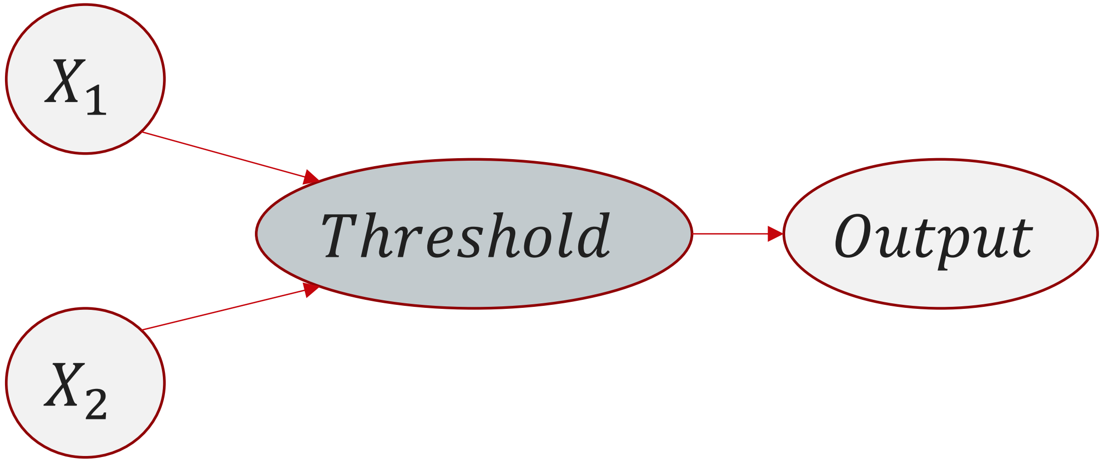
- McCulloch & Pitts neuron – **Threshold only**


Warren McCulloch

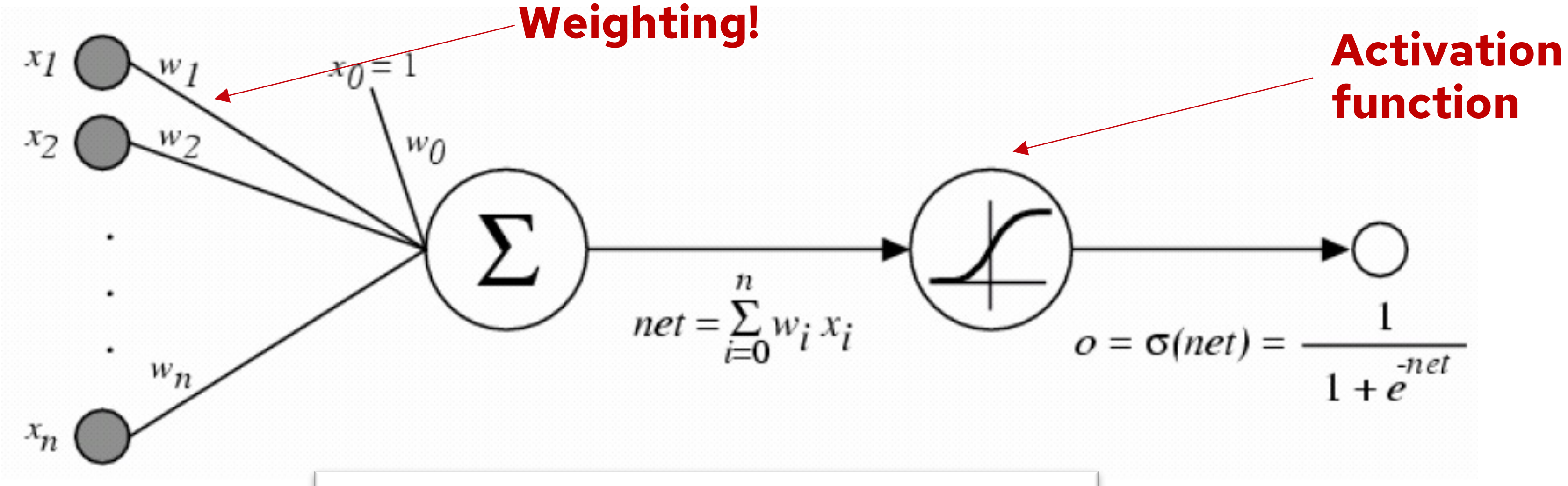

Walter Pitts

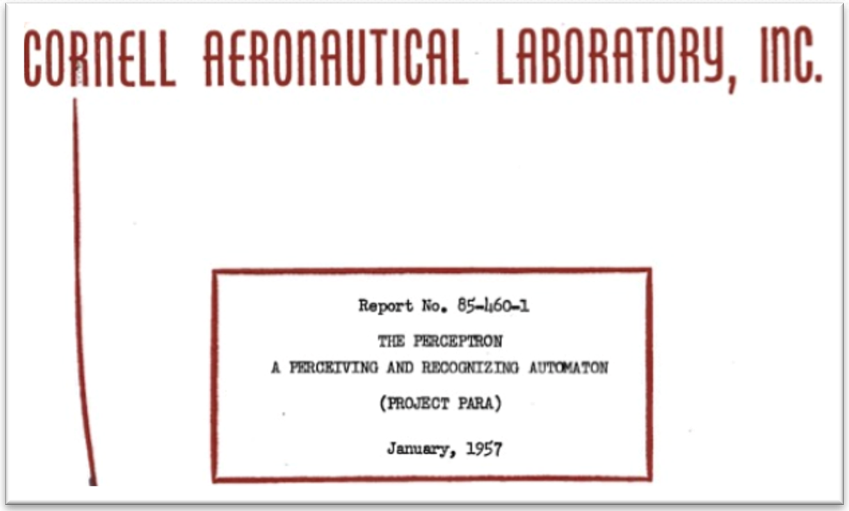
From biological neuron to artificial neuron

- McCulloch & Pitts neuron – **Threshold only**
- Can represent "AND", "OR"
- But not "NOT", "XOR"



Perceptrons generalize MP neurons





CORNELL AERONAUTICAL LABORATORY, INC.

Report No. 85-460-1

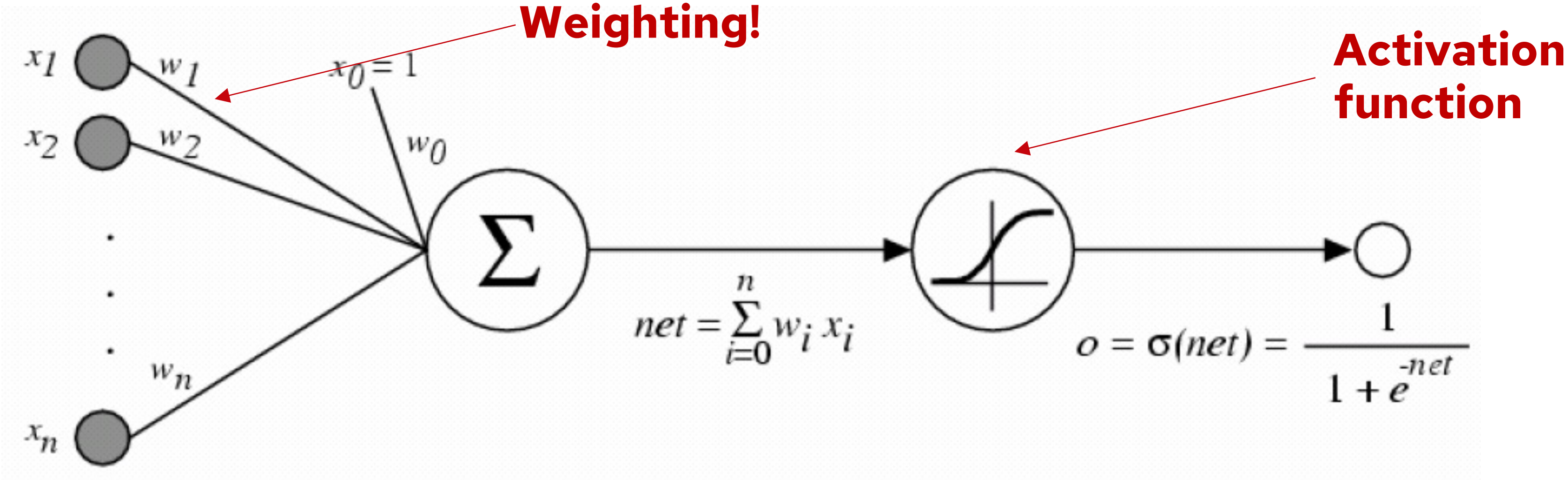
THE PERCEPTRON

A PERCEIVING AND RECOGNIZING AUTOMATON

(PROJECT PARA)

January, 1957

Perceptrons generalize MP neurons



- Consider regression problem f: X→Y for scalar Y
  - Let $Y \sim N(f(x), \Sigma^2)$
  - Then $\text{argmax}_w \log \prod_i P(y_i \mid x_i; w) = \text{argmin}_w \sum_i \frac{1}{2}\left(y_i - f(x_i; w)\right)^2$

Perceptron learning algorithm

- Recall the nice property of sigmoid: $\frac{d\sigma}{dt} = \sigma(1-\sigma)$

$$
\begin{aligned}
\frac{\partial E_D[\vec{w}])}{\partial w_j} &= \frac{\partial}{\partial w_i}\frac{1}{2}\sum_d (t_d - o_d)^2 \\
&= \frac{1}{2}\sum_d 2(t_d - o_d)\frac{\partial}{\partial w_i}(t_d - o_d) \\
&= \sum_d (t_d - o_d)\left(-\frac{\partial o_d}{\partial w_i}\right) \\
&= -\sum_d (t_d - o_d)\frac{\partial o_d}{\partial net_i}\frac{\partial net_d}{\partial w_i} \\
&= -\sum_d (t_d - o_d) o_d (1 - o_d) x_d^i
\end{aligned}
$$

x_d = input

t_d = target output

o_d = observed output

w_i = weight i

Batch mode:

Do until converge:

1. compute gradient $\nabla E_D[w]$
2. $\vec{w} = \vec{w} - \eta \nabla E_D[\vec{w}]$

Incremental mode:

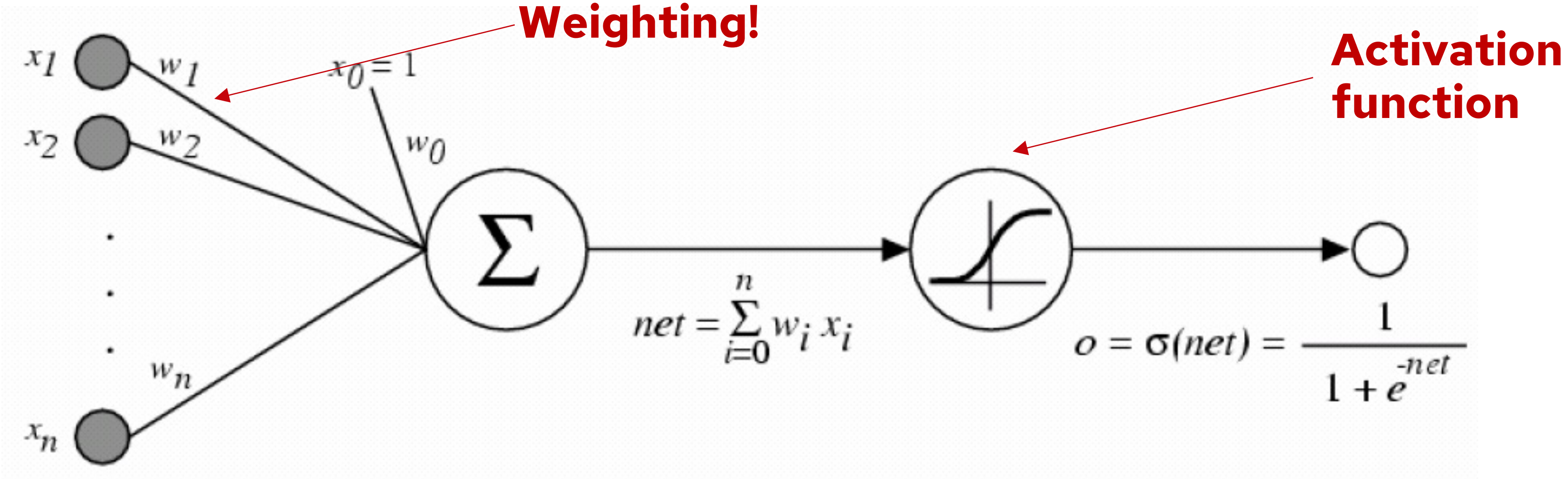
Do until converge:

- For each training example *d* in *D*
  1. compute gradient $\nabla E_d[w]$
  2. $\vec{w} = \vec{w} - \eta \nabla E_d[\vec{w}]$

  where

  $\nabla E_d[\vec{w}] = -(t_d - o_d) o_d (1 - o_d) \vec{x}_d$

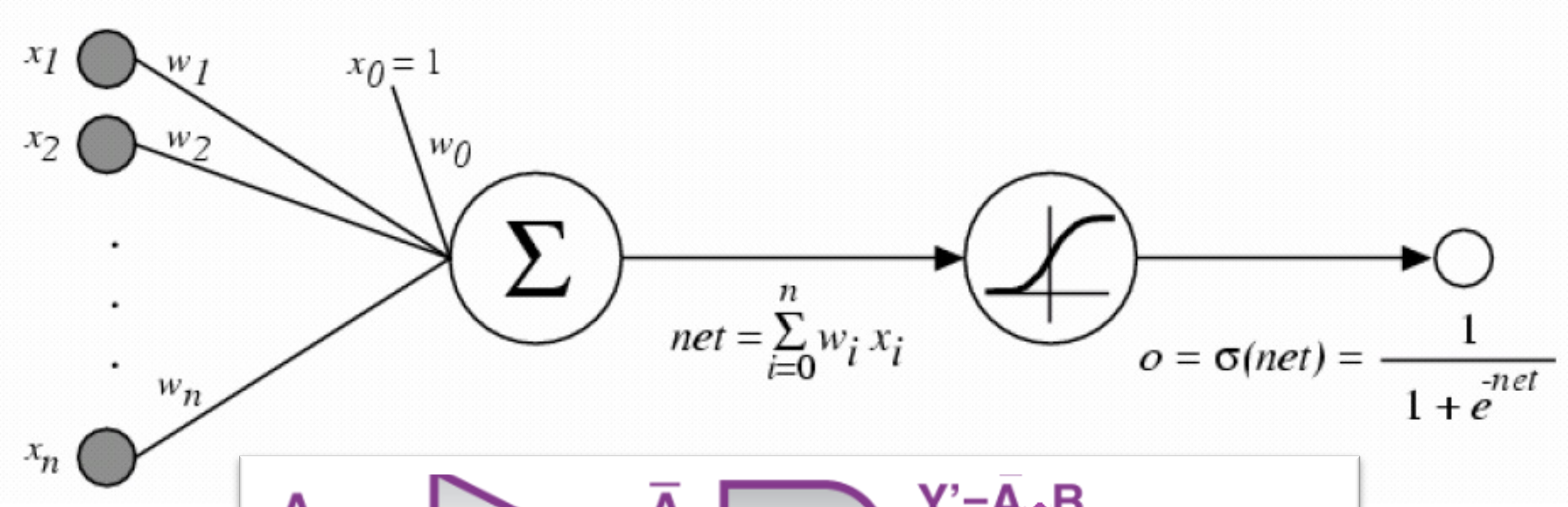
Can a Perceptron represent XOR?



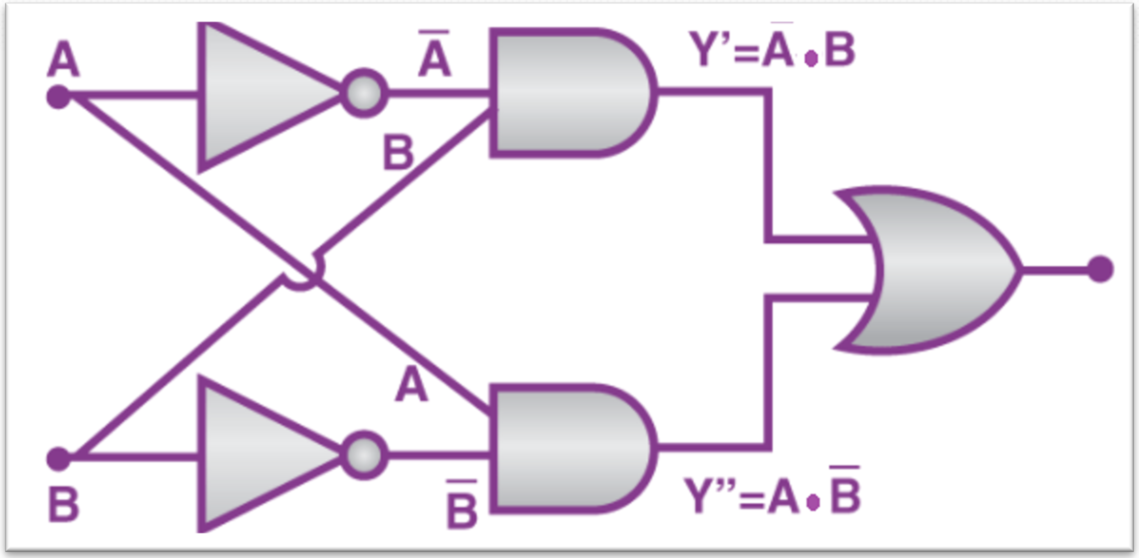
- No!
- If there were, then there would be constants w_1 and w_2 such that:
  - When $x_1 = x_2$, then $\sigma(w_1 x_1 + w_2 x_2) < \theta$
  - When $x_1 \neq x_2$, then $\sigma(w_1 x_1 + w_2 x_2) \geq \theta$
  - Let $x_1 = 1, x_2 = 0$
    - Eq. (1): $\sigma(\mathrm{w}_1) \geq \theta$
  - Let $x_1 = 0, x_2 = 1$
    - Eq. (2): $\sigma(\mathrm{w}_2) \geq \theta$
  - Let $x_1 = 1, x_2 = 1$:
    - Eq. (3): $\sigma(w_1 + w_2) < \theta$

Eq. (1) + Eq. (2) contradicts Eq. (3)

An XOR Logic Gate

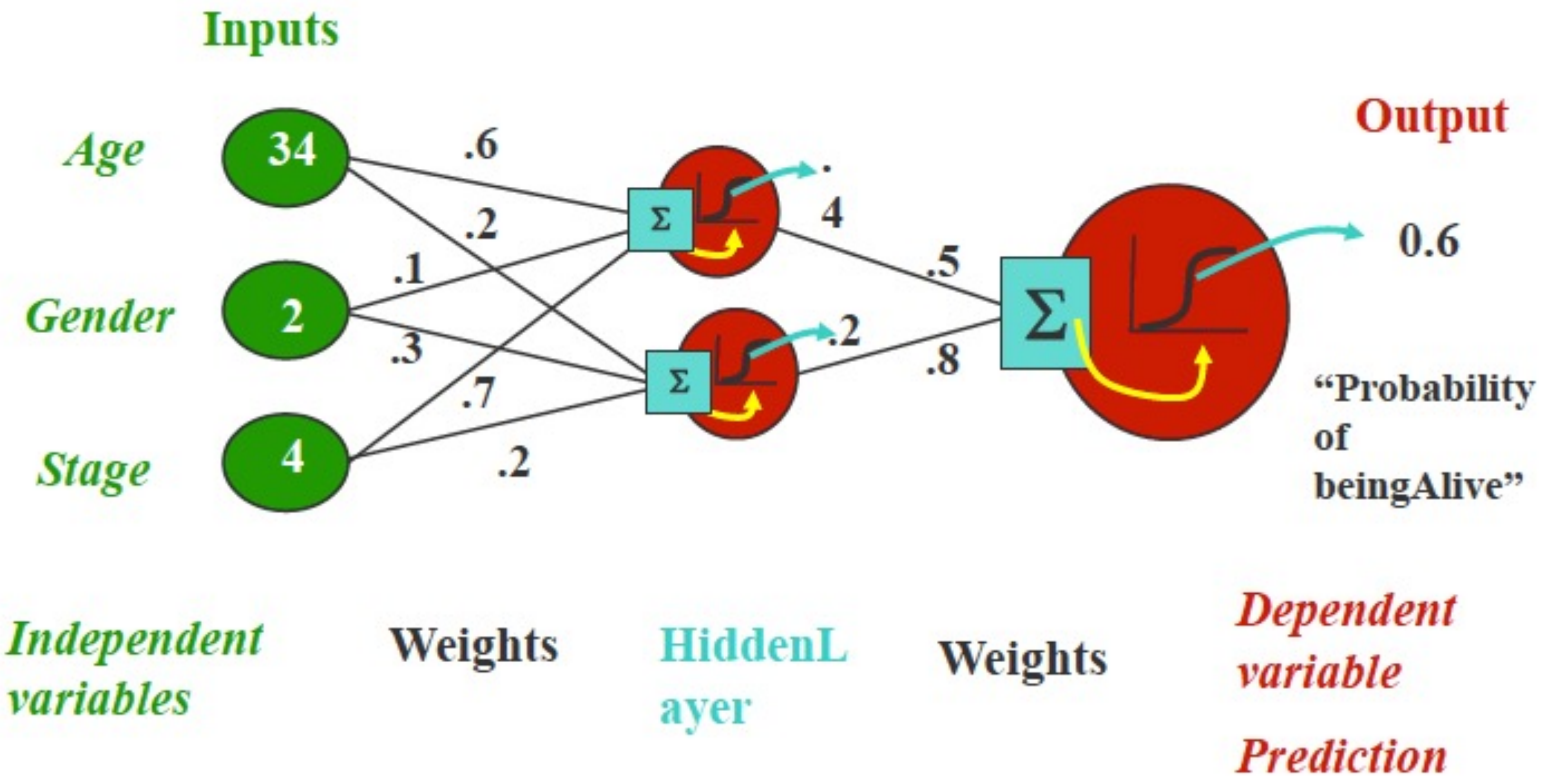


Multi-layer Perceptron?

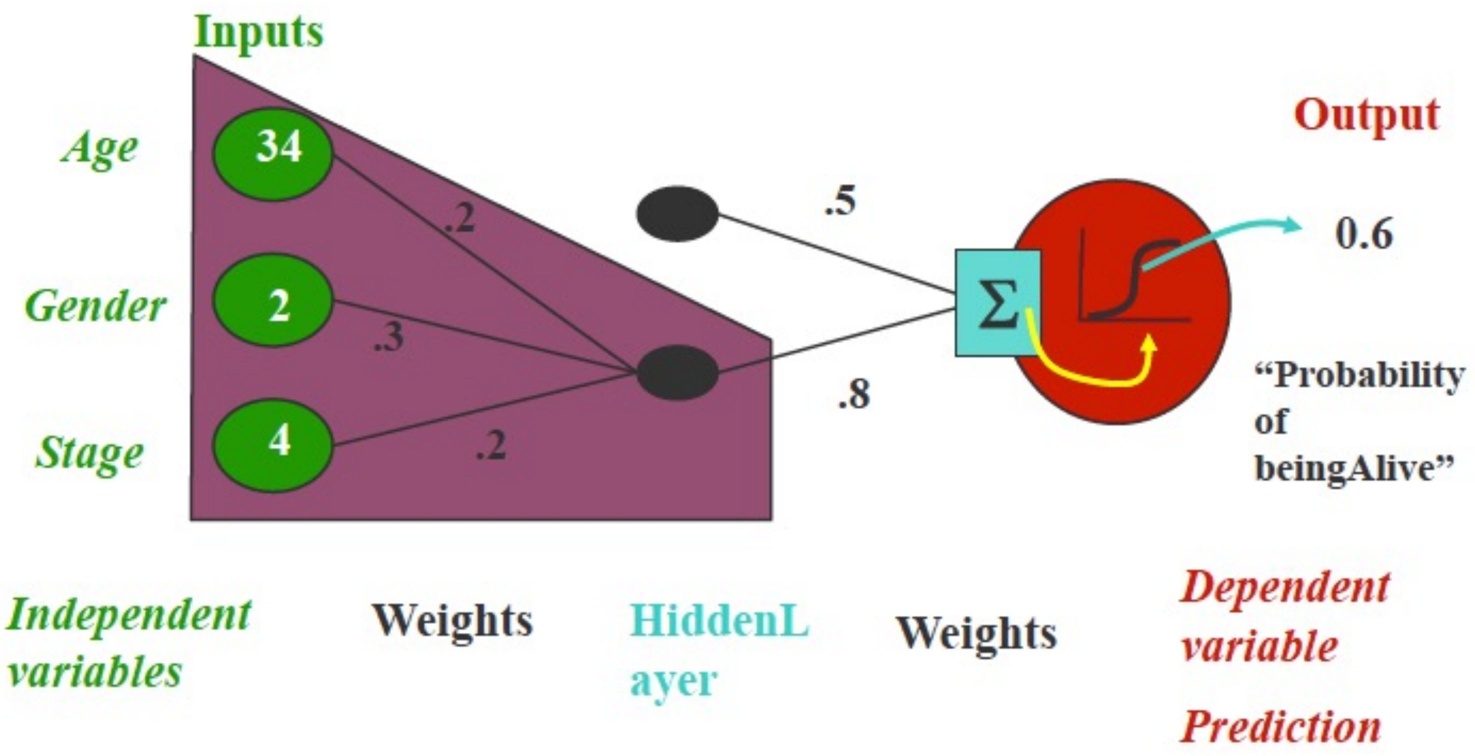


https://byjus.com/jee/basic-logic-gates/

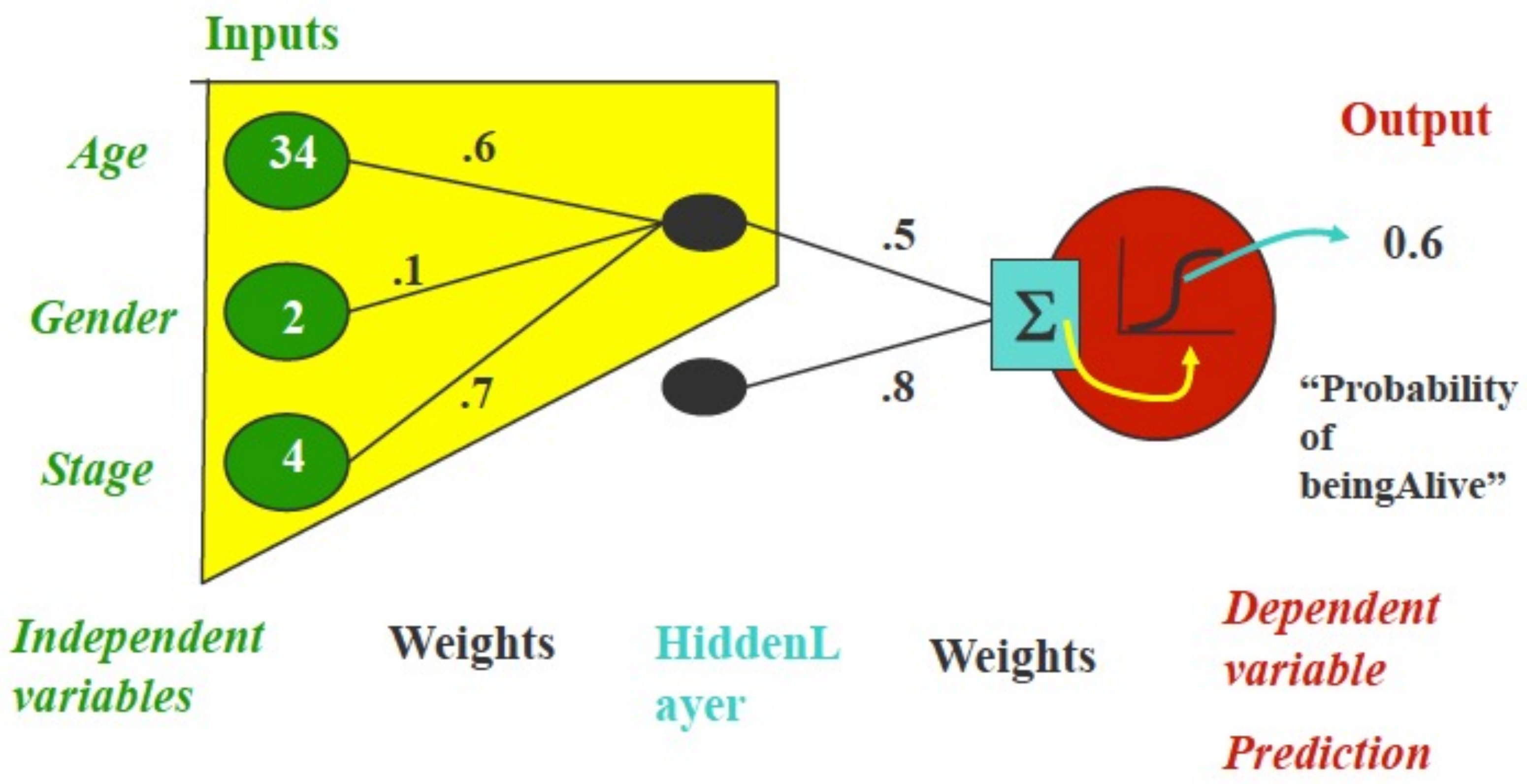
Neural Network Model (MLP)



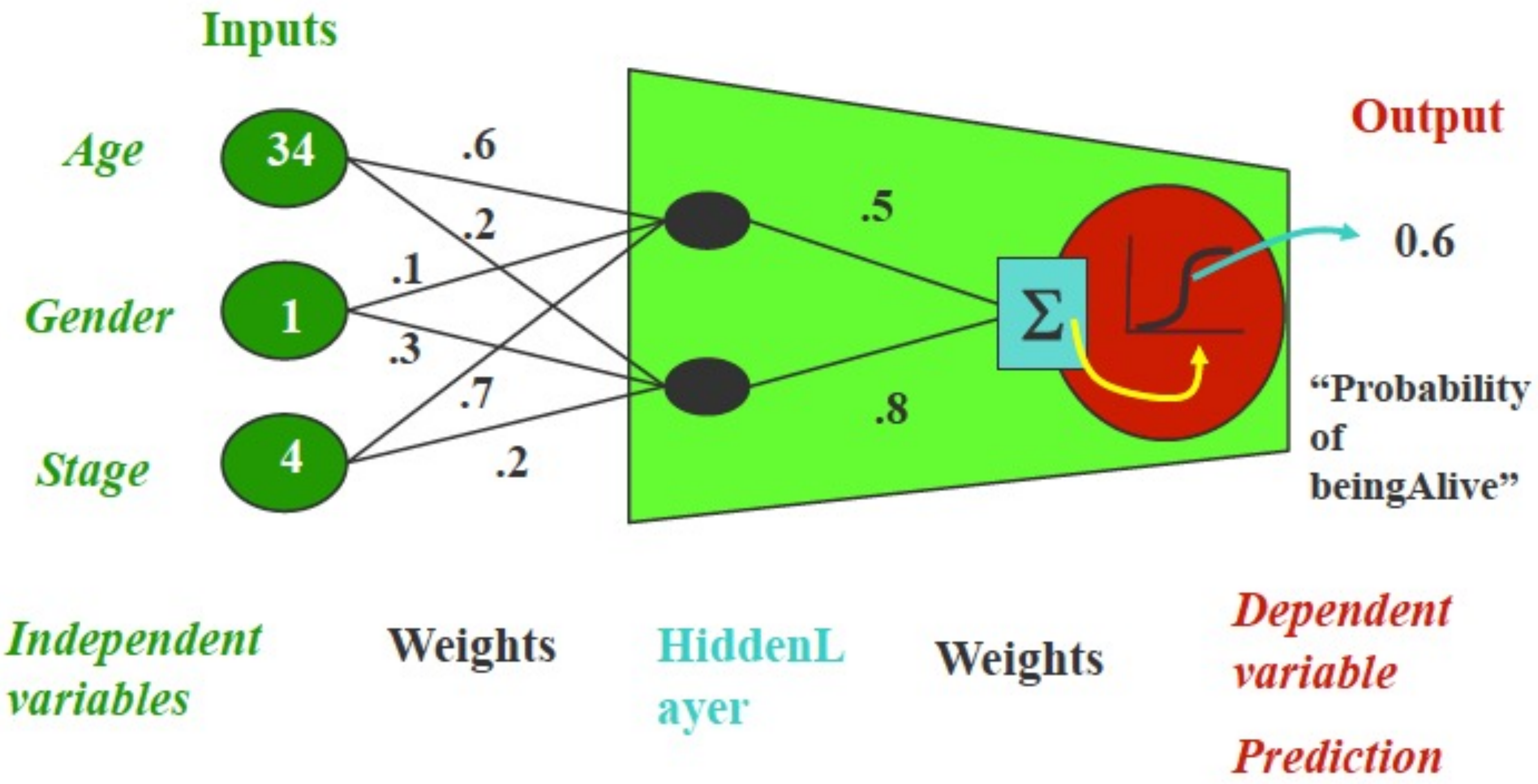
"Combined Logistic Models"…



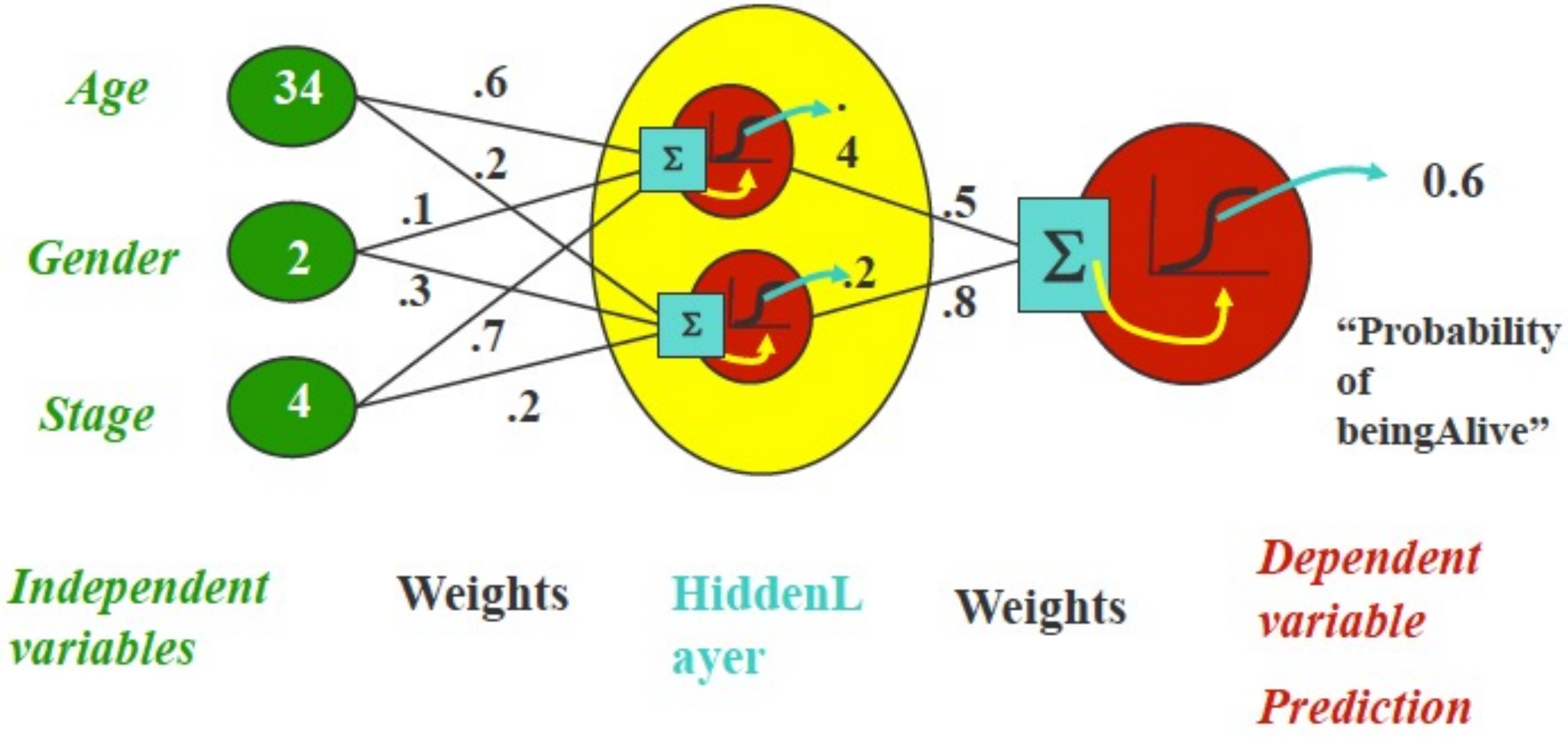
"Combined Logistic Models"…



“Combined Logistic Models”…

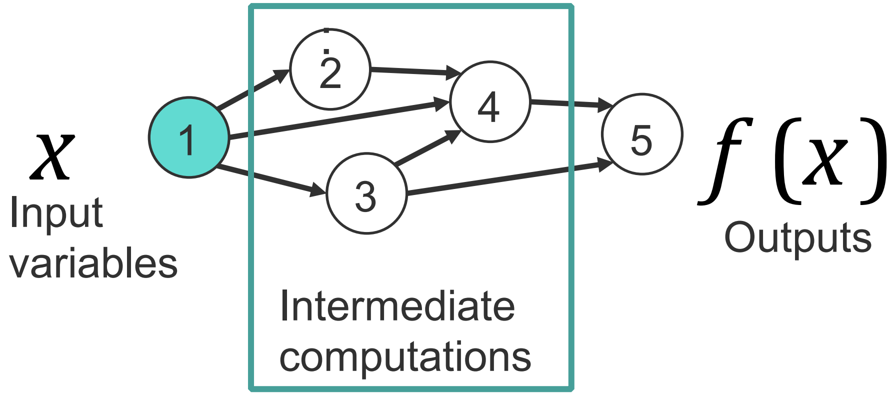


...But no target for hidden units



Backpropagation

- Neural networks are function compositions that can be represented as computation graphs:



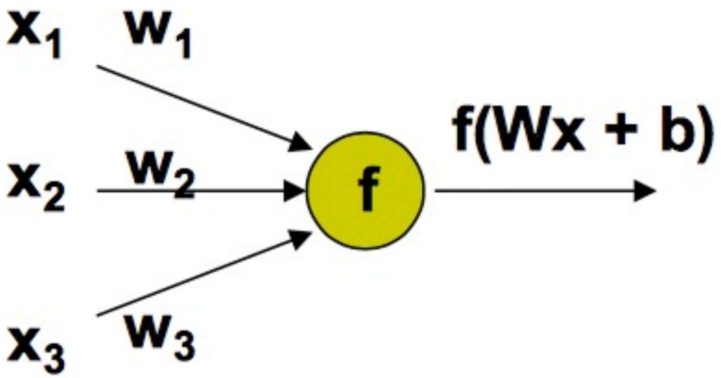
$$\frac{\partial f_n}{\partial x} =$$

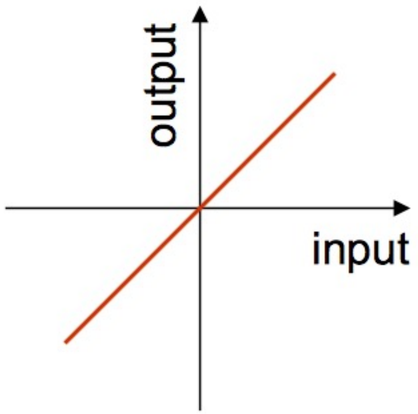
- By applying the chain rule, and working in reverse order, we get:

$$\frac{\partial f_n}{\partial x} = \sum_{i_1 \in \pi(n)} \frac{\partial f_n}{\partial f_{i_1}} \frac{\partial f_{i_1}}{\partial x} = \sum_{i_1 \in \pi(n)} \frac{\partial f_n}{\partial f_{i_1}} \sum_{i_2 \in \pi(i_1)} \frac{\partial f_{i_1}}{\partial f_{i_2}} \frac{\partial f_{i_1}}{\partial x} = \dots$$

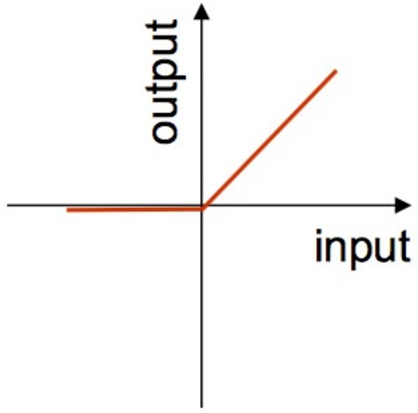
Model building blocks

- Activation functions
  - Linear and ReLU





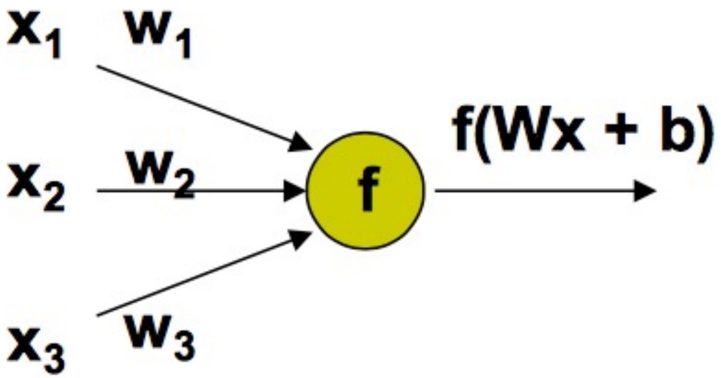
Linear

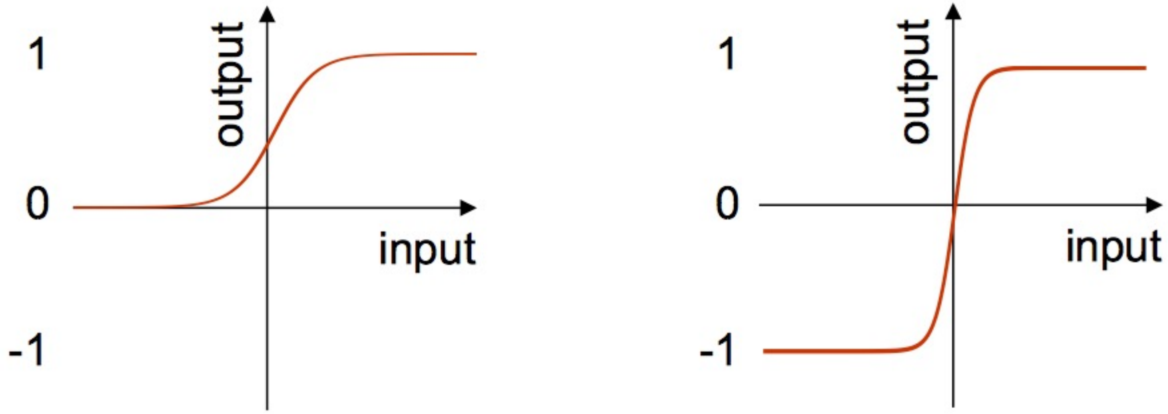


Rectified linear

Model building blocks

- Activation functions
  - Linear and ReLU
  - Sigmoid and tanh
  - Etc.

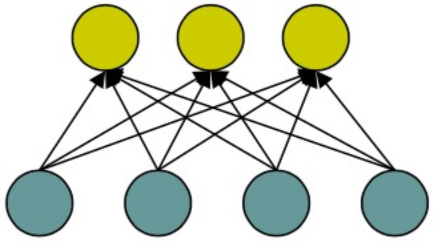


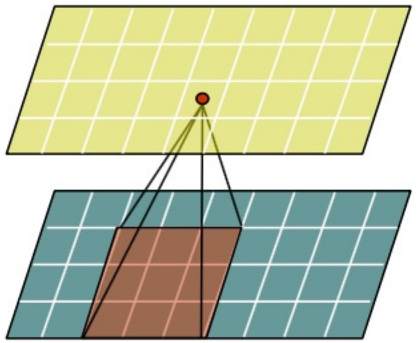


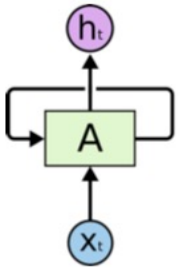
Sigmoid **Hyperbolic tangent**

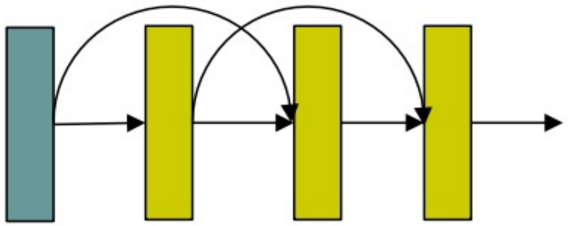
Model building blocks

- Activation functions
  - Linear and ReLU
  - Sigmoid and tanh
  - Etc.
- Layers
  - Fully connected
  - Convolutional & pooling
  - Recurrent
  - ResNets
  - Etc.







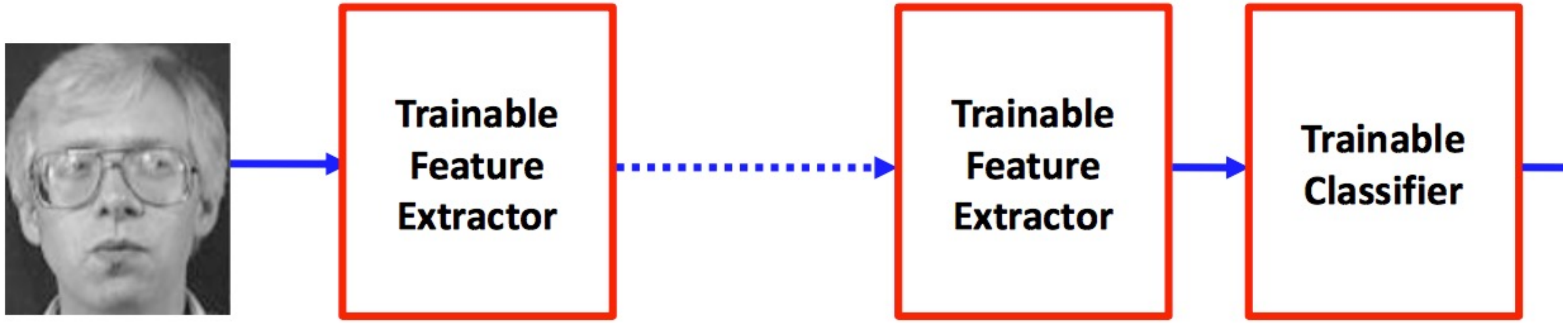


Model building blocks

- Activation functions
  - Linear and ReLU
  - Sigmoid and tanh
  - Etc.
- Layers
  - Fully connected
  - Convolutional & pooling
  - Recurrent
  - ResNets
  - Etc.
- Loss functions
  - Cross-entropy loss
  - Mean squared error
  - Etc.

- Arbitrary combinations of the basic building blocks
- Multiple loss functions – multi-target prediction, transfer learning, and more
- Given enough data, deeper architectures just keep improving
- Representation learning: the networks learn increasingly more abstract representations of the data that are “disentangled,” i.e., amenable to linear separation.

Using DNNs for hierarchical representations



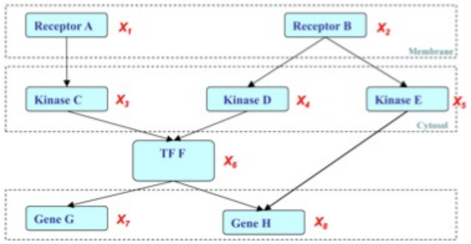
- In Language: hierarchy in syntax and semantics
  - Words → Parts of Speech → Sentences → Text
  - Objects, Actions, Attributes... → Phrases → Statements → Stories
- In Vision: part-whole hierarchy
  - Pixels → Edges → Textons → Parts → Objects → Scenes

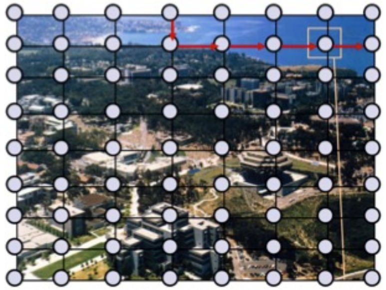
	DL	$\lesseqgtr$? ML (e.g., GM)
Empirical goal:	e.g., classification, feature learning	e.g., latent variable inference, transfer learning
Structure:	Graphical	Graphical
Objective:	Something aggregated from local functions	Something aggregated from local functions
Vocabulary:	Neuron, activation function, …	Variable, potential function, …
Algorithm:	A single, unchallenged, inference algorithm – Backpropagation (BP)	A major focus of open research, many algorithms, and more to come
Evaluation:	On a black-box score – end performance	On almost every intermediate quantity
Implementation:	Many tricks	More or less standardized
Experiments:	Massive, real data (GT unknown)	Modest, often simulated data (GT known)

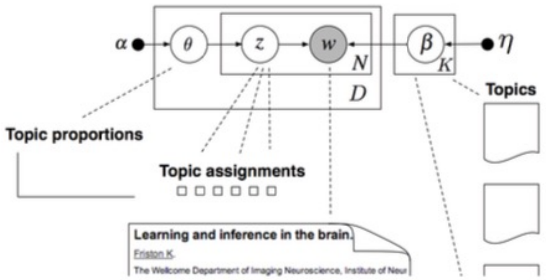
Graphical models vs Deep nets

Graphical models

- Representation for encoding meaningful knowledge and the associated uncertainty in a graphical form

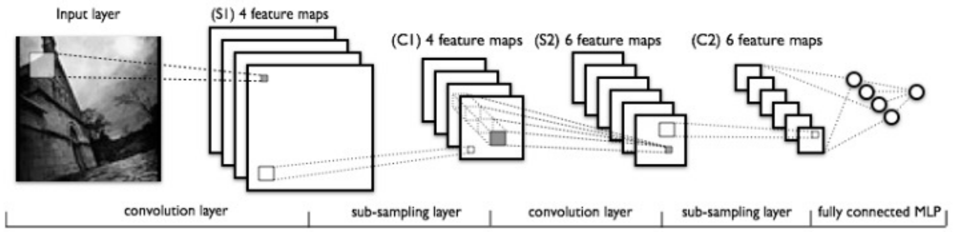


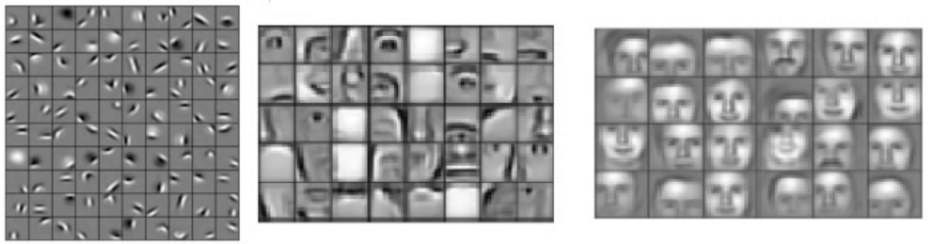




Deep neural networks

- Learn representations that facilitate computation and performance on the end-metric (intermediate representations may not be meaningful)





Graphical models vs Deep nets

Graphical models

- Representation for encoding meaningful knowledge and the associated uncertainty in a graphical form
- Learning and inference are based on a rich toolbox of well-studied (structure-dependent) techniques (e.g., EM, message passing, VI, MCMC, etc.)
- Graphs represent models

Deep neural networks

- Learn representations that facilitate computation and performance on the end-metric (intermediate representations may not be meaningful)
- Learning is predominantly based on the gradient descent method (aka backpropagation); Inference is often trivial and done via a "forward pass"
- Graphs represent computation

Graphical models vs Deep nets

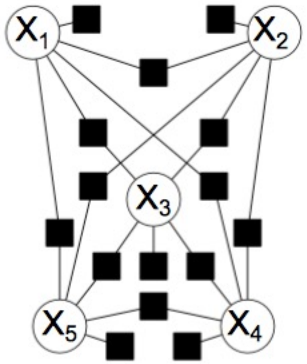
Graphical models

Utility of the graph

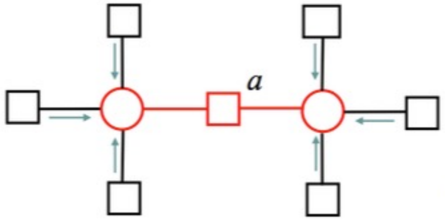
- A vehicle for synthesizing a global loss function from local structure
  - potential function, feature function, etc.
- A vehicle for designing sound and efficient inference algorithms
  - Sum-product, mean-field, etc.
- A vehicle to inspire approximation and penalization
  - Structured MF, Tree-approximation, etc.
- A vehicle for monitoring theoretical and empirical behavior and accuracy of inference

Utility of the loss function

- A major measure of quality of the learning algorithm and the model



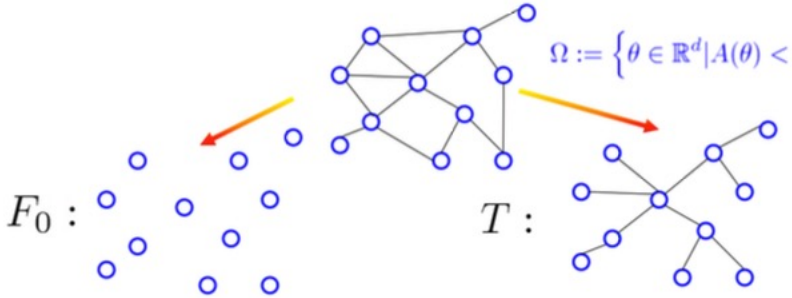
$$\log P(X) = \sum_i \log \phi(x_i) + \sum_{i,j} \log \psi(x_i, x_j)$$



$$m_{i\to a}(x_i) = \prod_{c\in N(i)\setminus a} m_{c\to i}(x_i)$$

$$b_a(X_a) \propto f_a(X_a) \prod_{i\in N(a)} m_{i\to a}(x_i)$$

$$m_{a\to i}(x_i) = \sum_{X_a\setminus x_i} f_a(X_a) \prod_{j\in N(a)\setminus i} m_{j\to a}(x_j)$$



$$\Omega := \{\theta \in \mathbb{R}^d \,|\, A(\theta) < +\infty\}$$

$$\Omega(F_0) := \{\theta \in \Omega \,|\, \theta_{(s,t)} = 0 \;\forall\, (s,t) \in E\}. \quad \Omega(T) := \{\theta \in \Omega \,|\, \theta_{(s,t)} = 0 \;\forall\, (s,t) \notin E(T)\}$$

Graphical models vs Deep nets

Graphical models

Utility of the graph

- A vehicle for synthesizing a global loss function from local structure
  - potential function, feature function, etc.
- A vehicle for designing sound and efficient inference algorithms
  - Sum-product, mean-field, etc.
- A vehicle to inspire approximation and penalization
  - Structured MF, Tree-approximation, etc.
- A vehicle for monitoring theoretical and empirical behavior and accuracy of inference

Utility of the loss function

- A major measure of quality of the learning algorithm and the model

Deep neural networks

Utility of the network

- A vehicle to conceptually synthesize complex decision hypothesis
  - stage-wise projection and aggregation
- A vehicle for organizing computational operations
  - stage-wise update of latent states
- A vehicle for designing processing steps/computing modules
  - Layer-wise parallelization
- No obvious utility in evaluating DL inference algorithms

Utility of the Loss Function

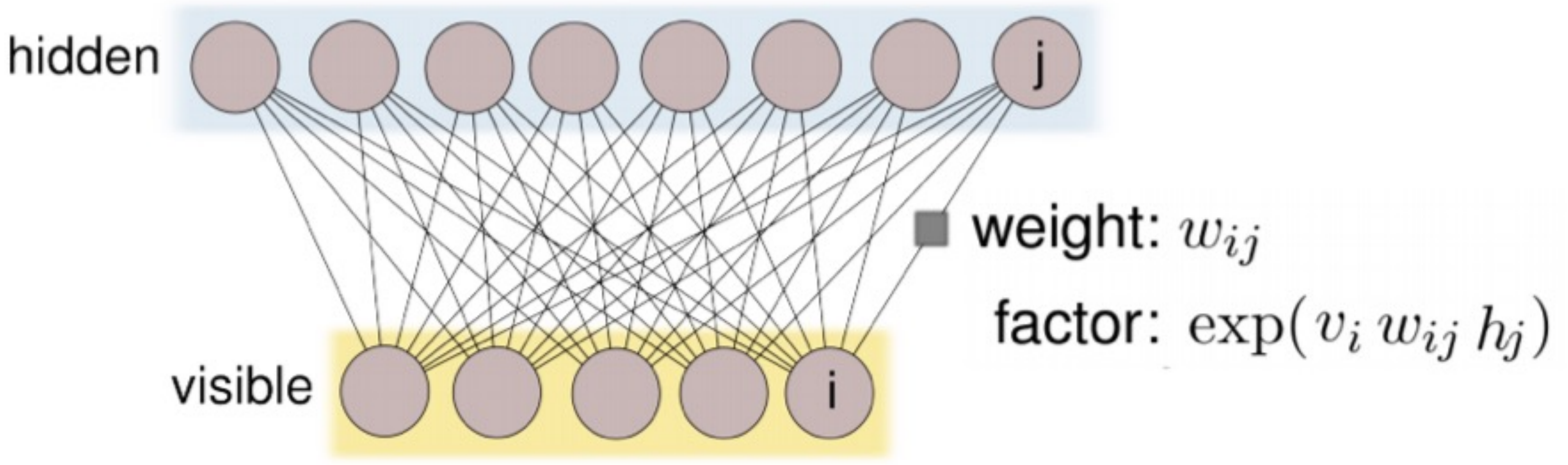
- Global loss? Well it is complex and non-convex...

Sometimes nets are proposed as true GMs:

- Boltzmann machines (Hinton & Sejnowsky, 1983)
- Restricted Boltzmann machines (Smolensky, 1986)
- Learning and Inference in sigmoid belief networks (Neal, 1992)
- Fast learning in deep belief networks (Hinton, Osindero, Teh, 2006)
- Deep Boltzmann machines (Salakhutdinov and Hinton, 2009)

Restricted Boltzmann Machines

- Assume visible units are one layer, and hidden units are another.
- Throw out all the connections within each layer.



Restricted Boltzmann Machines

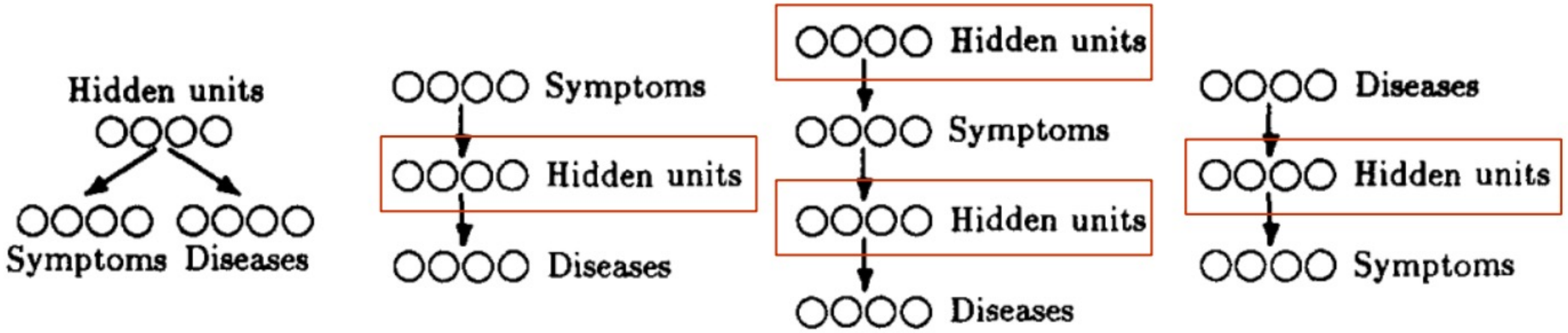
$$\frac{\partial}{\partial w} \log L \quad \propto$$

$$\underbrace{\frac{1}{N} \sum_{\mathbf{v} \in \mathcal{D}}}_{\text{data}} \underbrace{\sum_{\mathbf{h}} P(\mathbf{h} \mid \mathbf{v})}_{\text{av. over posterior}} \frac{\partial}{\partial w} \log P^{\star}(\mathbf{x}) \; - \; \underbrace{\sum_{\mathbf{v},\mathbf{h}} P(\mathbf{v}, \mathbf{h})}_{\text{av. over joint}} \frac{\partial}{\partial w} \log P^{\star}(\mathbf{x})$$

Both terms involve averaging over $\frac{\partial}{\partial w} \log P^{\star}(\mathbf{x})$.

Contrastive Divergence estimates the second term with a Monte Carlo estimate from 1-step of a Gibbs sampler!

Sigmoid Belief Networks



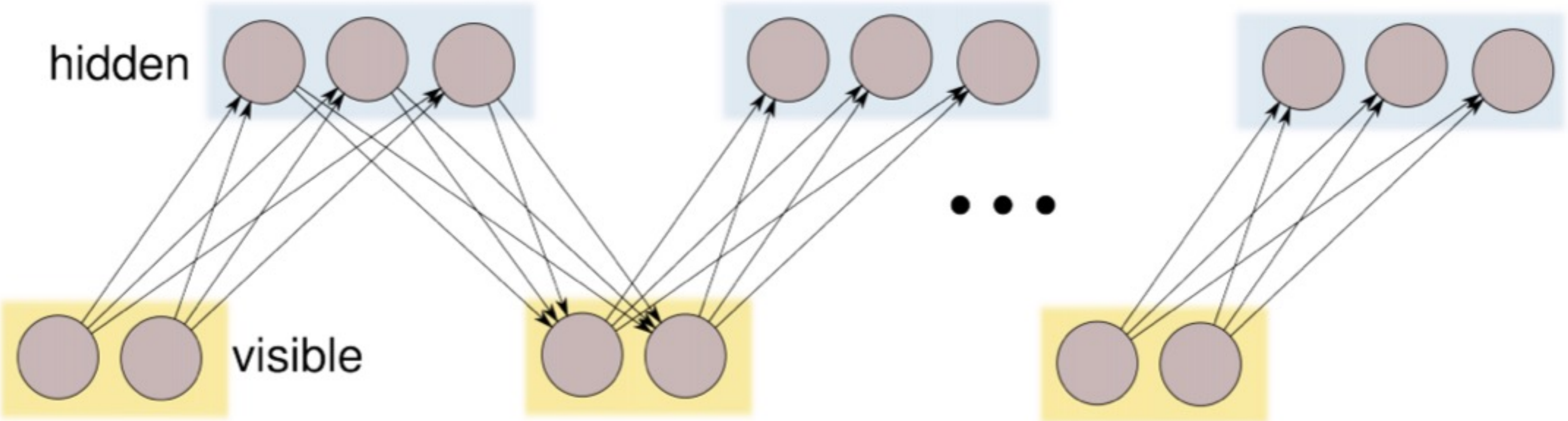
Sigmoid belief nets are simply Bayes networks conditionals represented in a particular form:

$$P(S_i = x \mid S_j = s_j : j \neq i)$$
$$\propto P(S_i = x \mid S_j = s_j : j < i)$$
$$\cdot \prod_{j>i} P(S_j = s_j \mid S_i = x,\ S_k = s_k : k < j,\ k \neq i)$$

$$P(S_i = s_i \mid S_j = s_j : j < i) = \sigma\left(s_i^* \sum_{j<i} s_j w_{ij}\right)$$

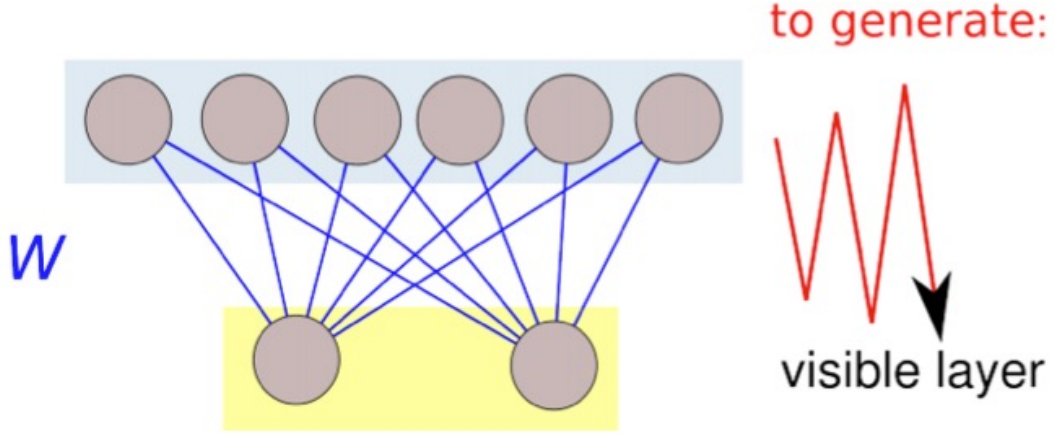
RBMs are infinite belief networks (with tied weights)

Since none of the units within a layer are interconnected, we can do Gibbs sampling by updating the whole layer at a time.

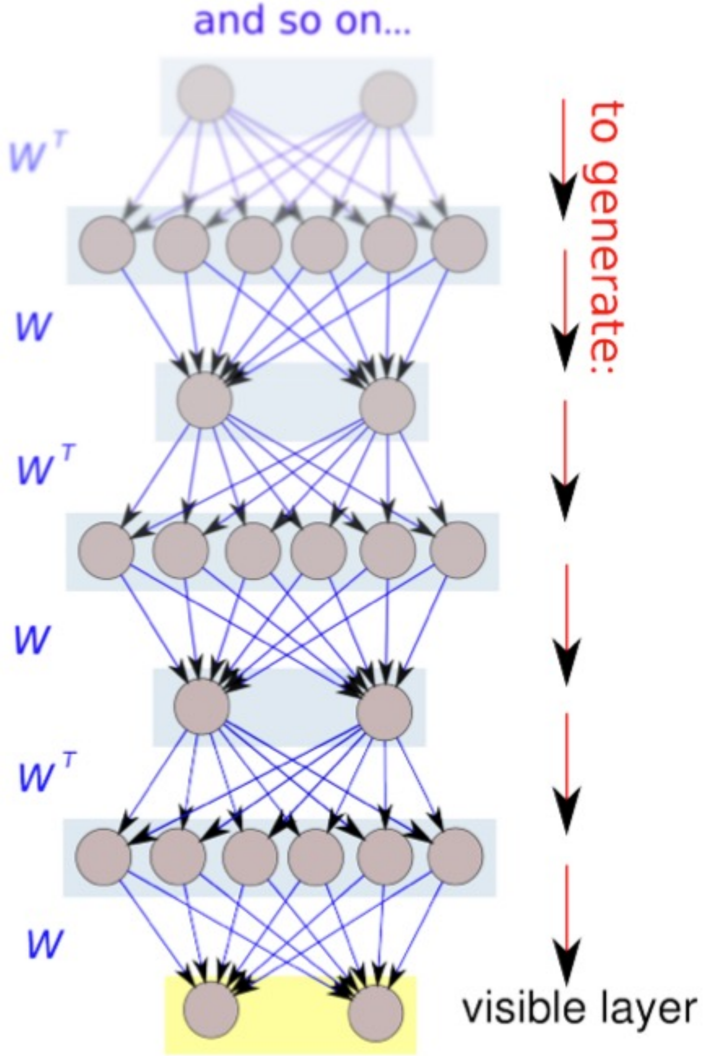


(with time running from left ⟶ right)

RBMs are infinite belief networks (with tied weights)



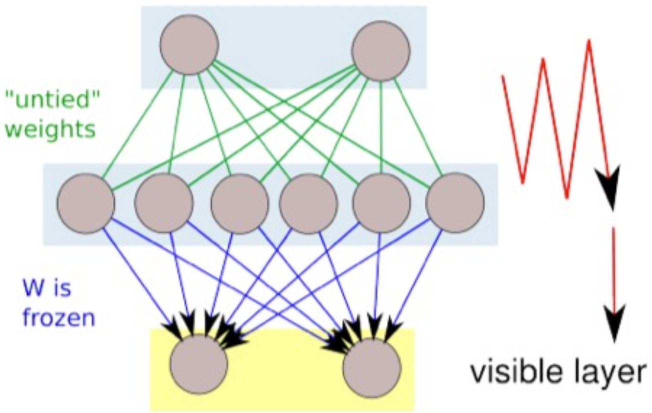
sampling from this is the same as sampling from the network on the right.

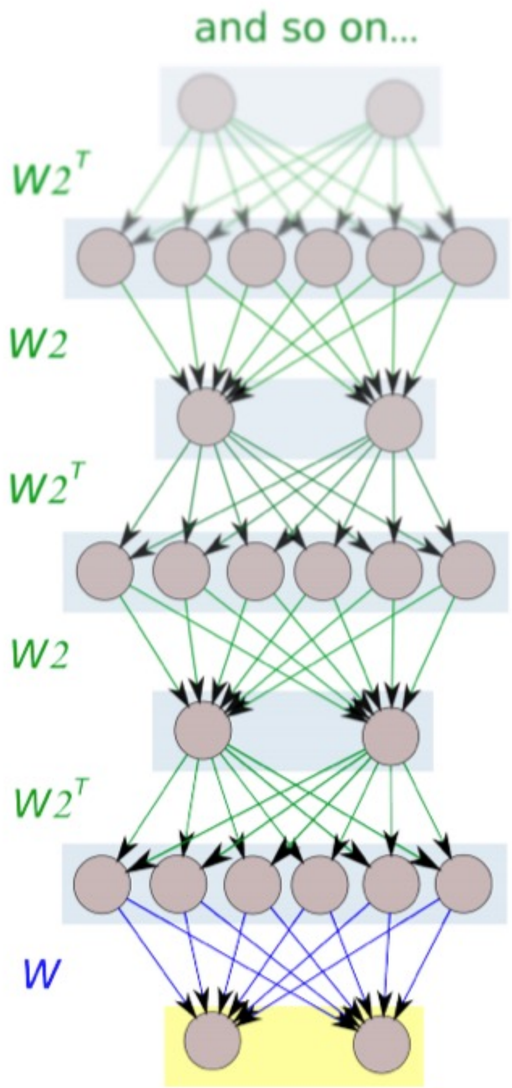


Deep Belief networks: layer-wise pre-training

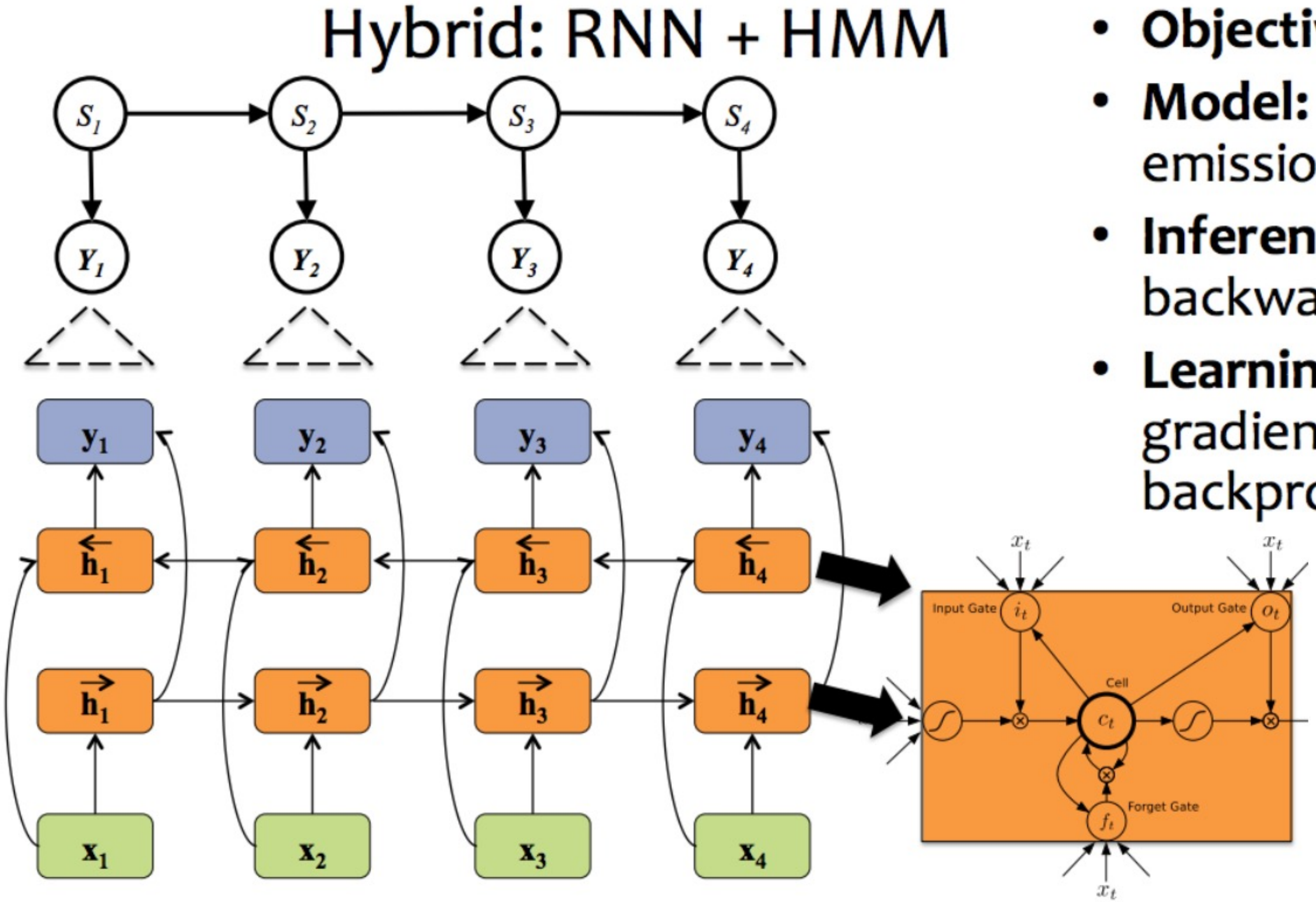
Un-tie the weights from layers 2 to infinity

If we freeze the first RBM, and then train another RBM atop it, we are untying the weights of layers 2+ in the ∞ net (which remain tied together).





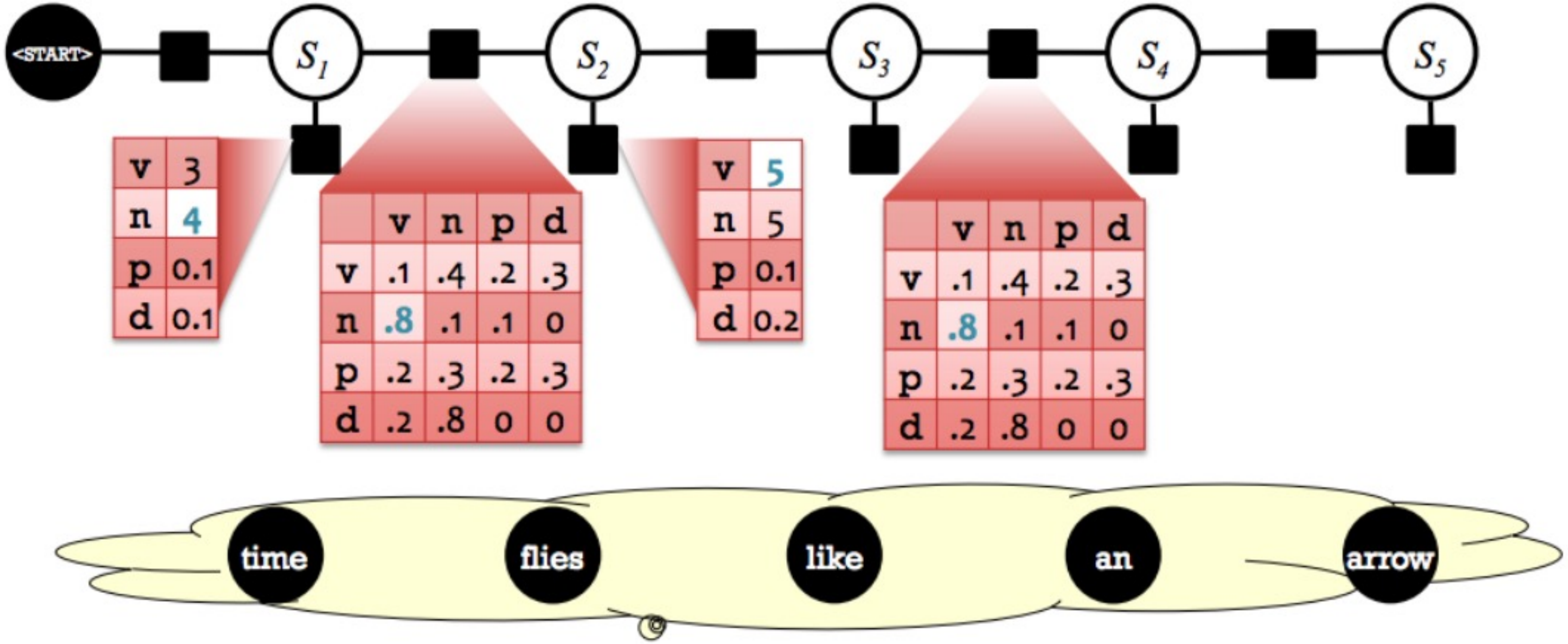
NNs and GMs: Natural Complements



- **Objective:** log-likelihood
- **Model:** HMM/Gaussian emissions
- **Inference:** forward-backward algorithm
- **Learning:** SGD with gradient by backpropagation

[Graves et al. 2013]

NNs and GMs: Natural Complements



- In a standard CRF, each of the factor cells is a parameter (e.g. transition or emission)
- In the hybrid model, these values are computed by a neural network with its own parameters

[Collobert & Weston 2011]

Looking ahead

Module 3: Modern Probabilistic AI		
4/1	Lecture #16 (Prof. Lengerich): **Deep Learning from a GM Perspective** [slides I notes]	• Goodfellow et al., Deep learning book, Ch. 6.2-5, 20.3-4 • Salakhutdinov and Hinton, Deep Boltzmann Machines • Ranganath et al., Deep exponential families
4/3	Lecture #17 (Prof. Lengerich): **CNNs, RNNs, Autoencoders** [slides I notes]	• Pascanu, Mikolov, Bengio, On the difficulty of training recurrent neural networks
4/8	Lecture #18 : **Deep Generative Models: GAN, VAEs** [slides I notes]	• Goodfellow et al., Deep learning book, Ch. 20.9-10 • Kingma and Welling, Variational Autoencoders • Goodfellow et al., Generative Adversarial Nets • Arora., Generative Adversarial Networks (GANs), Some Open Questions
4/10	Lecture #19 (Prof. Lengerich): **Attention and Transformers** [slides I notes]	• Vasvani et al., Attention is all you need • Devlin et al., BERT - Pre-training of Deep Bidirectional Transformers for Language Understanding. • Raschka, Build an LLM from Scratch 3 (video) • Sanderson, Visualizing transformers and attention (video)
4/15	Lecture #20 (Prof. Lengerich): **LLMs from a Probabilistic Perspective 1: Implementing a GPT from Scratch** [slides I notes]	• Radford et al., Improving Language Understanding by Generative Pre-Training (the GPT-1 paper) • Radford et al., Language Models are Unsupervised Multitask Learners (the GPT-2 paper) • Brown et al., Language Models are Few-Shot Learners (the GPT-3 paper) • Raschka, Build an LLM from Scratch 4 (video) • Karpathy, Let's Build GPT from Scratch (video)
4/17	Lecture #21 (Prof. Lengerich): **LLMs from a Probabilistic Perspective 2: Training on Unlabeled Data** [slides I notes]	• Bi. et al, DeepSeek LLM Scaling Open-Source Language Models with Longtermism • Liu et al., DeepSeekV2 A Strong, Economical, and Efficient Mixture-of-Experts Language Model • Liu et al., DeepSeekV3 Technical Report
4/22	Lecture #22 (Prof. Lengerich): **LLMs from a Probabilistic Perspective 3: Fine-tuning on Labeled Data** [slides I notes]	• Raffel et al., Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer • Ouyang et al., Training language models to follow instructions with human feedback • Li & Liang, Prefix-tuning - Optimizing continuous prompts for generation
4/24	Lecture #23 (Prof. Lengerich): **Context-Adaptive Graphical Models** [slides I notes]	• Lengerich et al., Contextualized Machine Learning
4/29		Project Presentations
5/1		Project Presentations

Questions?

WISCONSIN
UNIVERSITY OF WISCONSIN–MADISON