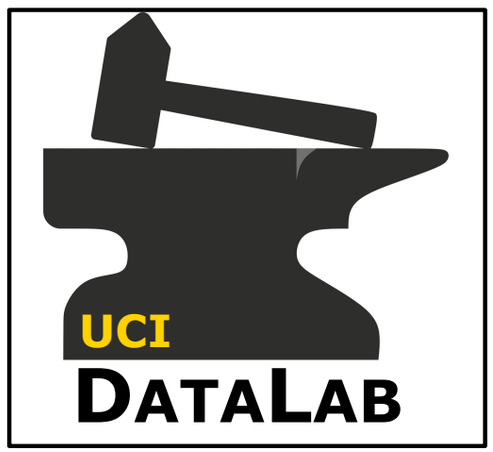
Deep Learning: A Synthesis from Probabilistic Foundations

Eric Nalisnick and **Padhraic Smyth**

University of California, Irvine



A Familiar Story

Your collaborator in medicine/astronomy/social science/etc:

"*Do we really need to build these statistical models with all this complicated math and p-values?*

A Familiar Story

Your collaborator in medicine/astronomy/social science/etc:

"*Do we really need to build these statistical models with all this complicated math and p-values?*

My grad student tells me we could just build a deep network using TensorFlow and solve all our problems.

A Familiar Story

Your collaborator in medicine/astronomy/social science/etc:

"*Do we really need to build these statistical models with all this complicated math and p-values?*

My grad student tells me we could just build a deep network using TensorFlow and solve all our problems.

And I read a really cool article about deep learning in the New York Times yesterday"

A Familiar Story

Your collaborator in medicine/astronomy/social science/etc:

"*Do we really need to build these statistical models with all this complicated math and p-values?*

My grad student tells me we could just build a deep network using TensorFlow and solve all our problems.

And I read a really cool article about deep learning in the New York Times yesterday"

You: "*Umm……*"

Motivation for This Talk

- Deep learning/neural networks are attracting a lot of attention

  …but language of deep learning is alien to a statistical audience

Motivation for This Talk

- Deep learning/neural networks are attracting a lot of attention

  …but language of deep learning is alien to a statistical audience

- Nonetheless, deep learning has statistical ideas at the core

  This talk: bridge the gap between statistics and deep learning

Motivation for This Talk

- [ ] Deep learning/neural networks are attracting a lot of attention

  …but language of deep learning is alien to a statistical audience

- [ ] Nonetheless, deep learning has statistical ideas at the core

  This talk: bridge the gap between statistics and deep learning

- [ ] **Caveat:** Deep learning is a huge field….and this is a 50 minute talk
  (not speaking about convolutional neural networks, for instance)

Audience

- **Applied statisticians:** Be aware of…
  - (a) what deep learning is.
  - (b) what it can do well.
  - (c) what its limits are.

- **Research statisticians**: Opportunities to…
  - (a) use deep learning ideas in statistical modeling.
  - (b) bring more statistical thinking to deep learning.

Notation: A Simple Prediction Problem

- Data Set : $\{(\mathbf{x}_i, y_i)\}, \quad i = 1, \ldots, N$

  Would like to build a model to predict *y* given ***x***

  Use a model f(***x***; ***W***) with unknown parameters ***W***

Notation: A Simple Prediction Problem

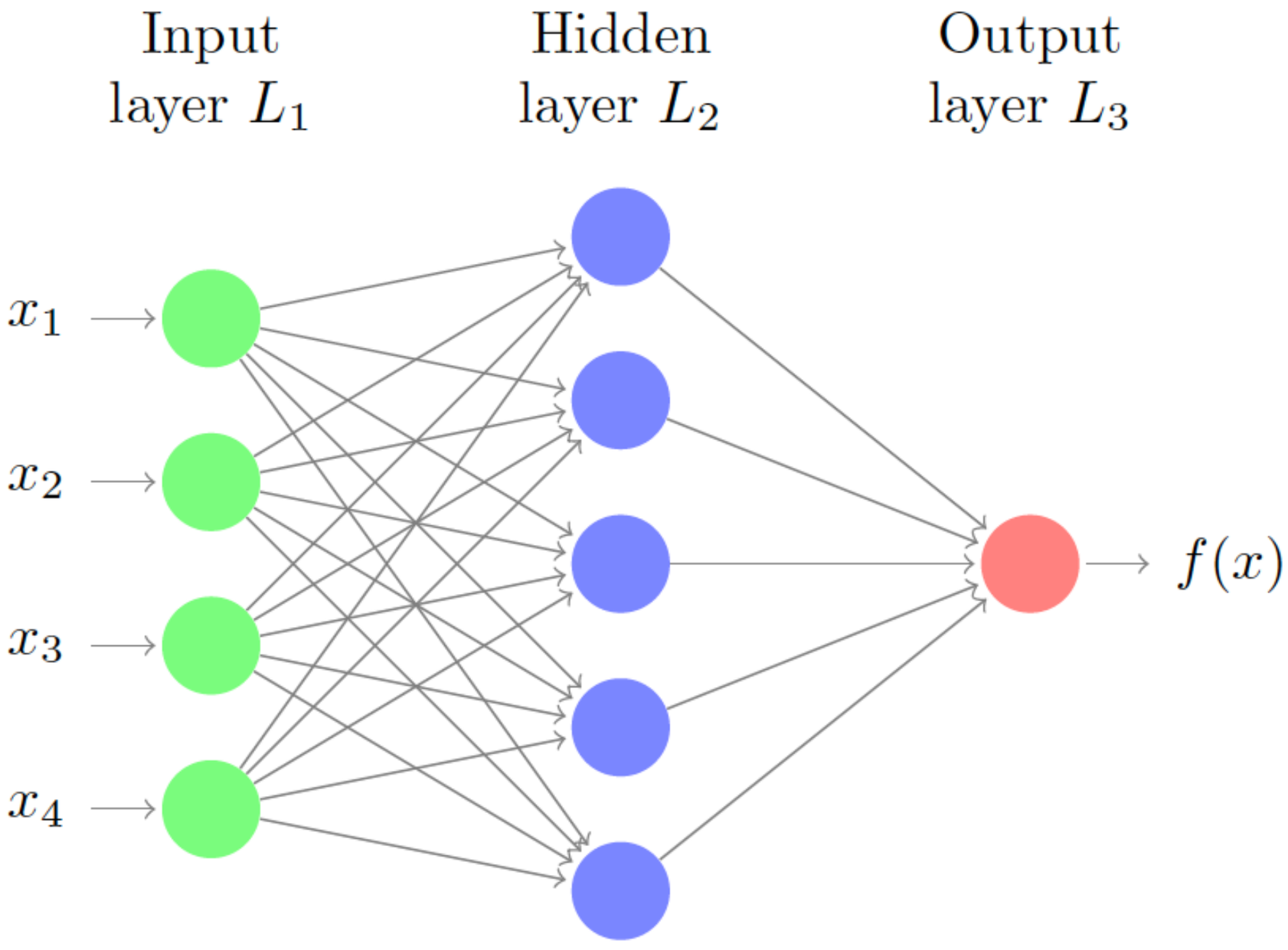
- Data Set : $\{(\mathbf{x}_i, y_i)\}, \quad i = 1, \ldots, N$

  Would like to build a model to predict *y* given ***x***

  Use a model f(***x***; ***W***) with unknown parameters ***W***

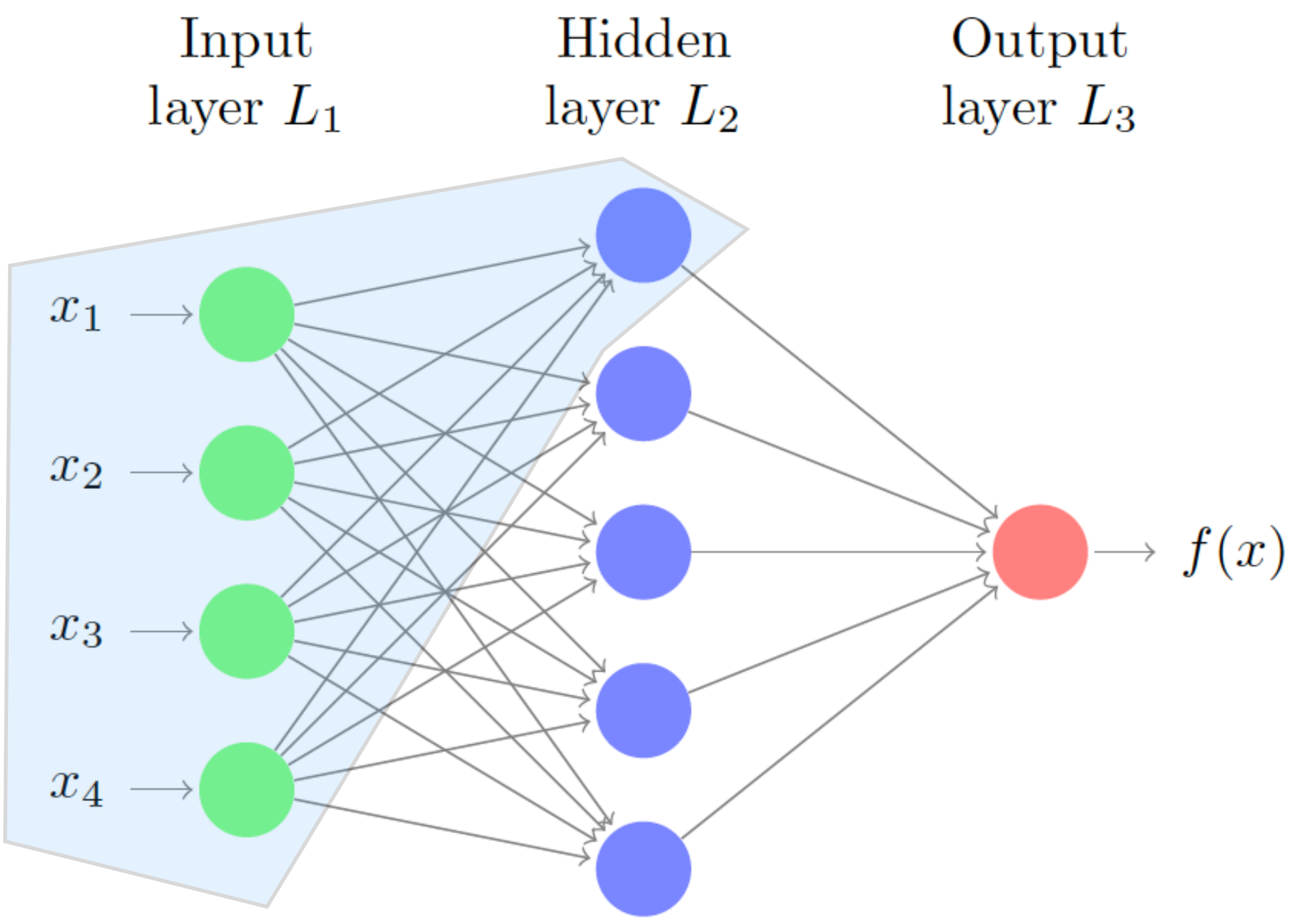
- **Example:**
  x = high-dimensional data (e.g., pixels)
  y = categorical label

  Could use a logistic (linear) model…
  ….. but would like to explore more complex models

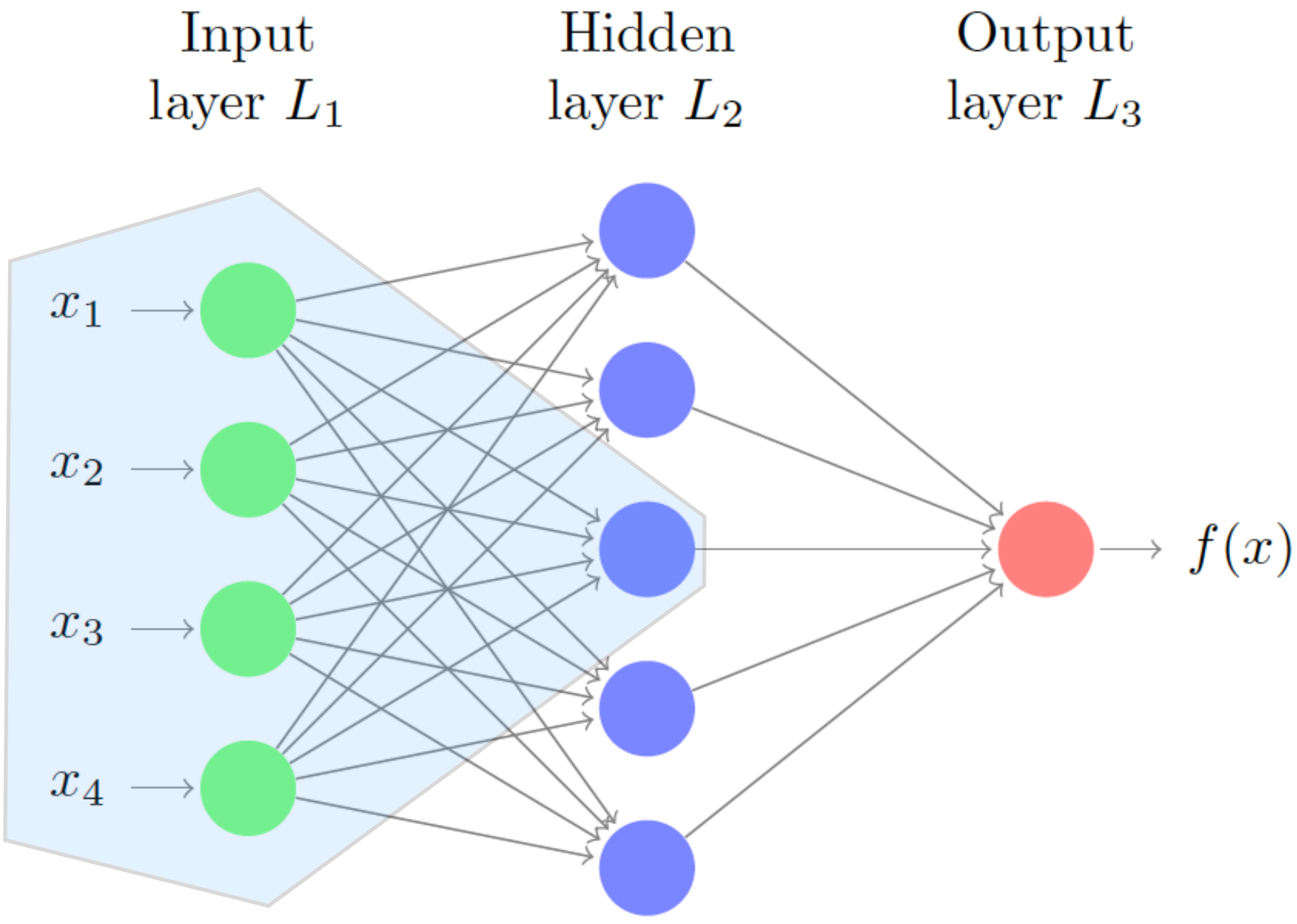


State-of-the-art Neural Network in early 1990's

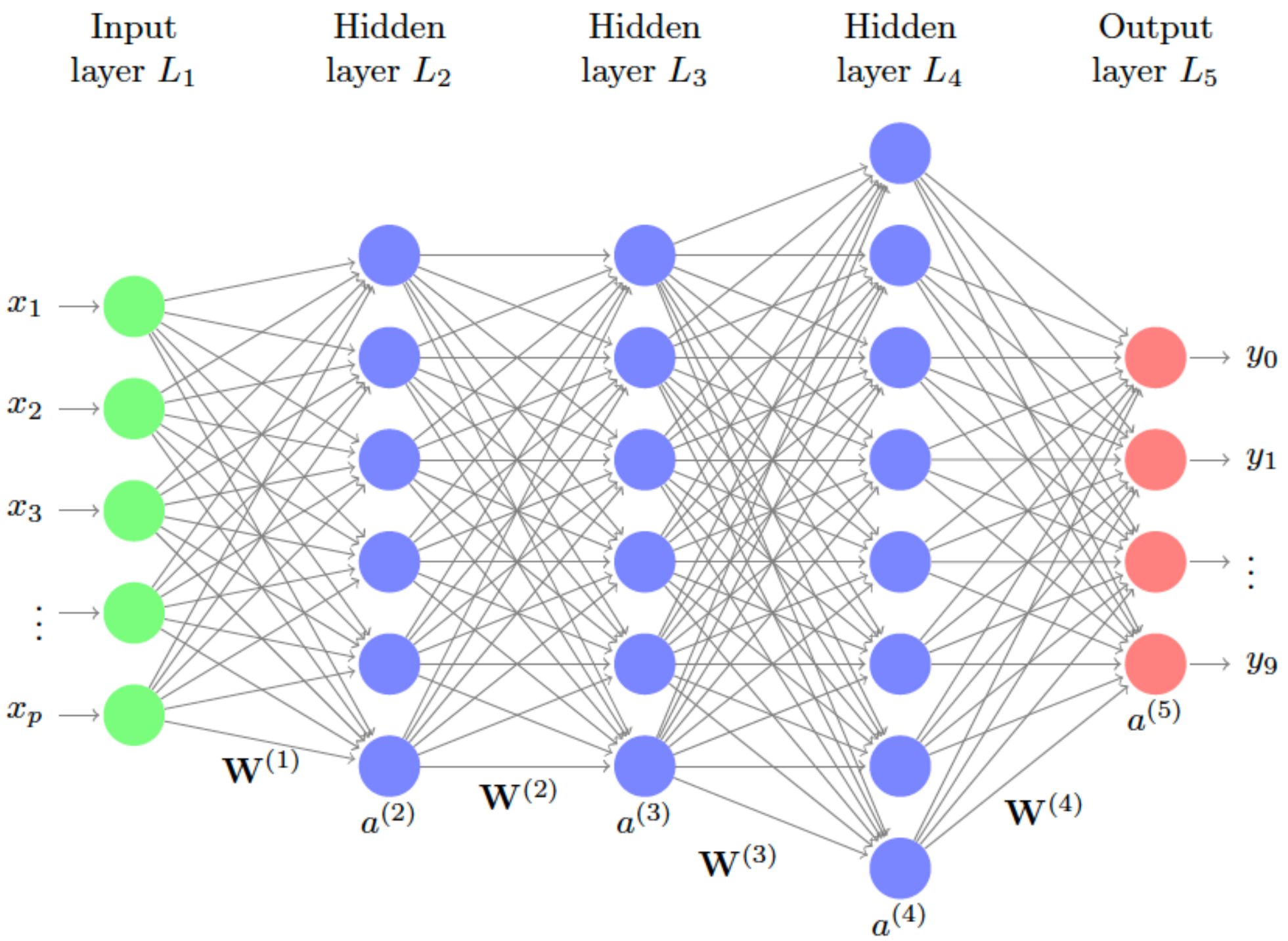
From Efron and Hastie, Computer Age Statistical Inference, 2016



From Efron and Hastie, Computer Age Statistical Inference, 2016

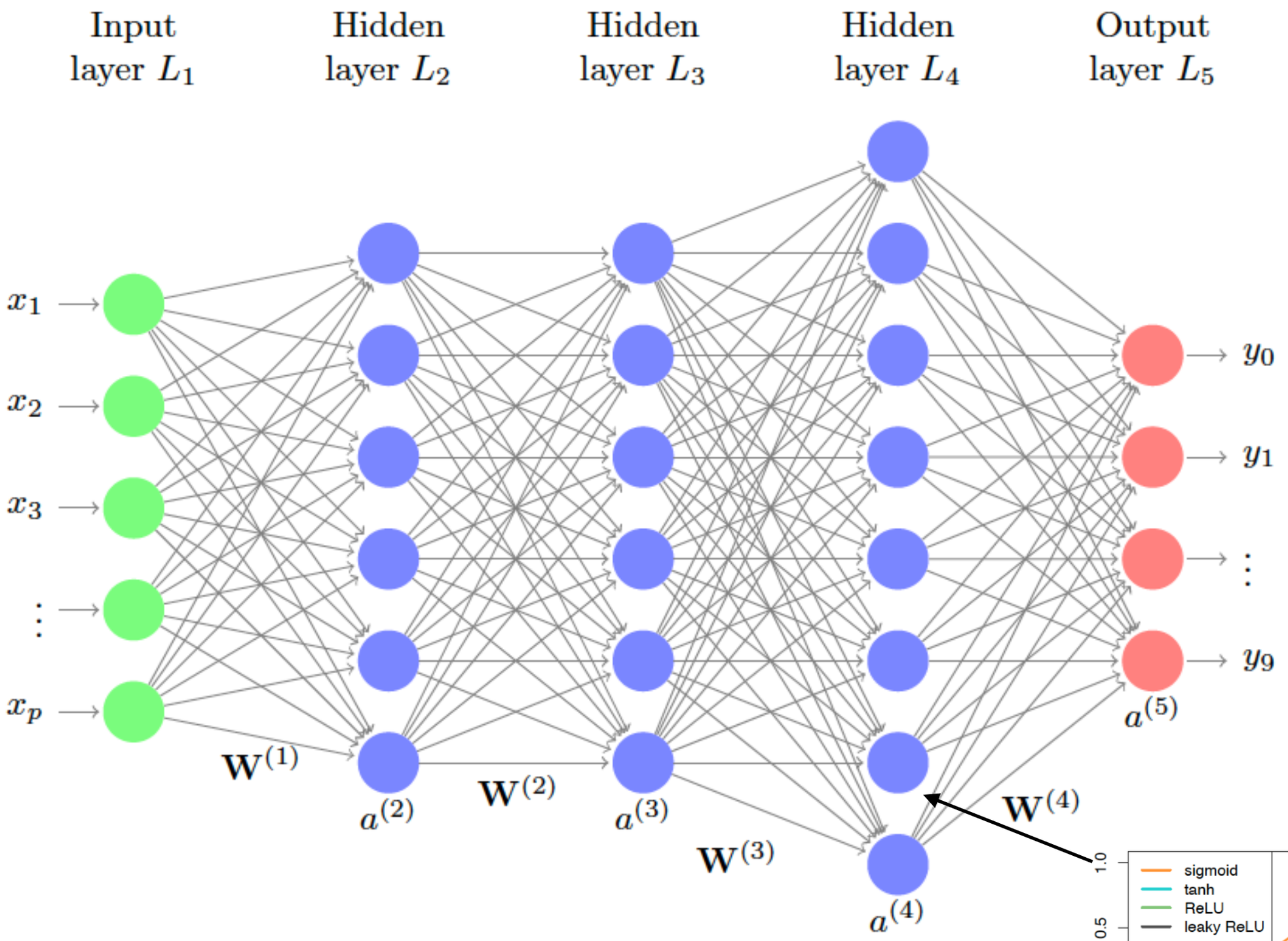


From Efron and Hastie, Computer Age Statistical Inference, 2016

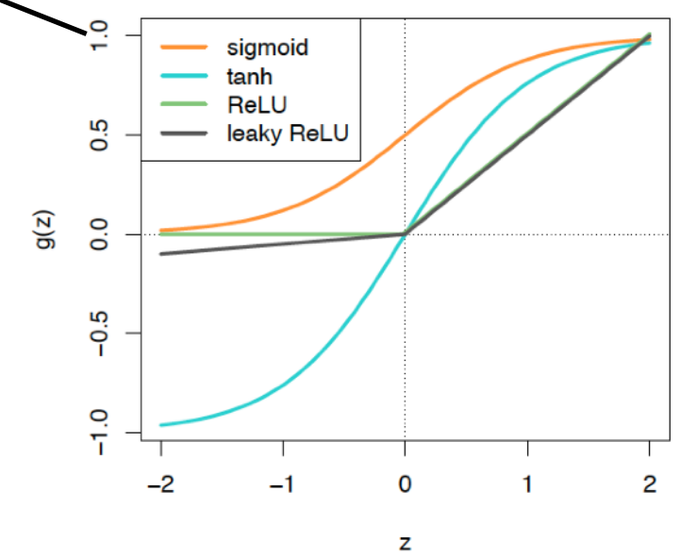


Deep Neural Network in 2018

From Efron and Hastie, Computer Age Statistical Inference, 2016



Deep Neural Network in 2018



From Efron and Hastie, Computer Age Statistical Inference, 2016

Example of a Network for Image Recognition

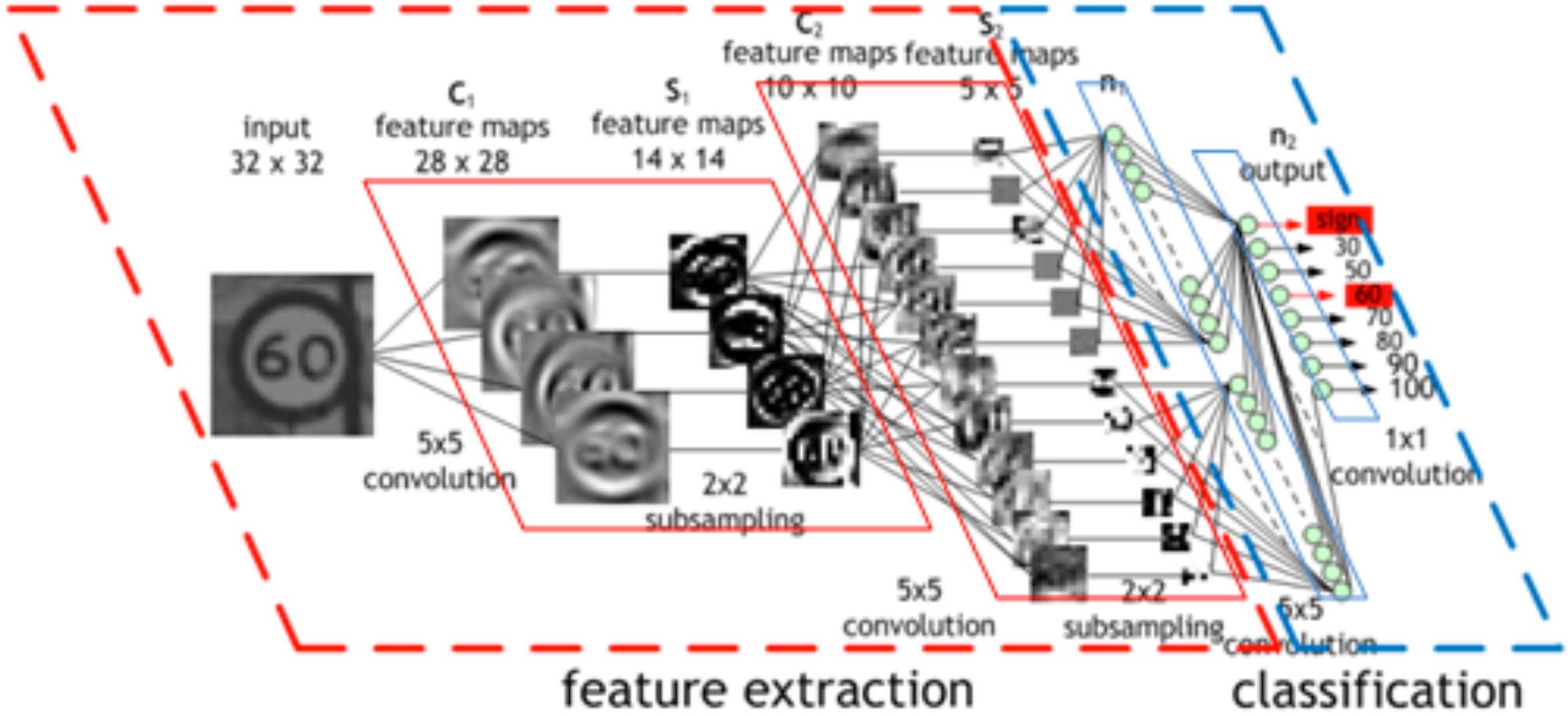
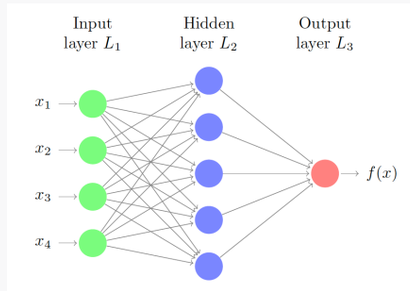


Figure from https://developer.nvidia.com/discover/convolutionalneuralnetwork

How are the Weights Estimated?

Simple Network Model

$$f(\mathbf{x}; \mathbf{W}) = \text{logistic}\left(\sum_{k=1}^{H} w_k^o \; h_k(\mathbf{w}_k^T \mathbf{x}) \right)$$



Model output

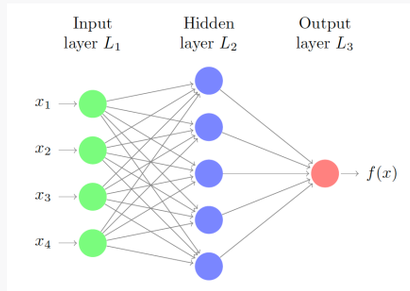
Weights/Parameters

Output of kth hidden unit

How are the Weights Estimated?

Simple Network Model

$$f(\mathbf{x};\mathbf{W}) = \text{logistic}\left(\sum_{k=1}^{H} w_k^o \; h_k(\mathbf{w}_k^T\mathbf{x})\right)$$



Model output

Weights/Parameters

Output of kth hidden unit

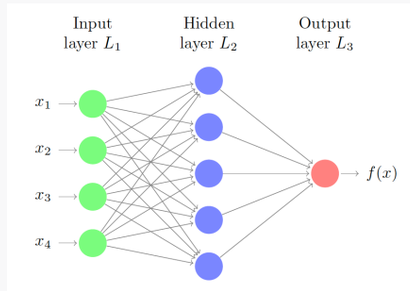
Optimization Problem

$$\text{minimize}\left\{ -\sum_{i=1}^{N} \log f(\mathbf{x}_i;\mathbf{W}) + \lambda J(\mathbf{W}) \right\}$$

How are the Weights Estimated?

Simple Network Model

$$f(\mathbf{x};\mathbf{W}) = \text{logistic}\left(\sum_{k=1}^{H} w_k^o \; h_k(\mathbf{w}_k^T\mathbf{x})\right)$$



Model output

Weights/Parameters

Output of kth hidden unit

Optimization Problem

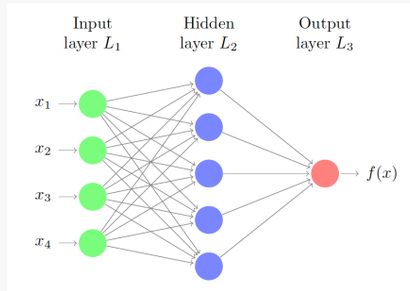
$$\text{minimize}\left\{ -\sum_{i=1}^{N} \log f(\mathbf{x}_i;\mathbf{W}) + \lambda J(\mathbf{W}) \right\}$$

Negative Conditional Log-likelihood

How are the Weights Estimated?

Simple Network Model

$$f(\mathbf{x}; \mathbf{W}) = \text{logistic}\left(\sum_{k=1}^{H} w_k^o \; h_k(\mathbf{w}_k^T \mathbf{x}) \right)$$



Model output

Weights/Parameters

Output of kth hidden unit

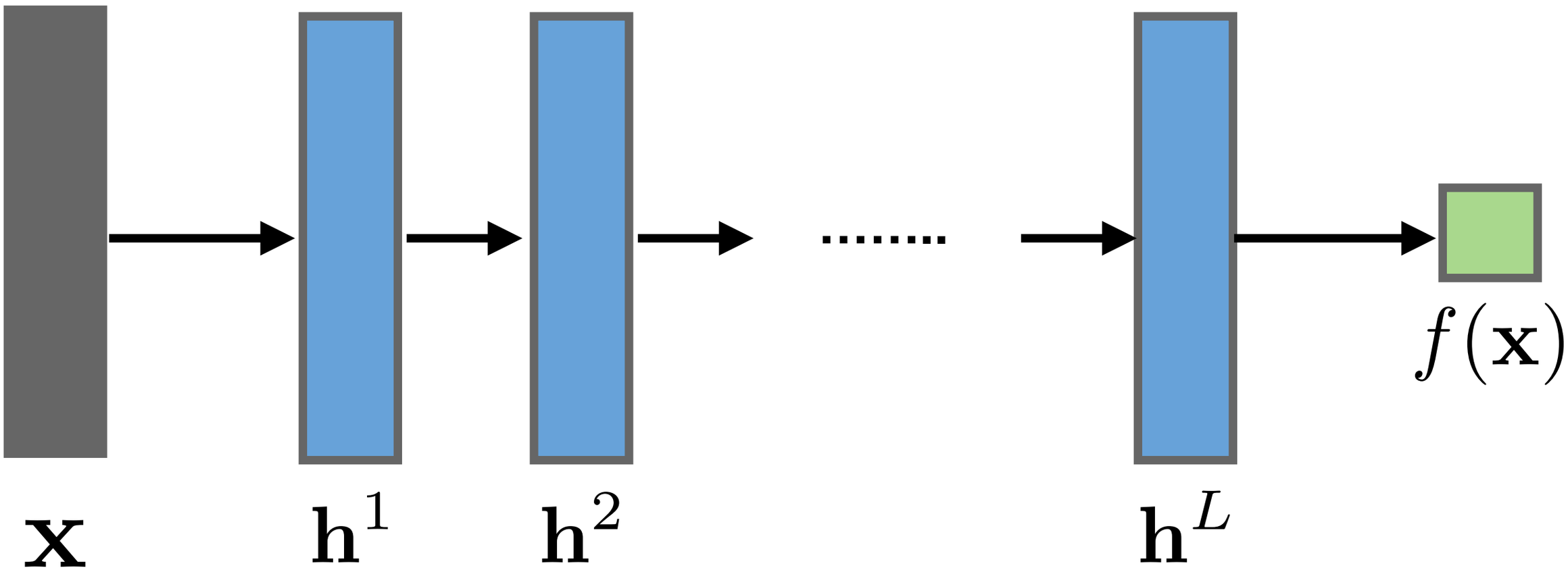
Optimization Problem

Regularization -> Log Prior

$$\text{minimize}\left\{ -\sum_{i=1}^{N} \log f(\mathbf{x}_i; \mathbf{W}) + \lambda J(\mathbf{W}) \right\}$$

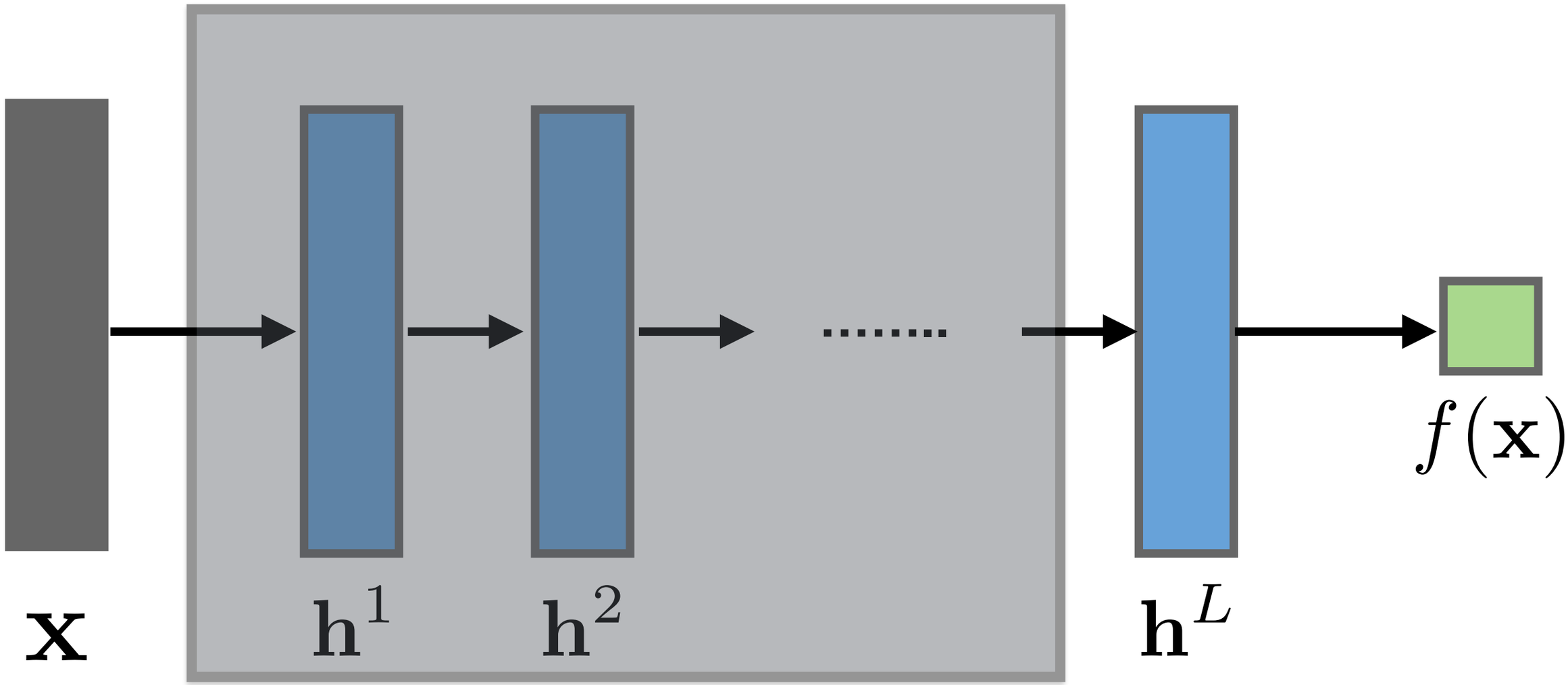
Negative Conditional Log-likelihood

Deep Neural Networks



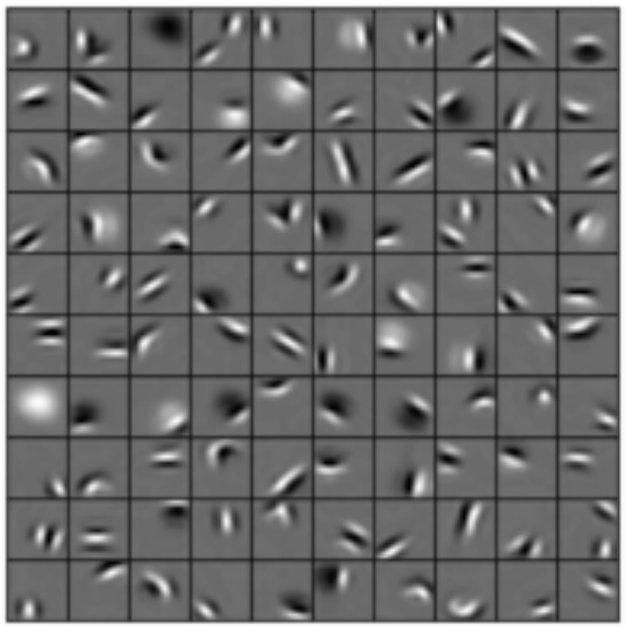
Parameters = matrices of weights between layers

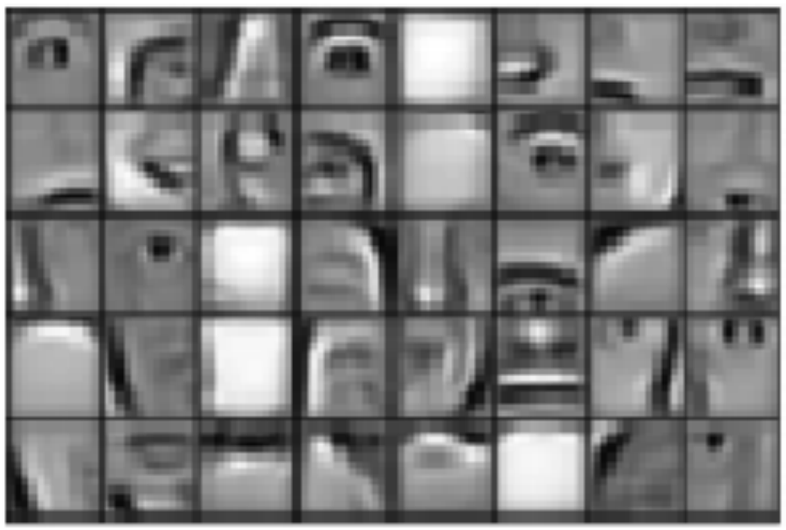
Deep Neural Networks



Feature Extractor **Statistical Model**

What Representations do Networks Learn?



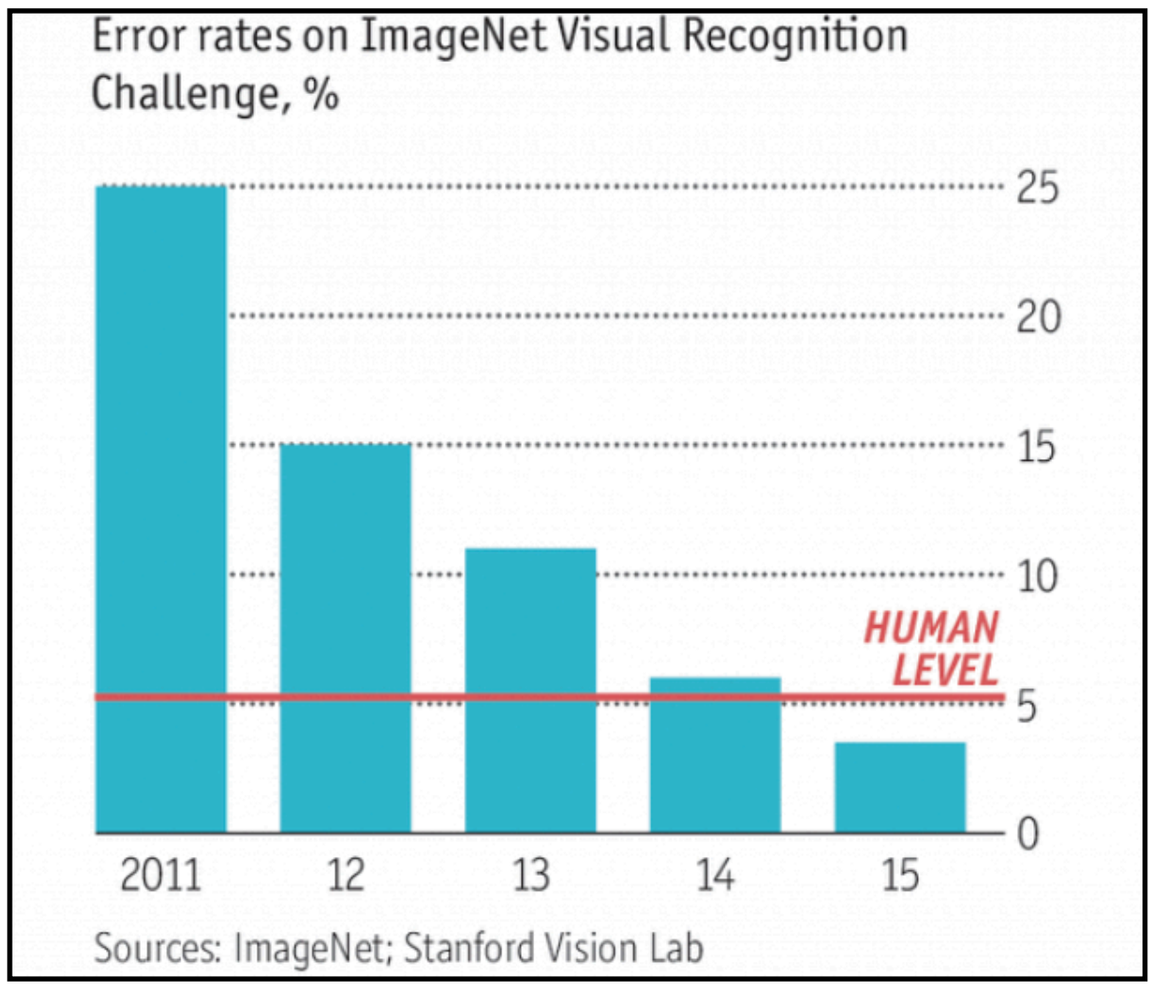


Weight Matrices (or Filters) Learned at Different Layers for Face Detection

Figures from Lee et al, ICML 2009

Deep Learning Results

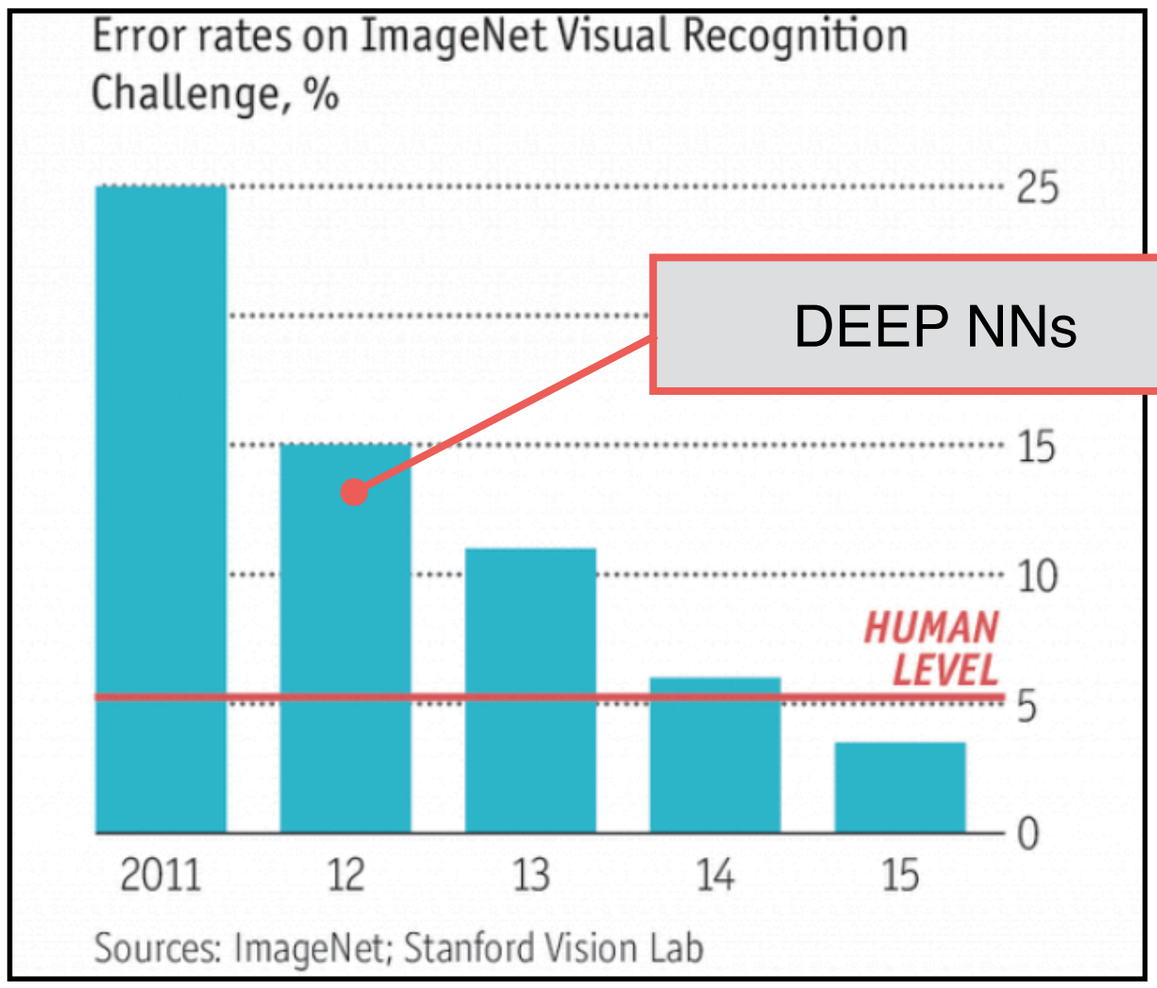
- **Computer Vision**: Results on ImageNet object classification dataset.



https://www.economist.com/news/special-report/21700756-artificial-intelligence-boom-based-old-idea-modern-twist-not

Deep Learning Results

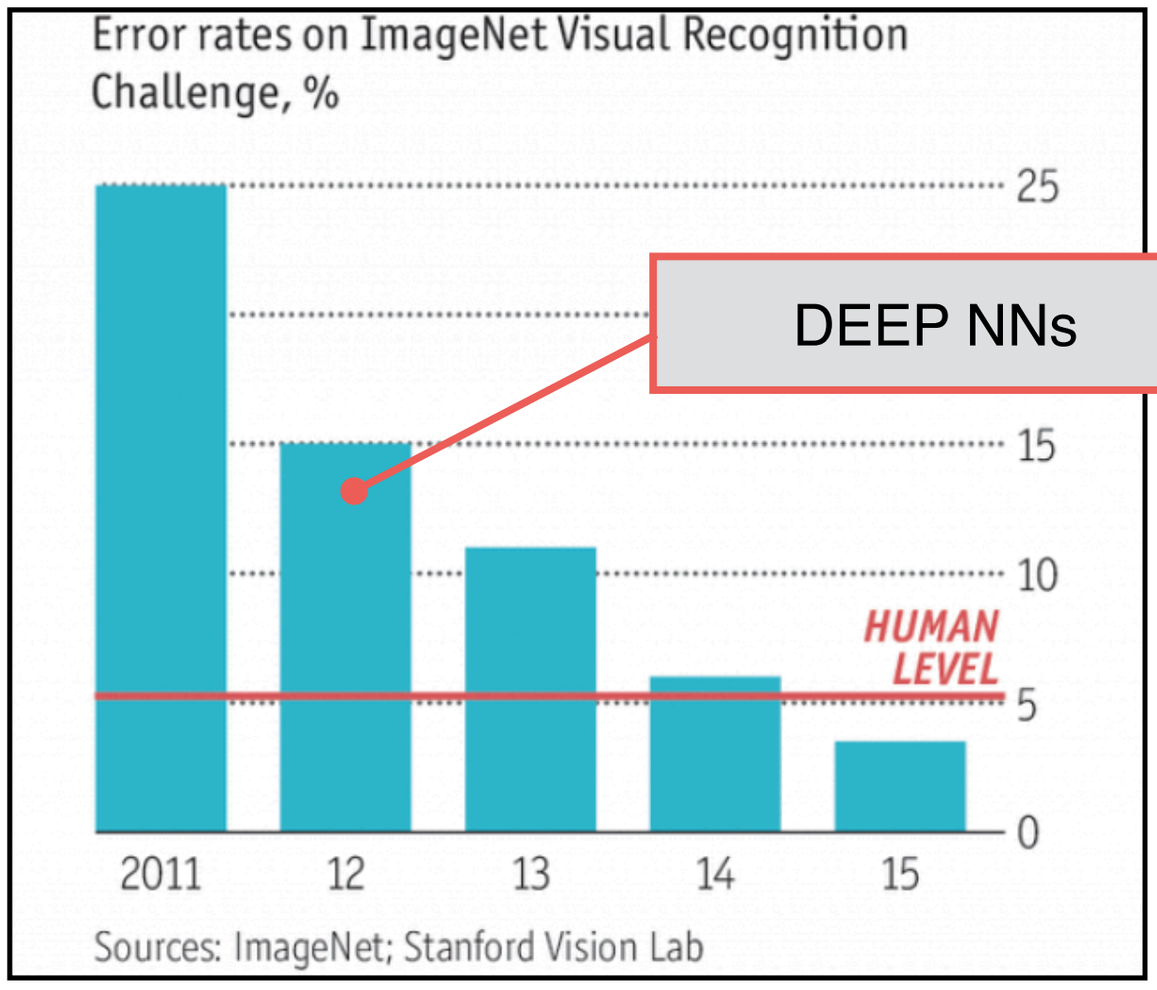
- **Computer Vision**: Results on ImageNet object classification dataset.

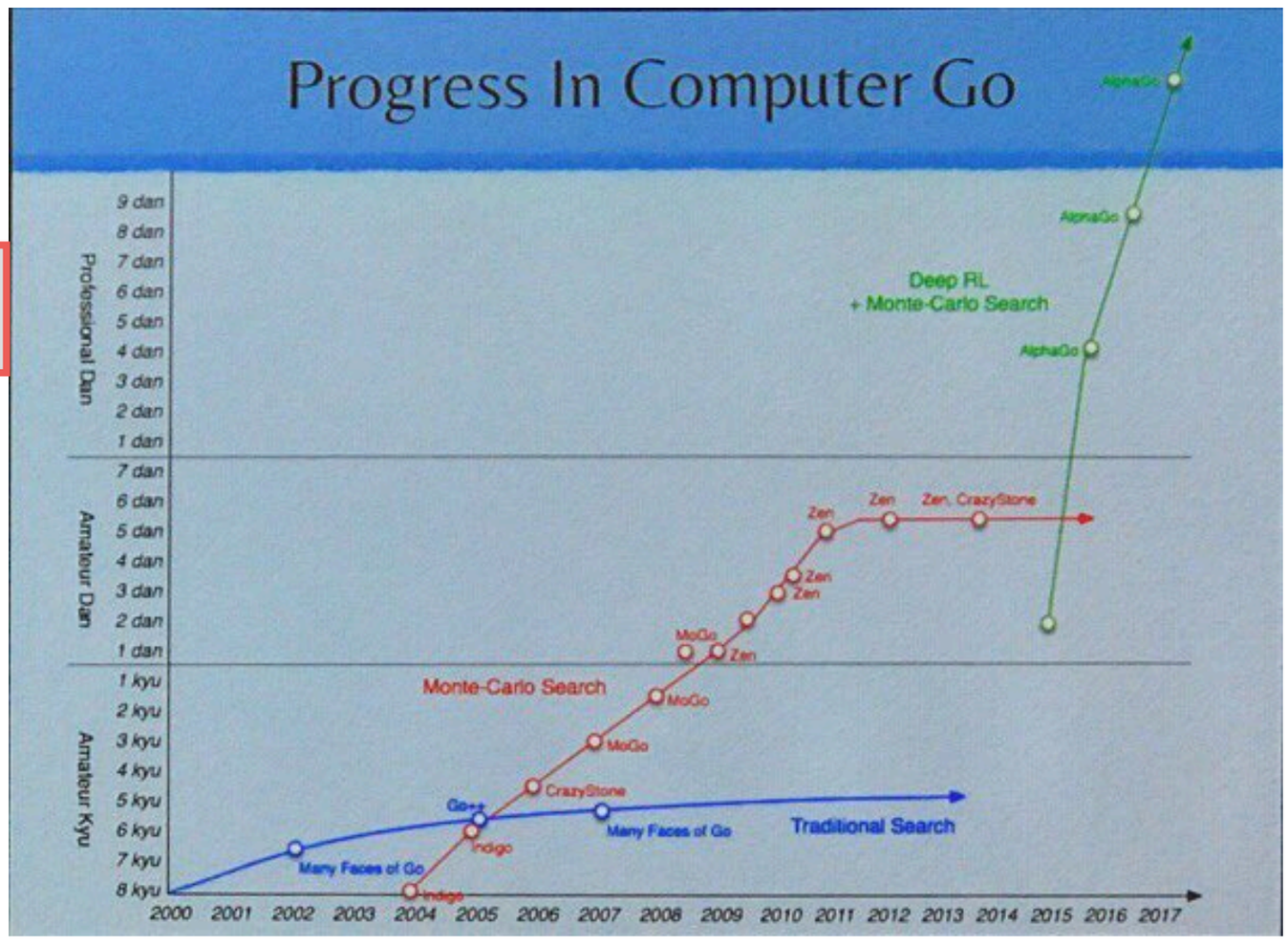


https://www.economist.com/news/special-report/21700756-artificial-intelligence-boom-based-old-idea-modern-twist-not

Deep Learning Results

- **Computer Vision**: Results on ImageNet object classification dataset.
- **Reinforcement Learning**: Results in playing Go.

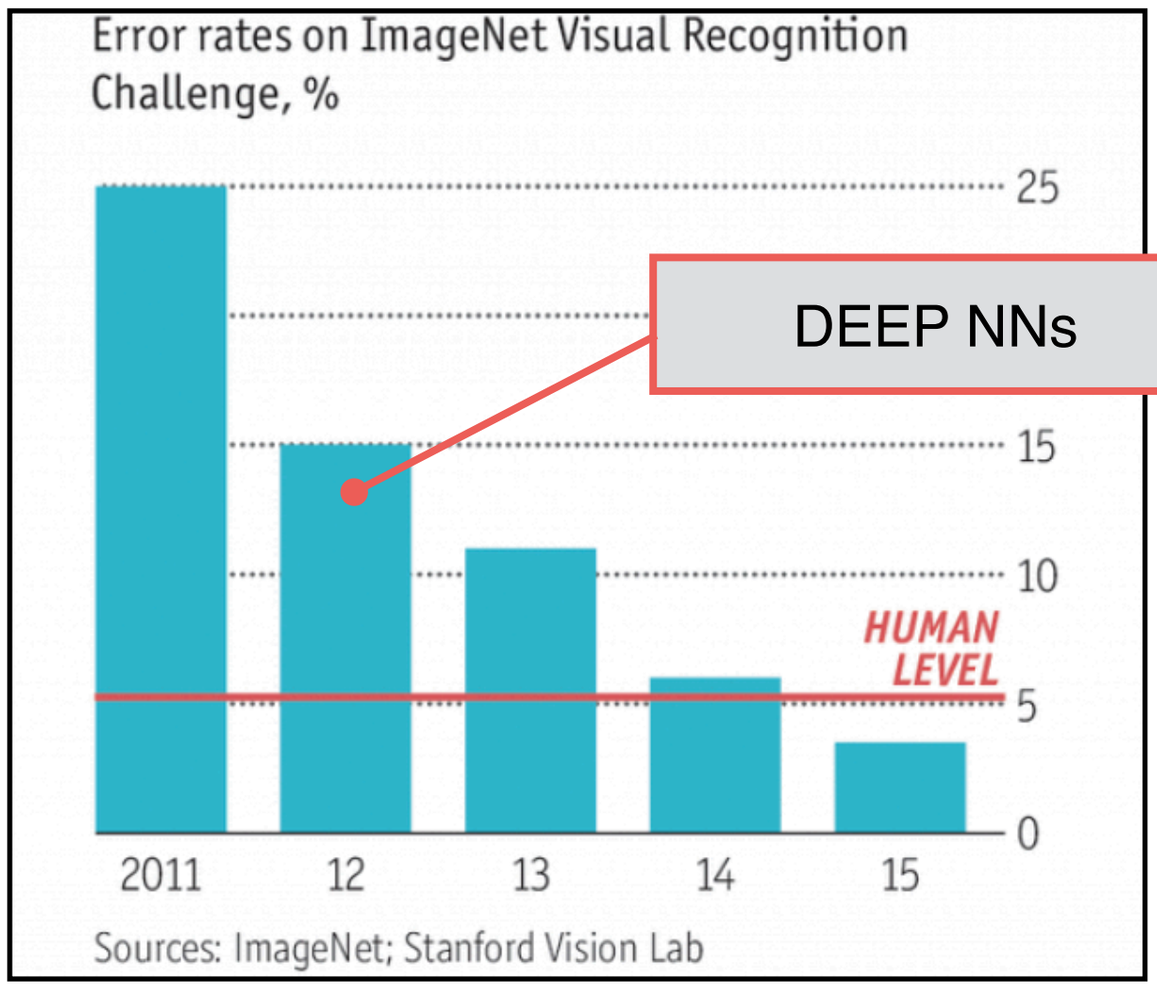




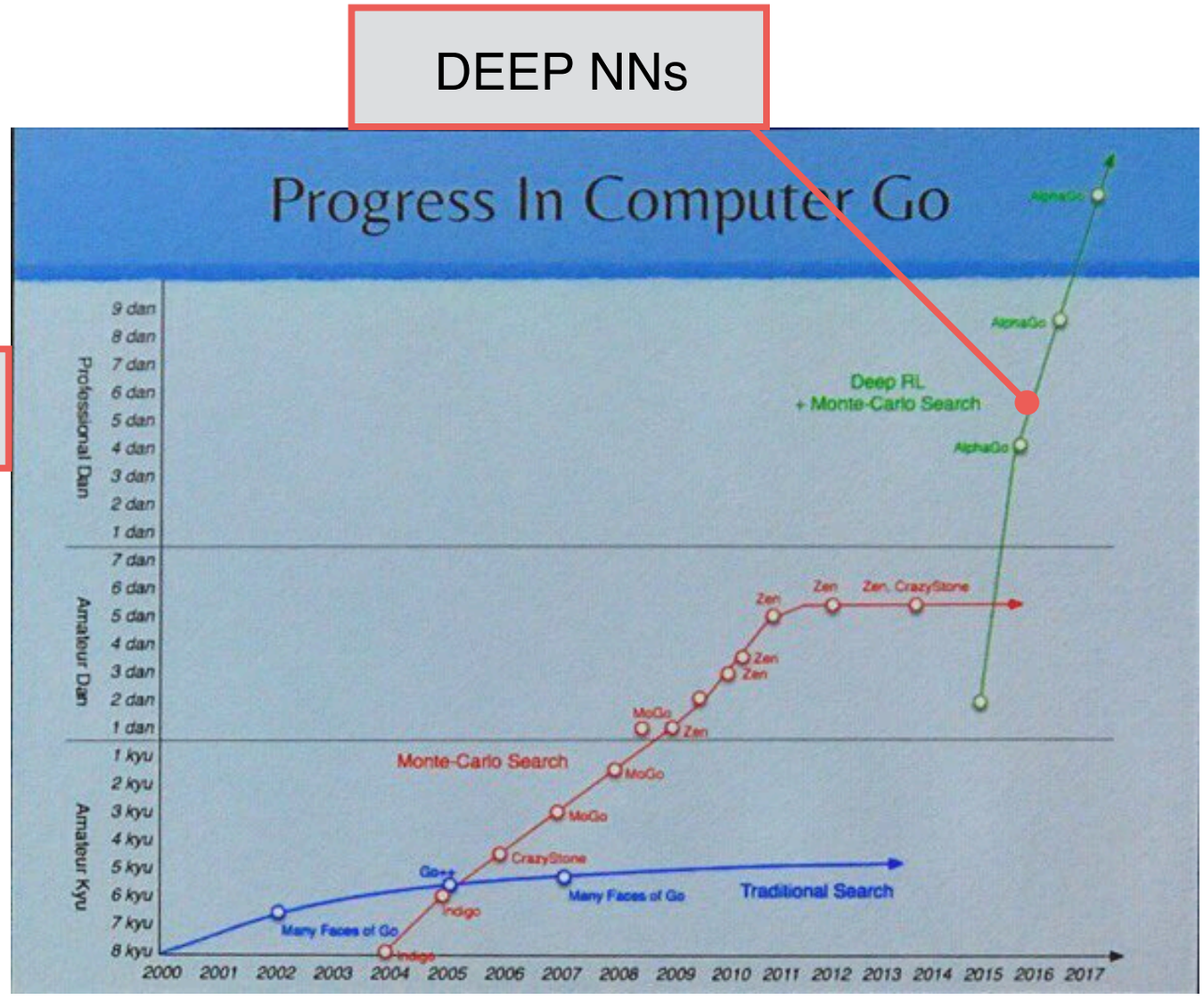
https://www.economist.com/news/special-report/21700756-artificial-intelligence-boom-based-old-idea-modern-twist-not

Deep Learning Results

- **Computer Vision**: Results on ImageNet object classification dataset.



- **Reinforcement Learning**: Results in playing Go.



https://www.economist.com/news/special-report/21700756-artificial-intelligence-boom-based-old-idea-modern-twist-not

Some Observations

- **A deep network is a mathematical function with parameters** (in effect a very flexible non-linear regression model)
- **The hidden layers act as latent feature extractors** - but they are deterministic, no distributional semantics (unlike statistical models)

Some Observations

- [ ] **A deep network is a mathematical function with parameters** (in effect a very flexible non-linear regression model)

- [ ] **The hidden layers act as latent feature extractors** - but they are deterministic, no distributional semantics (unlike statistical models)

- [ ] **There can be millions of parameters**: But these are not parameters in a traditional statistical sense. Focus is typically on prediction, not parameter interpretation

- [ ] **The parameters are estimated by minimizing a loss function** - often equivalent to a negative conditional log-likelihood

More Observations

- **Often need very large amounts of labeled training data**
  (N ~ millions)

- **Significant "engineering" to get these models to work:**
  - model structure
  - hyperparameter tuning
  - optimization algorithms

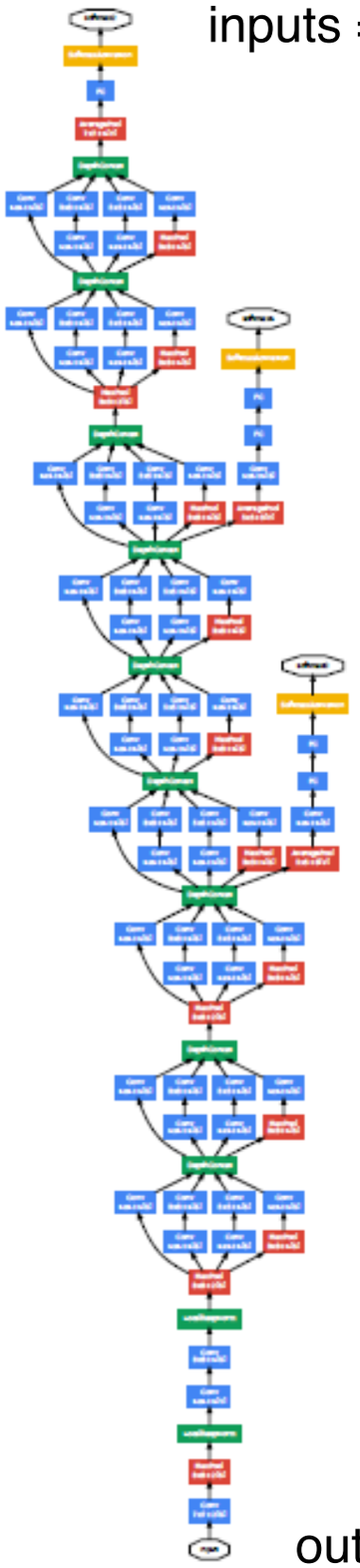
More Observations

- **Often need very large amounts of labeled training data**
  (N ~ millions)

- **Significant "engineering" to get these models to work:**
  - model structure
  - hyperparameter tuning
  - optimization algorithms

- **Many of the success stories have been in "pattern recognition":**
  - large amounts of labeled data
  - inputs are high-dimensional, need feature extraction (e.g., pixels)
  - model interpretability is not a priority
  - image analysis, speech recognition, text analysis

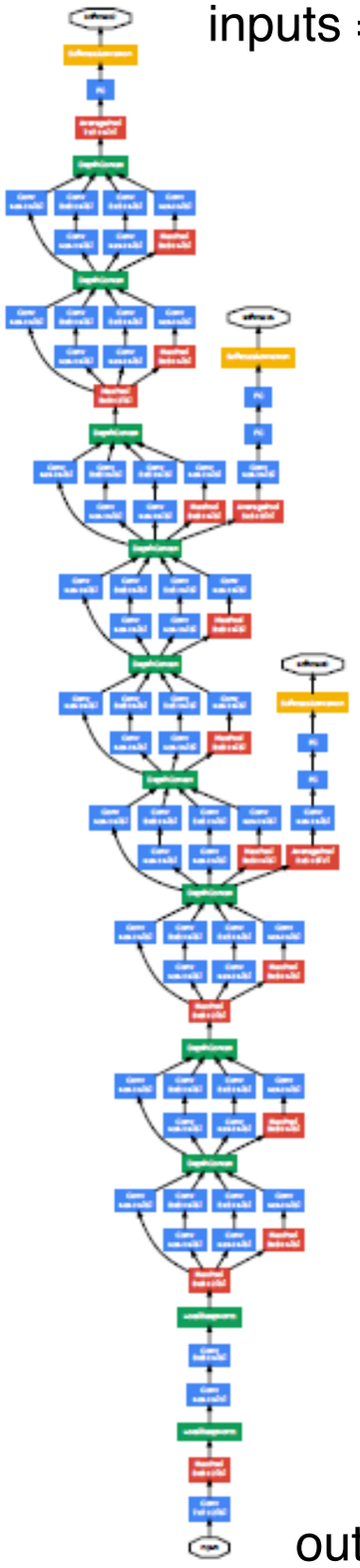
inputs = pixels



Deep network (from Google)
with 27 layers
for image recognition

outputs = P(class I pixels)

Figure from Szegedy et al, CVPR, 2015



inputs = pixels

Deep network (from Google)
with 27 layers
for image recognition

outputs = P(class I pixels)

Rube Goldberg machine

From www.cartoonaday.com

Figure from Szegedy et al, CVPR, 2015

We need principled approaches…

https://xkcd.com/1838/

Outline of Remainder of Talk

1. Neural Networks and Generalized Linear Models
2. Model Fitting via Gradient-Based Optimization
3. Autoencoders and Factor Analysis
4. Recurrent Networks and State-Space Models
5. Open Problems and Summary

Building Neural Nets from GLMs

Generalized Linear Models (GLMs)

$$\mathbf{x}_i \in \mathbb{R}^d, \mathbf{w} \in \mathbb{R}^d, b \in \mathbb{R}$$

Link Function

$$g^{-1}(\mathbf{x}_i^T \mathbf{w} + b) = \mathbb{E}[\mathbf{y}_i | \mathbf{x}_i]$$

Generalized Linear Models (GLMs)

$$\mathbf{x}_i \in \mathbb{R}^d, \mathbf{w} \in \mathbb{R}^d, b \in \mathbb{R}$$

Link Function

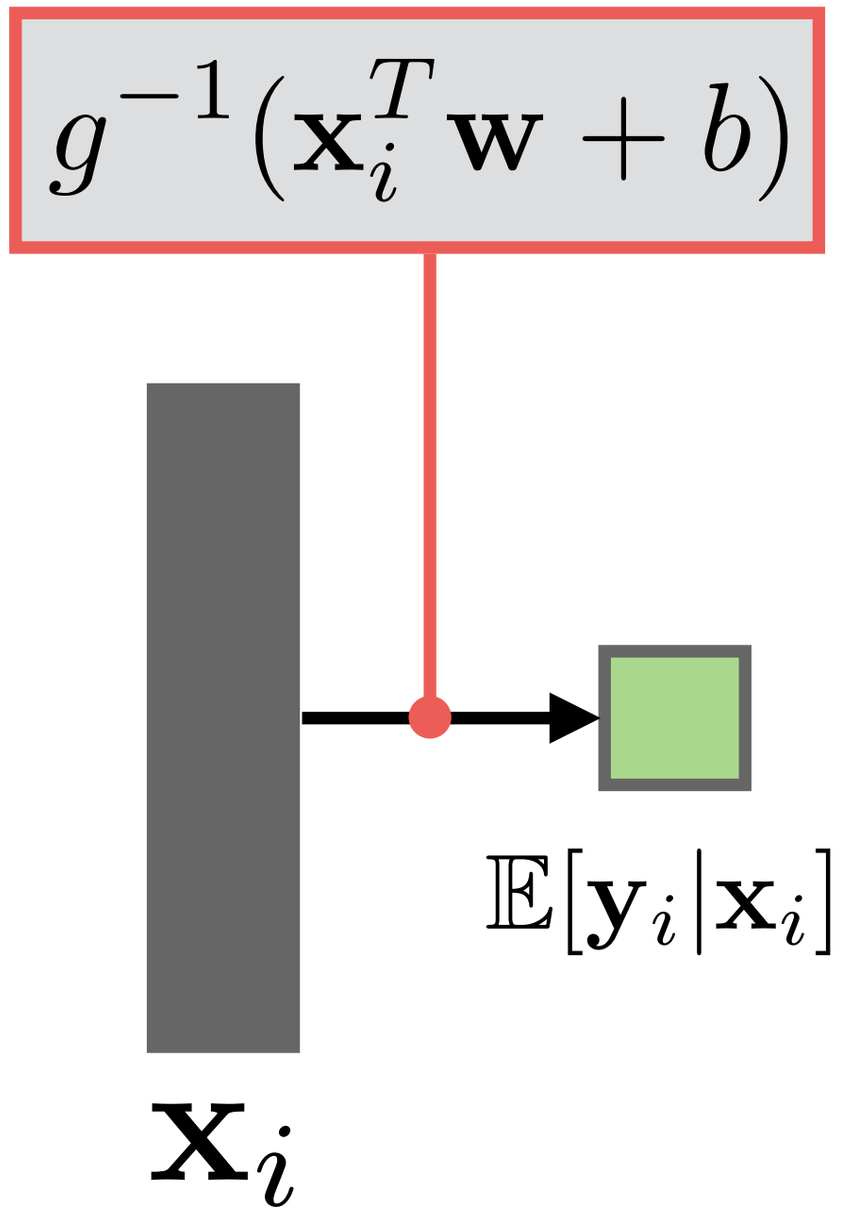
$$g^{-1}(\mathbf{x}_i^T \mathbf{w} + b) = \mathbb{E}[\mathbf{y}_i | \mathbf{x}_i]$$

Example: Logistic Regression

$$\mathbf{y}_i \in \{0, 1\} \qquad g^{-1}(z) = \frac{1}{1 + e^{-z}}$$

THE LOGISTIC FUNCTION

Generalized Linear Models (GLMs)



Recursive GLMs

Define a latent feature via a GLM:

$$g_1^{-1}(\mathbf{x}_i^T \mathbf{w}_1 + b_1) = h_i$$

[Mohamed, 2015]

Recursive GLMs

Define a latent feature via a GLM:

$$g_1^{-1}(\mathbf{x}_i^T \mathbf{w}_1 + b_1) = h_i$$

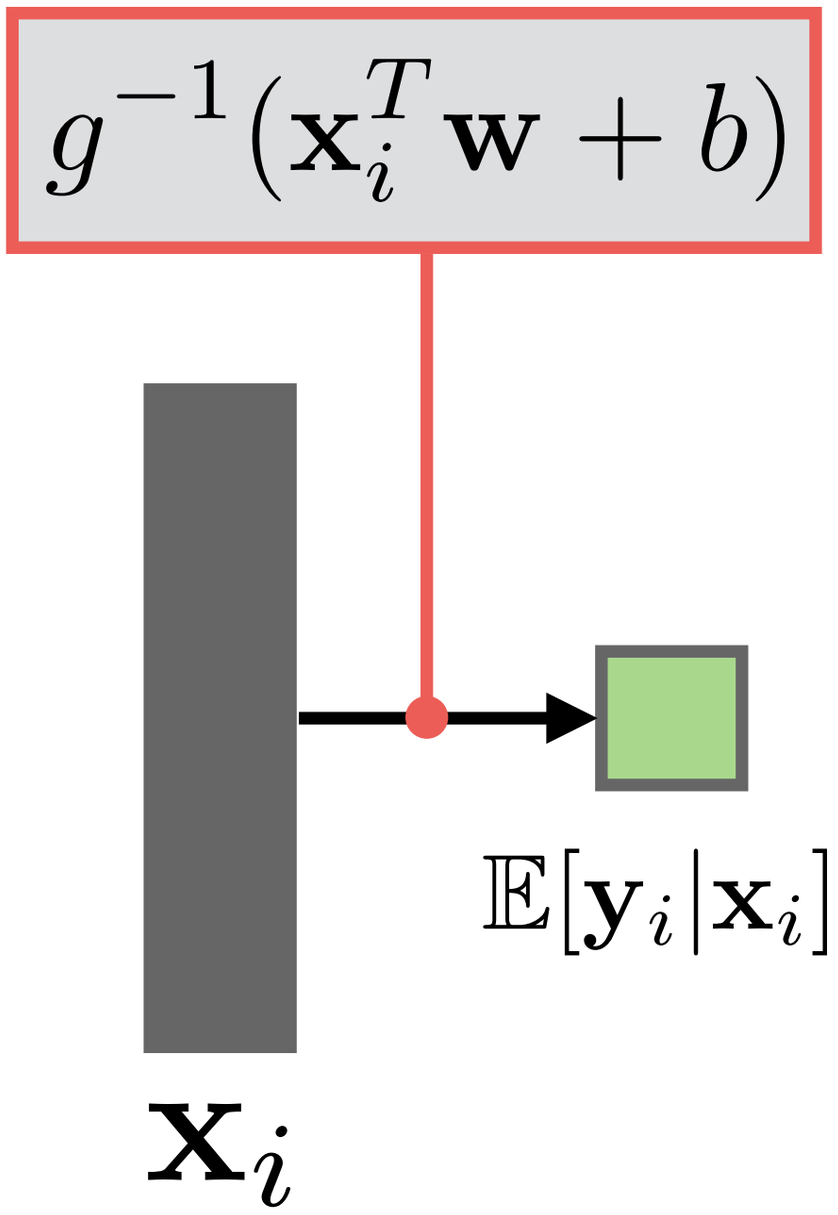
Define GLM on latent feature:

$$g_2^{-1}(h_i w_2 + b_2) = \mathbb{E}[\mathbf{y}_i | \mathbf{x}_i]$$

$$w_2 \in \mathbb{R}, b_2 \in \mathbb{R}$$

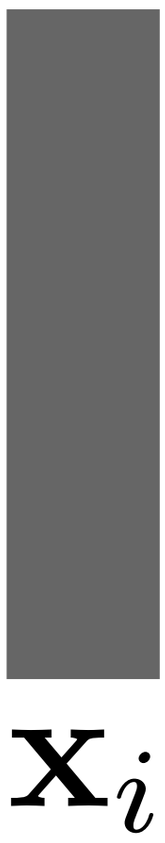
[Mohamed, 2015]

Recursive GLMs



[Mohamed, 2015]

Recursive GLMs

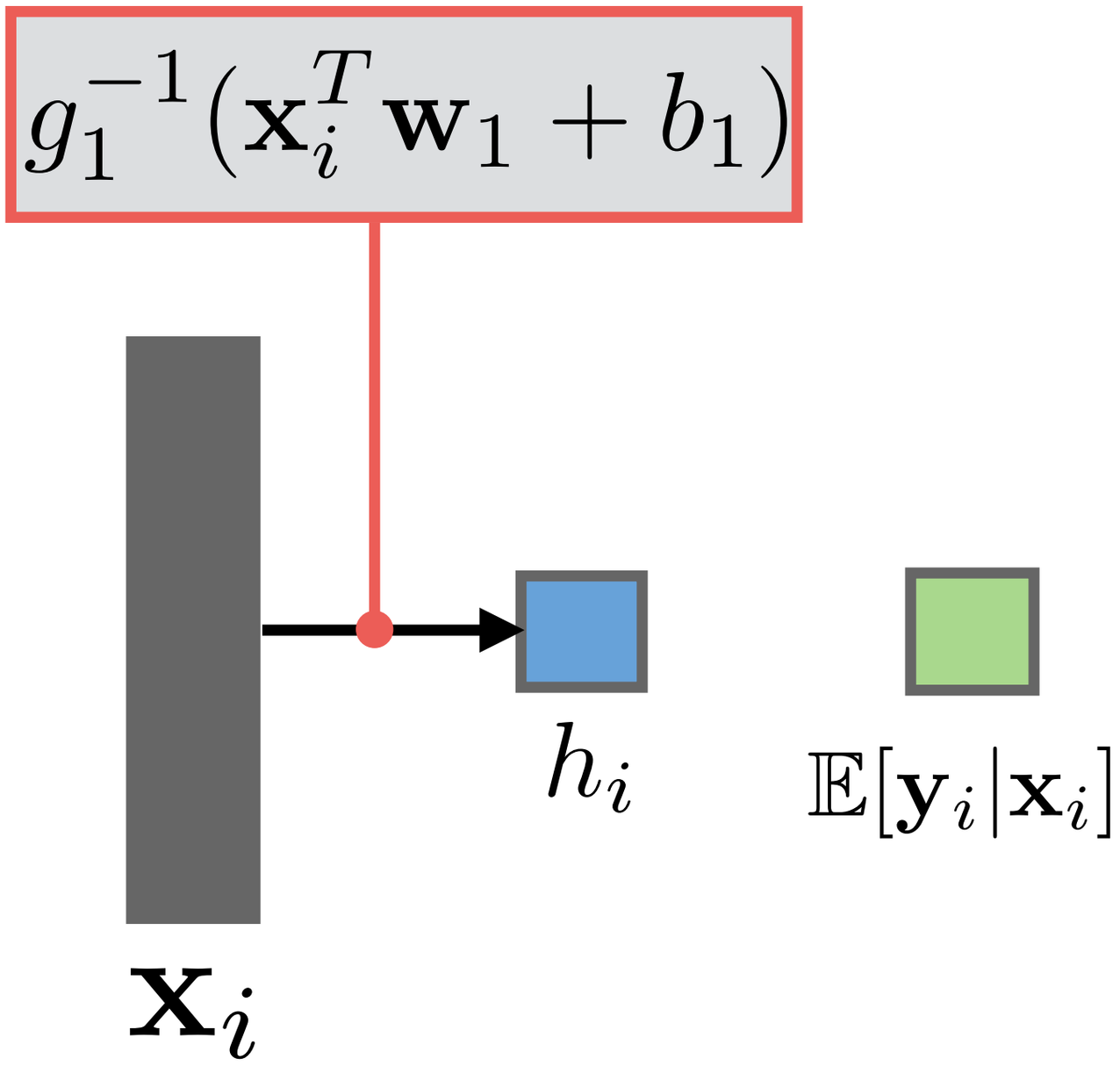


$\mathbf{x}_i$

$\mathbb{E}[\mathbf{y}_i|\mathbf{x}_i]$

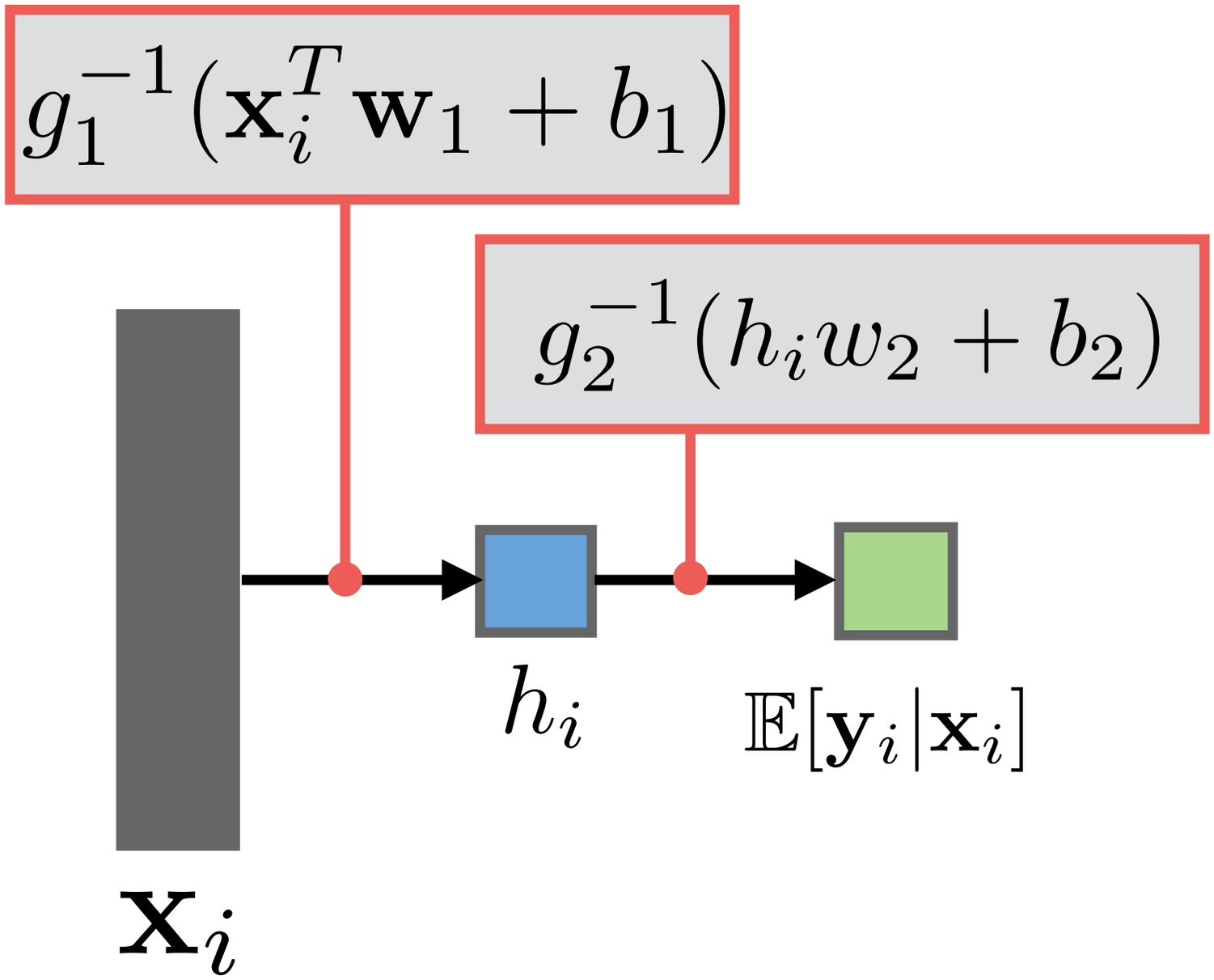
[Mohamed, 2015]

Recursive GLMs



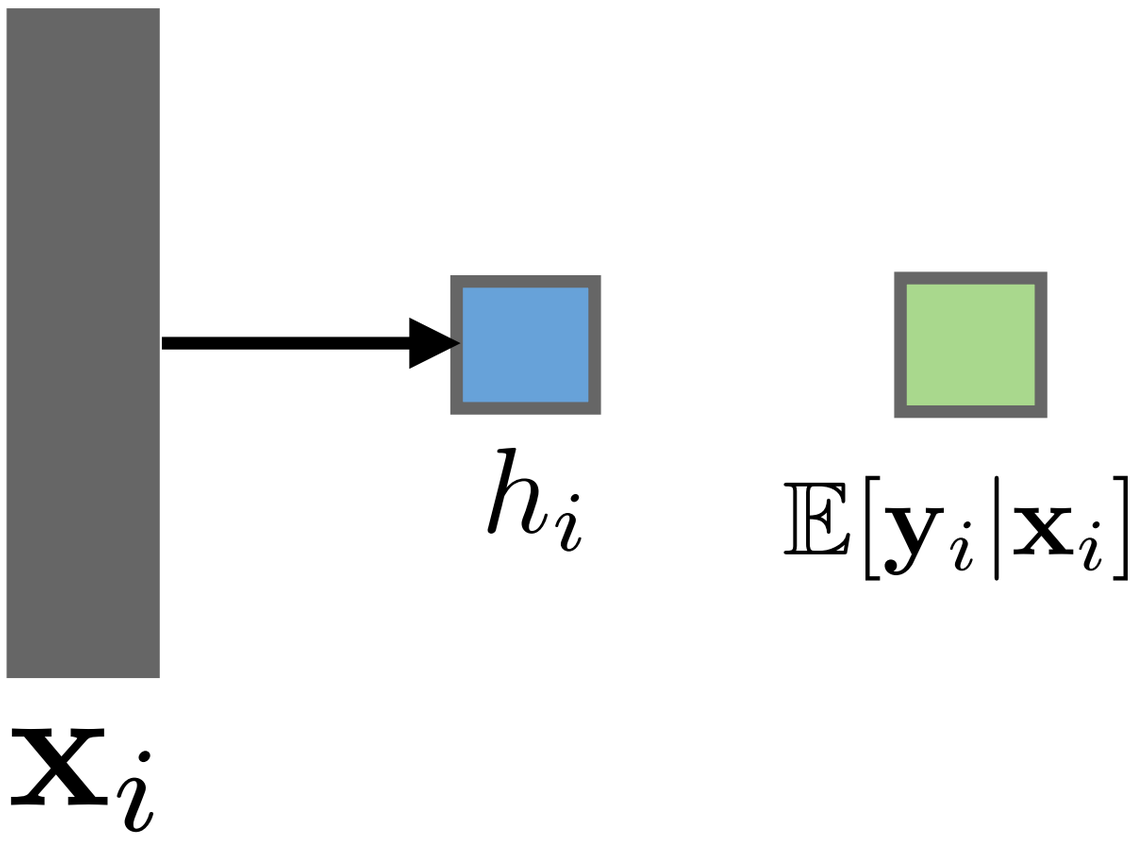
[Mohamed, 2015]

Recursive GLMs



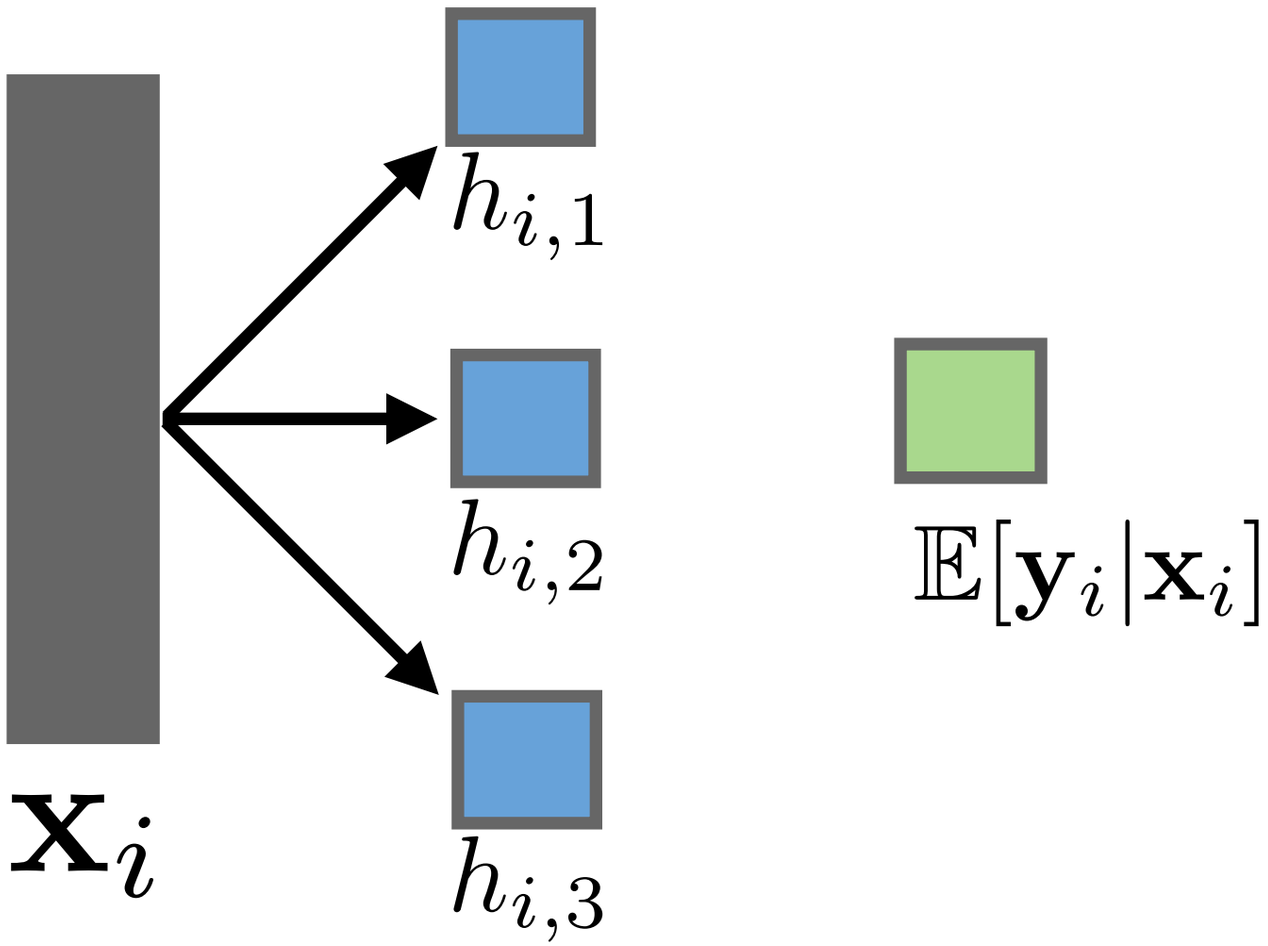
[Mohamed, 2015]

Building Neural Nets from Recursive GLMs



[Mohamed, 2015]

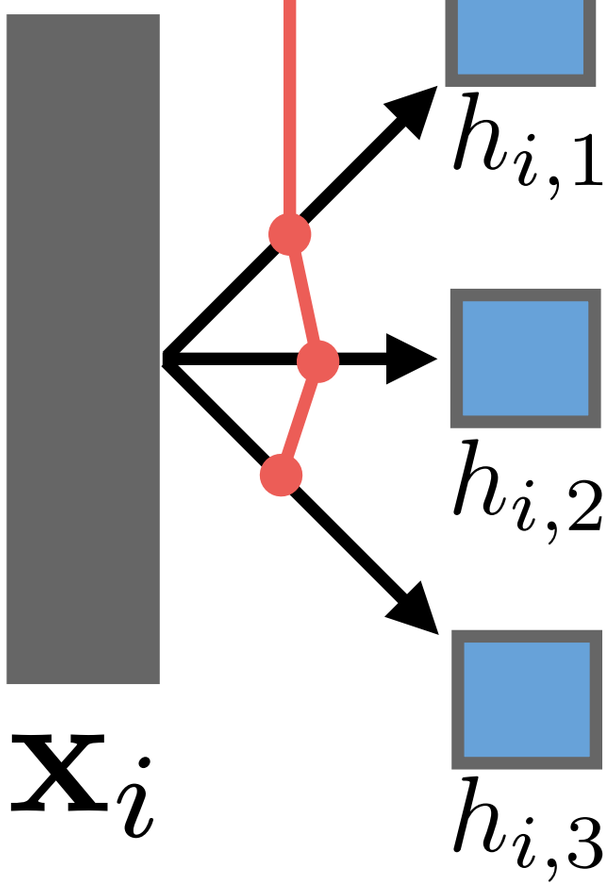
Building Neural Nets from Recursive GLMs



[Mohamed, 2015]

Building Neural Nets from Recursive GLMs

$$h_{i,j} = g_{1,j}^{-1}(\mathbf{x}_i^T \mathbf{w}_{1,j} + b_j)$$



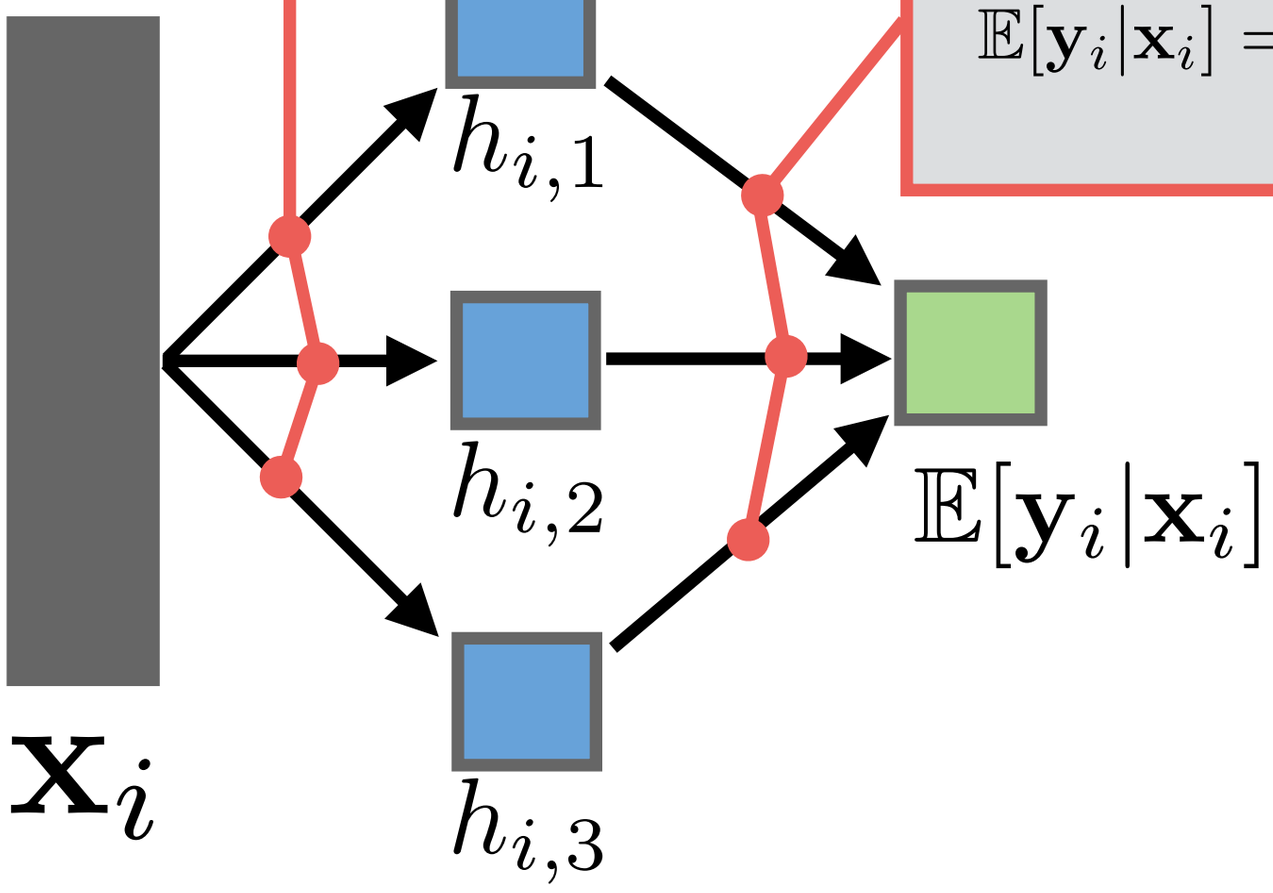
$\mathbb{E}[\mathbf{y}_i|\mathbf{x}_i]$

[Mohamed, 2015]

Building Neural Nets from Recursive GLMs

$$h_{i,j} = g_{1,j}^{-1}(\mathbf{x}_i^T \mathbf{w}_{1,j} + b_j)$$

$$\mathbb{E}[\mathbf{y}_i|\mathbf{x}_i] = g_2^{-1}\left(\sum_{j=1}^{H} h_{i,j} w_{2,j} + b_2\right)$$



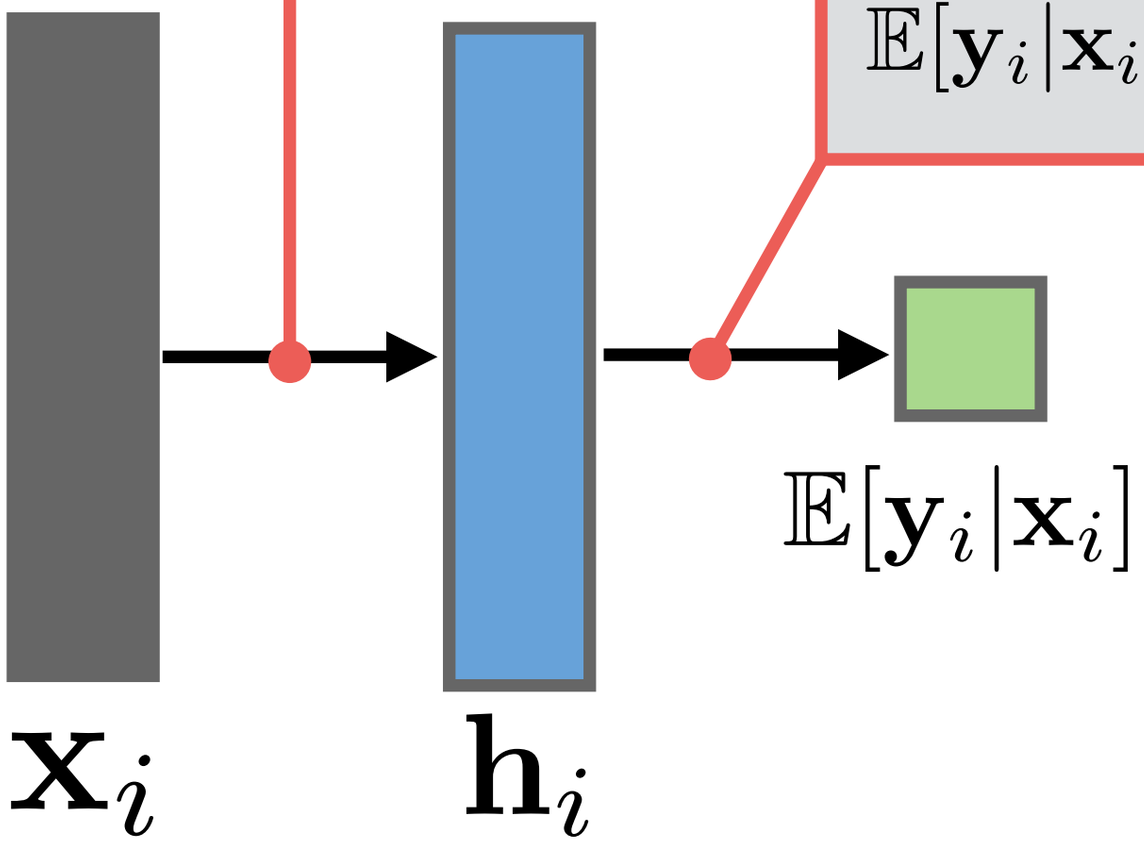
[Mohamed, 2015]

Building Neural Nets from Recursive GLMs

$\mathbf{h}_i \in \mathbb{R}^H, \mathbf{W}_1 \in \mathbb{R}^{d \times H}, \mathbf{b}_1 \in \mathbb{R}^H$

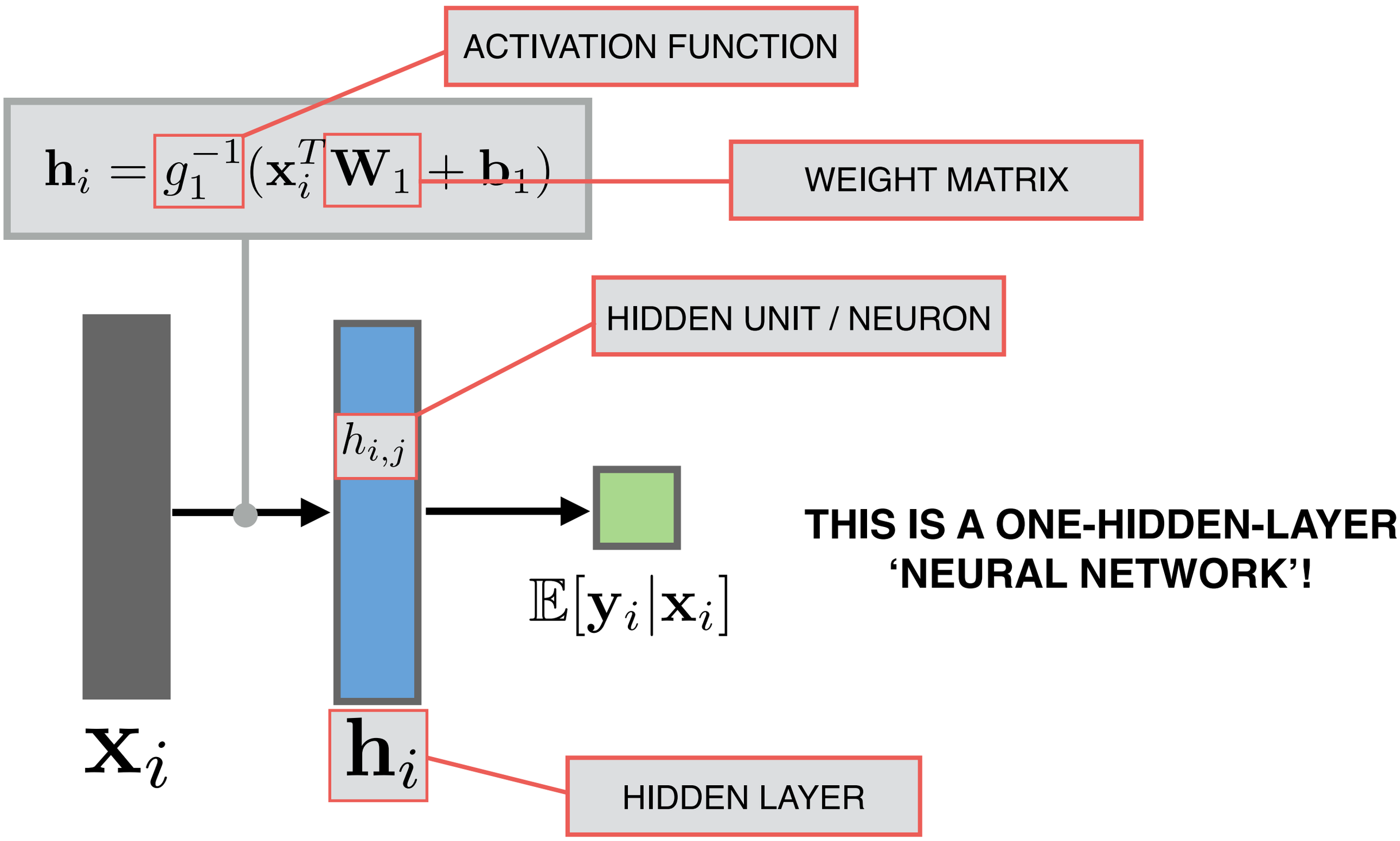
$$\mathbf{h}_i = g_1^{-1}(\mathbf{x}_i^T \mathbf{W}_1 + \mathbf{b}_1)$$

$$\mathbb{E}[\mathbf{y}_i|\mathbf{x}_i] = g_2^{-1}(\mathbf{h}_i \mathbf{w}_2 + b_2)$$



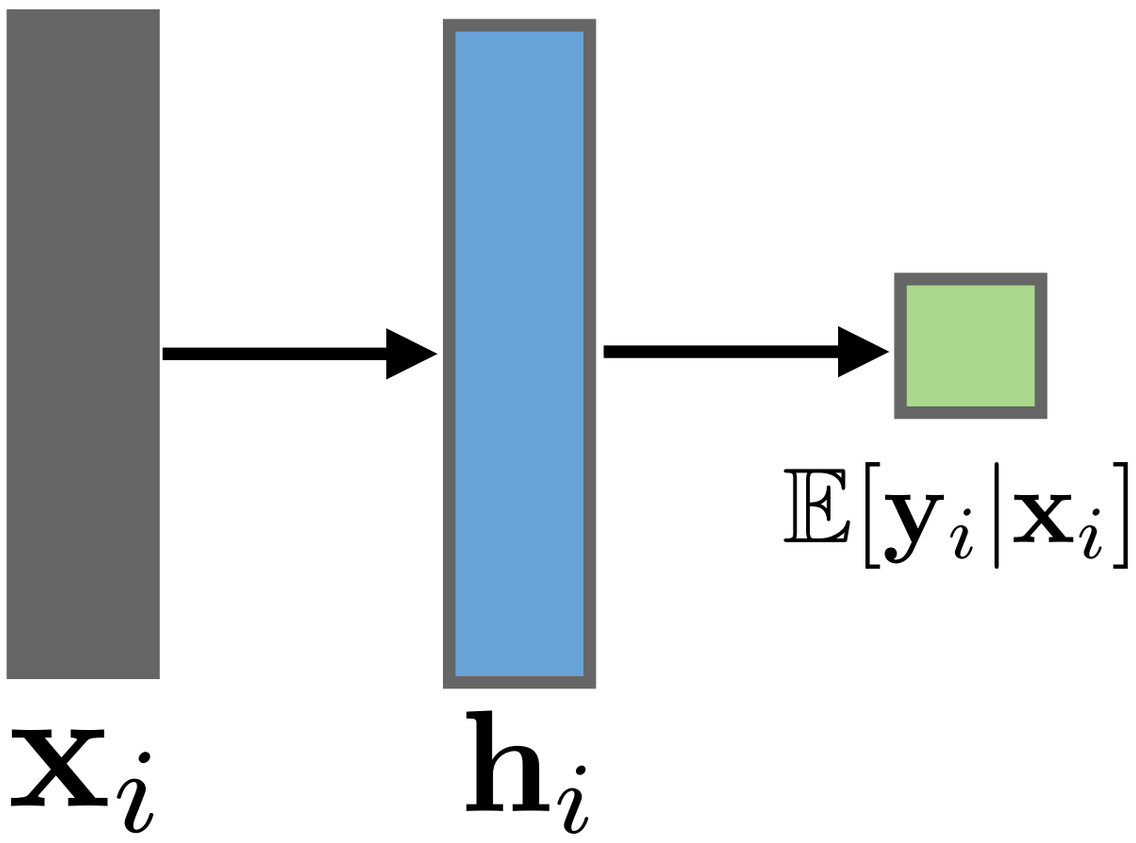
THIS IS A ONE-HIDDEN-LAYER ‘NEURAL NETWORK’!

Building Neural Nets from Recursive GLMs

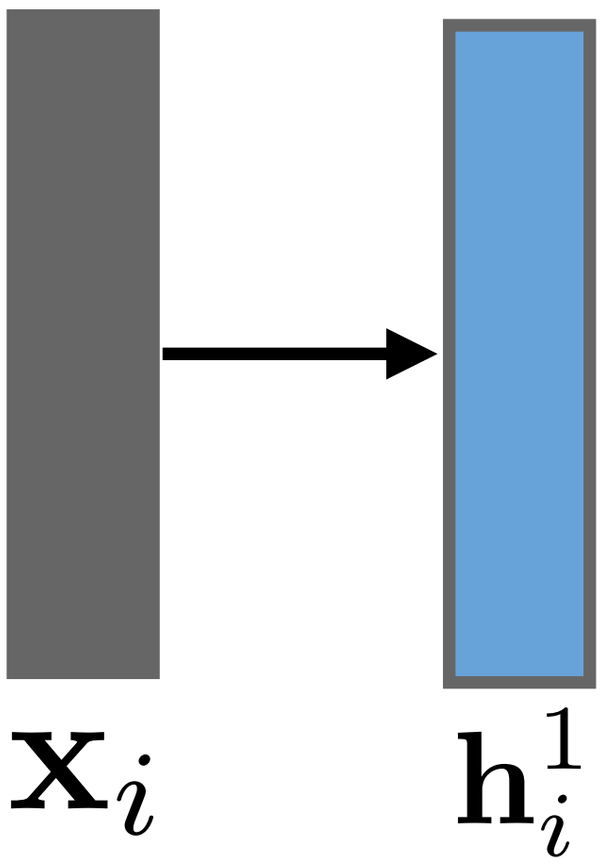


THIS IS A ONE-HIDDEN-LAYER ‘NEURAL NETWORK’!

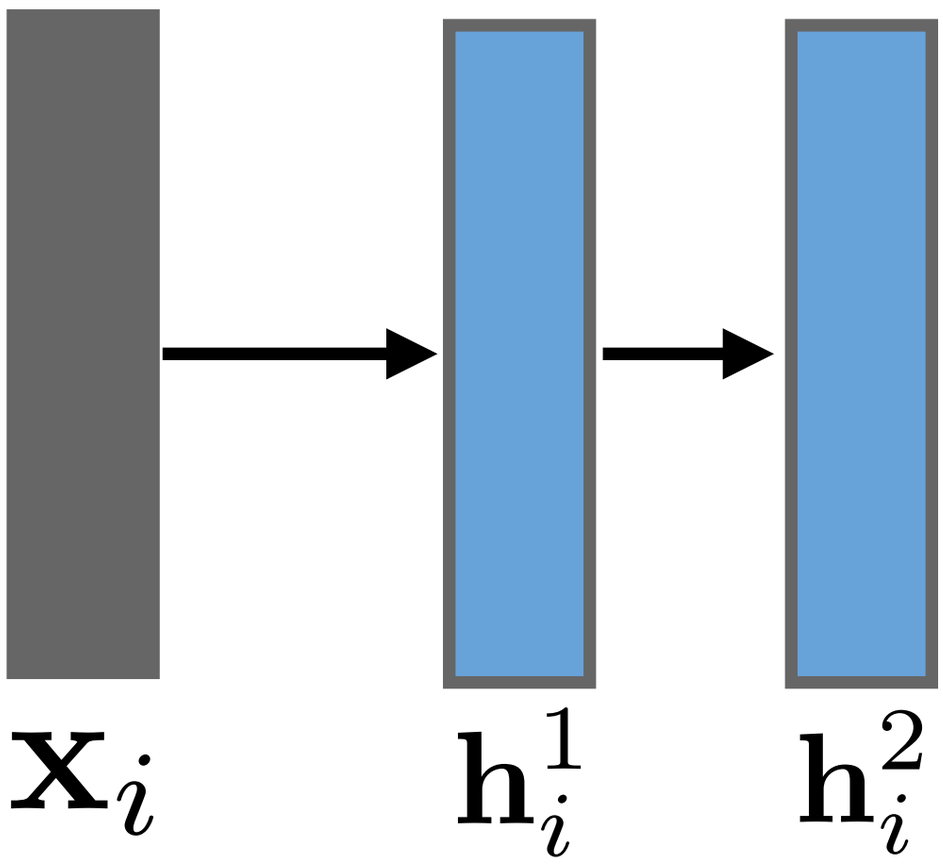
Deep Neural Networks



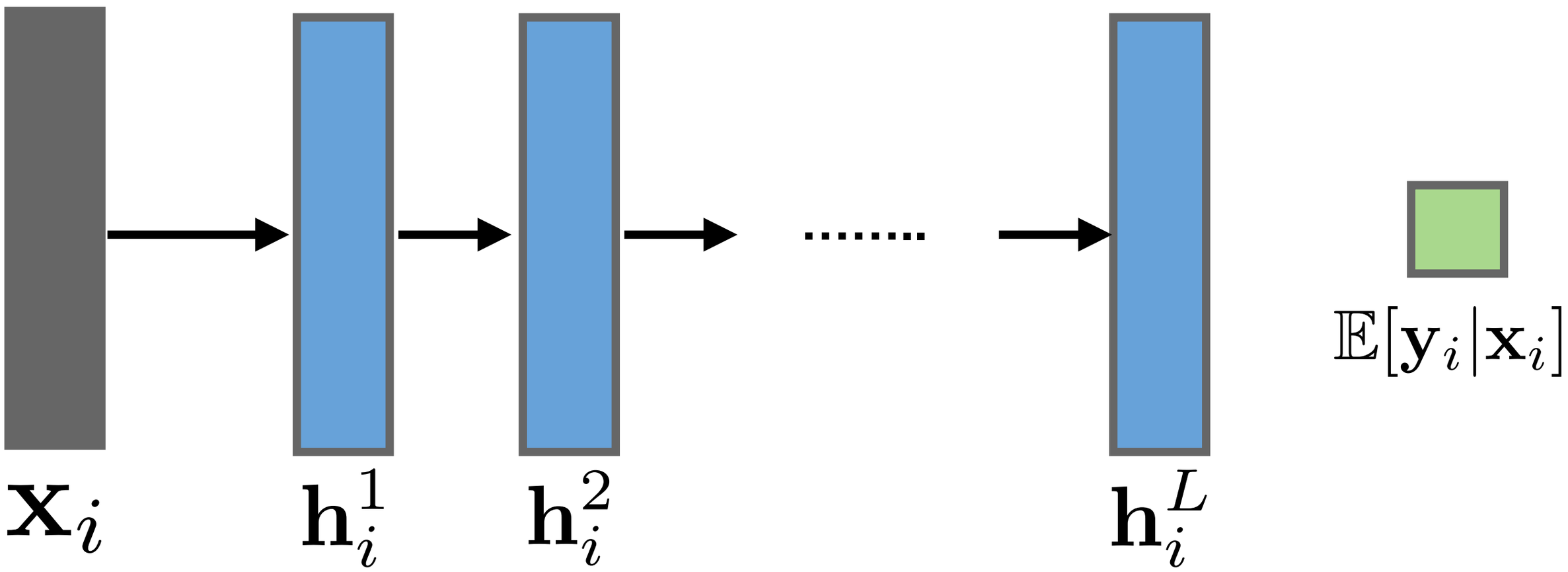
Deep Neural Networks



Deep Neural Networks



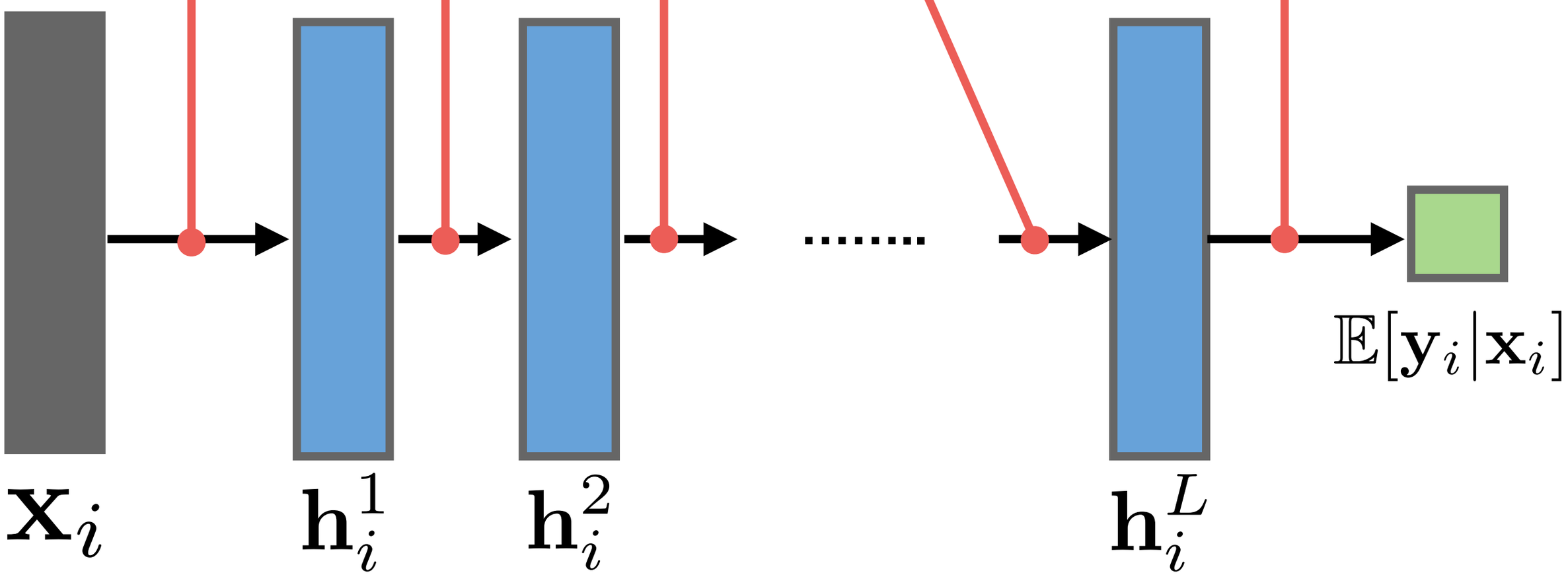
Deep Neural Networks



Deep Neural Networks

$$\mathbf{h}_i^l = f_l(\mathbf{h}_i^{l-1}\mathbf{W}_l + \mathbf{b}_l), \quad \mathbf{h}_i^0 = \mathbf{x}_i$$

$$\mathbb{E}[\mathbf{y}_i|\mathbf{x}_i] = g^{-1}(\mathbf{h}_i^L\mathbf{w}_o + b_o)$$



Deep Neural Networks

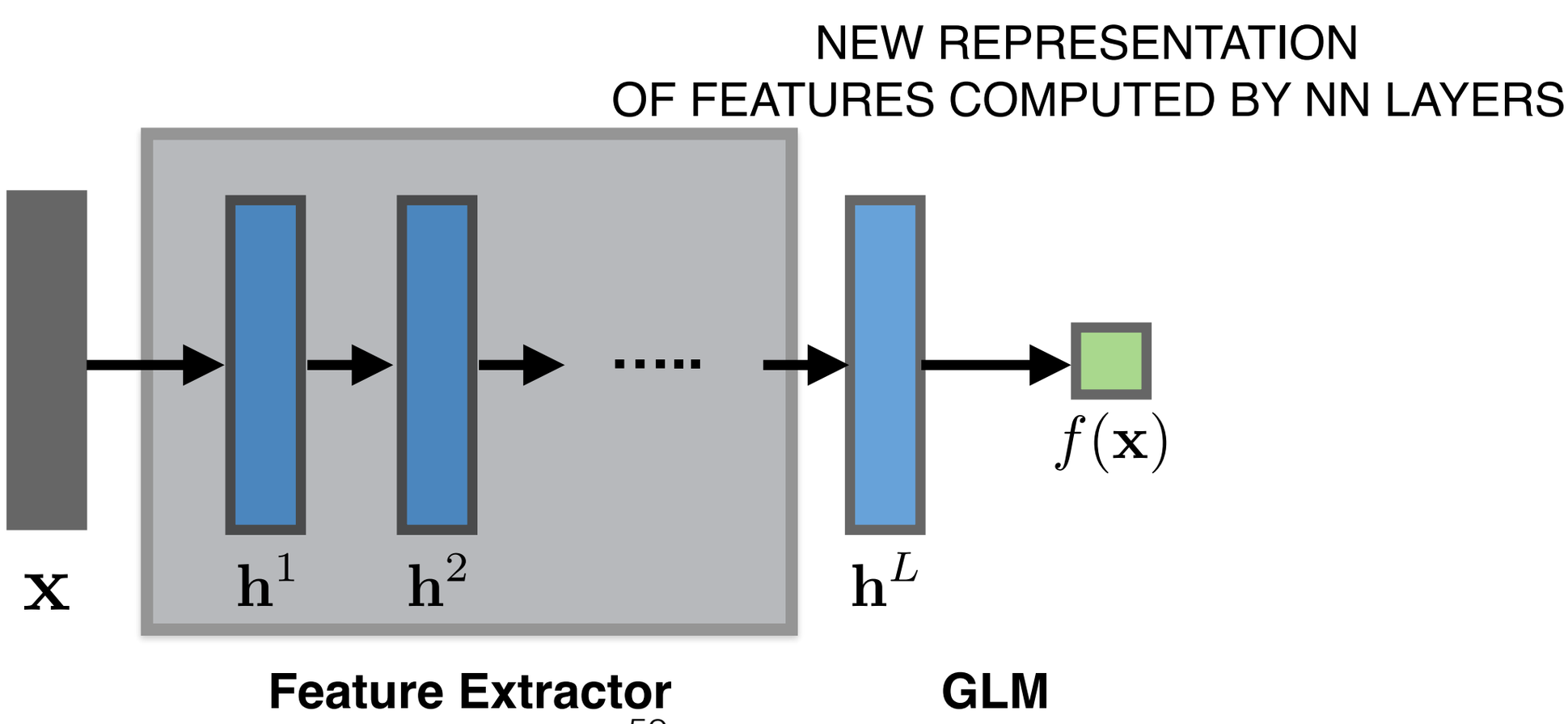
Adaptive Basis Function Regression: Neural networks are just performing regression with adaptive basis functions.

$$\mathbb{E}[\mathbf{y}_i|\mathbf{x}_i] = g^{-1}(\mathbf{h}_i^L \mathbf{w}_o + b_o) = g^{-1}(\mathbf{h}(\mathbf{x}_i)\mathbf{w}_o + b_o)$$

Deep Neural Networks

Adaptive Basis Function Regression: Neural networks are just performing regression with adaptive basis functions.

$$\mathbb{E}[\mathbf{y}_i|\mathbf{x}_i] = g^{-1}(\mathbf{h}_i^L \mathbf{w}_o + b_o) = g^{-1}(\mathbf{h}(\mathbf{x}_i)\mathbf{w}_o + b_o)$$



Gradient-Based Optimization

Model Fitting

Parameter Estimation: All parameters are estimated via gradient ascent on the log likelihood.

LOG LIKELIHOOD

$$\mathcal{L}(\mathbf{X}, \mathbf{Y}, \{\mathbf{W}_l\}_1^{L+1}, \{\mathbf{b}_l\}_1^{L+1}) = \sum_{i=1}^{N} \log p(\mathbf{y}_i | \mathbf{x}_i, \{\mathbf{W}_l\}_1^{L+1}, \{\mathbf{b}_l\}_1^{L+1})$$

Model Fitting

Parameter Estimation: All parameters are estimated via gradient ascent on the log likelihood.

LOG LIKELIHOOD

$$\mathcal{L}(\mathbf{X}, \mathbf{Y}, \{\mathbf{W}_l\}_1^{L+1}, \{\mathbf{b}_l\}_1^{L+1}) = \sum_{i=1}^{N} \log p(\mathbf{y}_i | \mathbf{x}_i, \{\mathbf{W}_l\}_1^{L+1}, \{\mathbf{b}_l\}_1^{L+1})$$

PARAMETER UPDATE

$$\mathbf{W}_l^{t+1} = \mathbf{W}_l^t + \alpha \, \nabla_{\mathbf{W}_l} \mathcal{L}(\mathbf{X}, \mathbf{Y}, \{\mathbf{W}_l\}_1^{L+1}, \{\mathbf{b}_l\}_1^{L+1})$$

Model Fitting

Parameter Estimation: All parameters are estimated via gradient ascent on the log likelihood.

LOG LIKELIHOOD

$$\mathcal{L}(\mathbf{X}, \mathbf{Y}, \{\mathbf{W}_l\}_1^{L+1}, \{\mathbf{b}_l\}_1^{L+1}) = \sum_{i=1}^{N} \log p(\mathbf{y}_i | \mathbf{x}_i, \{\mathbf{W}_l\}_1^{L+1}, \{\mathbf{b}_l\}_1^{L+1})$$

PARAMETER UPDATE

$$\mathbf{W}_l^{t+1} = \mathbf{W}_l^t + \alpha \nabla_{\mathbf{W}_l} \mathcal{L}(\mathbf{X}, \mathbf{Y}, \{\mathbf{W}_l\}_1^{L+1}, \{\mathbf{b}_l\}_1^{L+1})$$

learning rate / step size.

Model Fitting

Parameter Estimation: All parameters are estimated via gradient ascent on the log likelihood.

LOG LIKELIHOOD

$$\mathcal{L}(\mathbf{X}, \mathbf{Y}, \{\mathbf{W}_l\}_1^{L+1}, \{\mathbf{b}_l\}_1^{L+1}) = \sum_{i=1}^{N} \log p(\mathbf{y}_i | \mathbf{x}_i, \{\mathbf{W}_l\}_1^{L+1}, \{\mathbf{b}_l\}_1^{L+1})$$

PARAMETER UPDATE

$$\mathbf{W}_l^{t+1} = \mathbf{W}_l^t + \alpha \nabla_{\mathbf{W}_l} \mathcal{L}(\mathbf{X}, \mathbf{Y}, \{\mathbf{W}_l\}_1^{L+1}, \{\mathbf{b}_l\}_1^{L+1})$$

learning rate / step size.

Usually estimated with a subsample (i.e. 'stochastic gradient')

$$\nabla_{\mathbf{W}_l} \mathcal{L}(\mathbf{X}, \mathbf{Y}, \{\mathbf{W}_l\}_1^{L+1}, \{\mathbf{b}_l\}_1^{L+1}) \approx \nabla_{\mathbf{W}_l} \sum_{k=1}^{K} \log p(\mathbf{y}_k | \mathbf{x}_k, \{\mathbf{W}_l\}_1^{L+1}, \{\mathbf{b}_l\}_1^{L+1})$$

for $K << N$

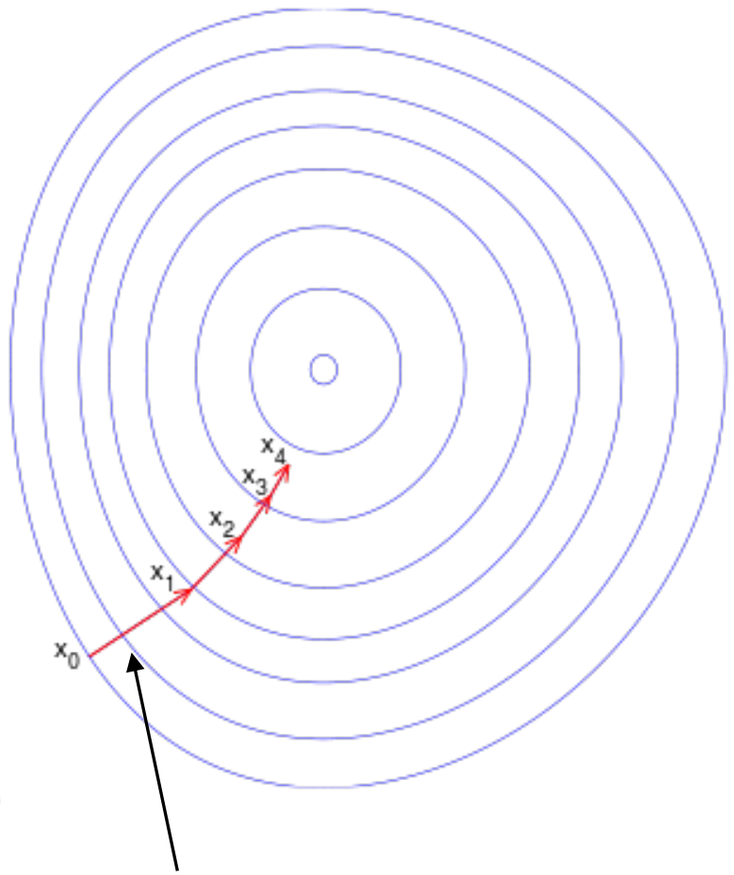
Model Fitting

PARAMETER UPDATE

$$\mathbf{W}_l^{t+1} = \mathbf{W}_l^t + \alpha \, \nabla_{\mathbf{W}_l} \mathcal{L}(\mathbf{X}, \mathbf{Y}, \{\mathbf{W}_l\}_1^{L+1}, \{\mathbf{b}_l\}_1^{L+1})$$

$$\nabla_{\mathbf{W}_l} \mathcal{L}(\mathbf{X}, \mathbf{Y}, \{\mathbf{W}_l\}_1^{L+1}, \{\mathbf{b}_l\}_1^{L+1}) \approx \nabla_{\mathbf{W}_l} \sum_{k=1}^{K} \log p(\mathbf{y}_k | \mathbf{x}_k, \{\mathbf{W}_l\}_1^{L+1}, \{\mathbf{b}_l\}_1^{L+1})$$

for $K << N$



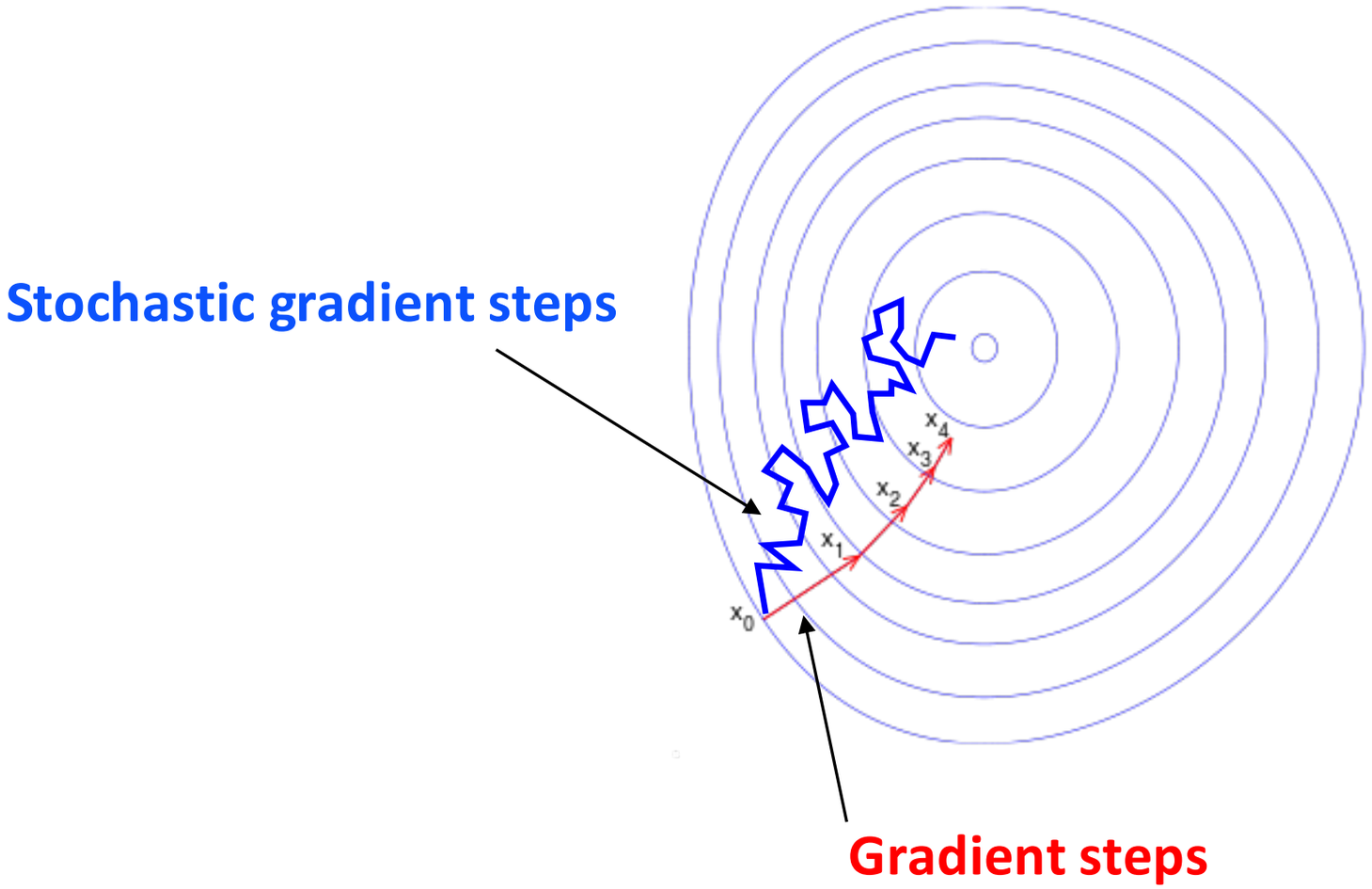
Model Fitting

PARAMETER UPDATE

$$\mathbf{W}_l^{t+1} = \mathbf{W}_l^t + \alpha \, \nabla_{\mathbf{W}_l} \mathcal{L}(\mathbf{X}, \mathbf{Y}, \{\mathbf{W}_l\}_1^{L+1}, \{\mathbf{b}_l\}_1^{L+1})$$

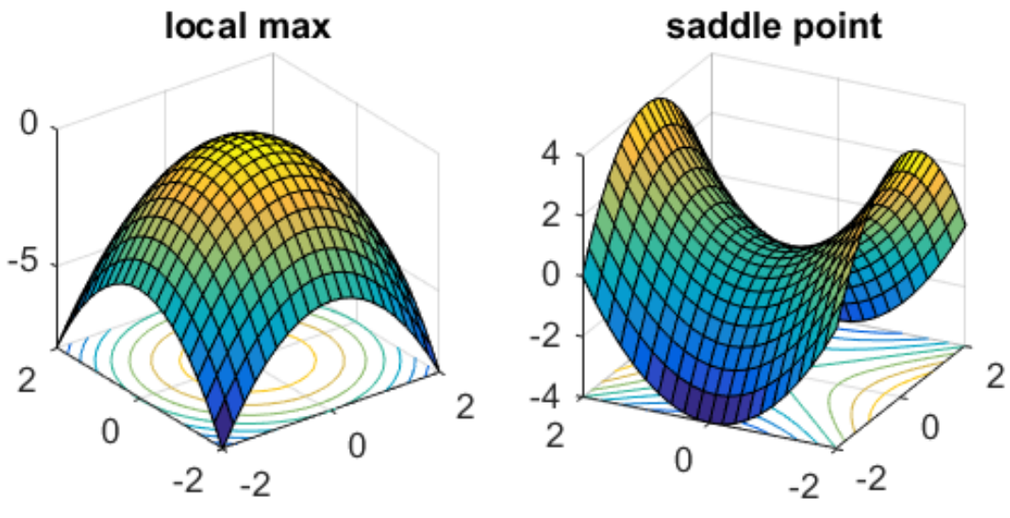
$$\nabla_{\mathbf{W}_l} \mathcal{L}(\mathbf{X}, \mathbf{Y}, \{\mathbf{W}_l\}_1^{L+1}, \{\mathbf{b}_l\}_1^{L+1}) \approx \nabla_{\mathbf{W}_l} \sum_{k=1}^{K} \log p(\mathbf{y}_k | \mathbf{x}_k, \{\mathbf{W}_l\}_1^{L+1}, \{\mathbf{b}_l\}_1^{L+1})$$

for $K << N$



Optimization Landscape

- Previously, optimization of deep NNs was thought to be hopelessly non-convex. However, many of the critical points are not maxima but rather ***saddle points***. [Dauphin et al., 2014]

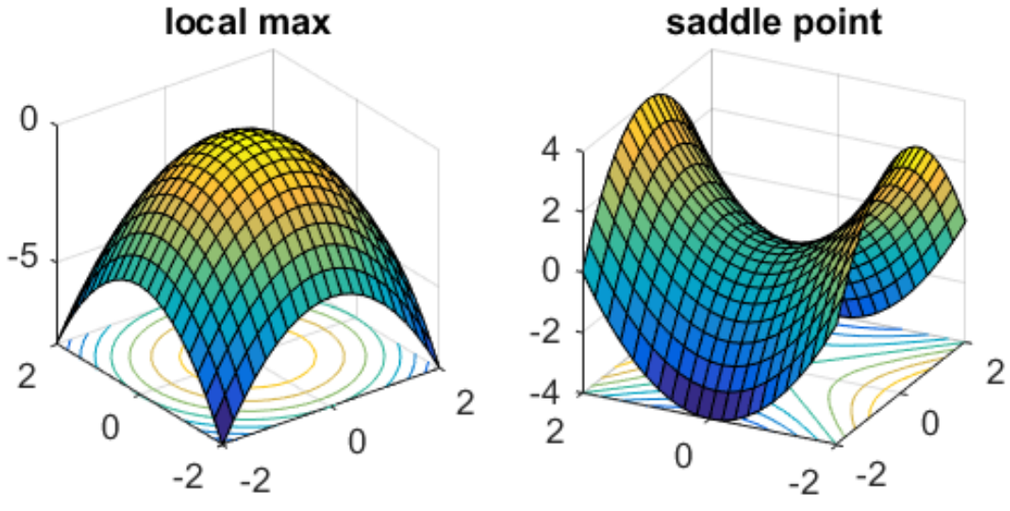


http://www.offconvex.org/2016/03/22/

Optimization Landscape

- Previously, optimization of deep NNs was thought to be hopelessly non-convex. However, many of the critical points are not maxima but rather ***saddle points***. [Dauphin et al., 2014]

- **Intuition:** It's hard to build a high-dimensional max / min because the surface must be going in the same direction in all dimensions.



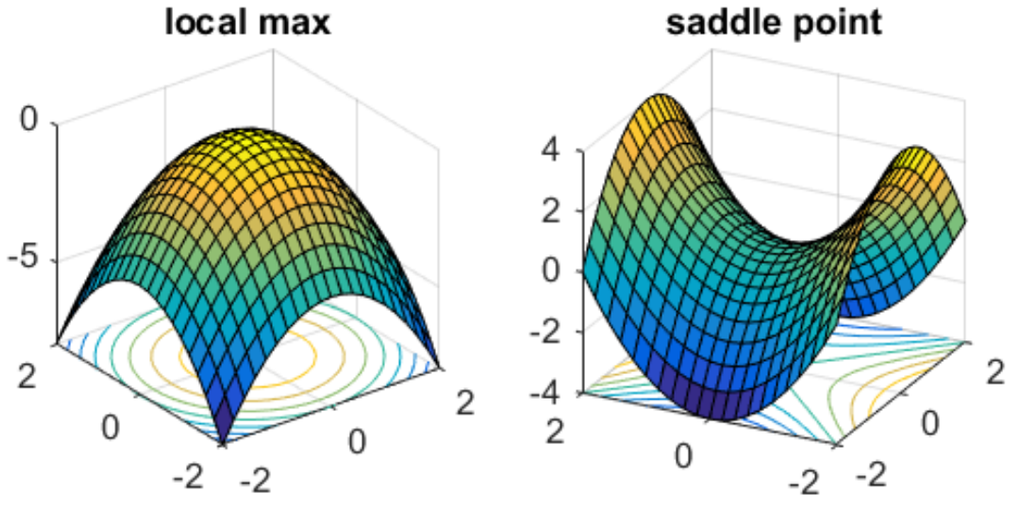
http://www.offconvex.org/2016/03/22/

Optimization Landscape

- Previously, optimization of deep NNs was thought to be hopelessly non-convex. However, many of the critical points are not maxima but rather ***saddle points***. [Dauphin et al., 2014]

- **Intuition:** It's hard to build a high-dimensional max / min because the surface must be going in the same direction in all dimensions.

- **Stochastic Gradient:** [Hardt et al., 2016] (+ others) suggest the noise from stochastic gradient regularizes, improving model generalization.



http://www.offconvex.org/2016/03/22/

Model Fitting

- **Backpropagation**: The *backpropagation* algorithm [Rumelhart et al., 1986] is just an efficient way to compute the chain rule through the network.

$$\frac{\partial}{\partial\ \mathbf{W}_l}\mathcal{L} = \frac{\partial}{\partial\ \mathbf{h}_i^L}\log p(\mathbf{y}_i|\mathbf{h}_i^L, \mathbf{w}_o, b_o)\ \frac{\partial\ \mathbf{h}_i^L}{\partial\ \mathbf{h}_i^{L-1}}\cdots\frac{\partial\ \mathbf{h}_i^{l+1}}{\partial\ \mathbf{h}_i^l}\ \frac{\partial}{\partial\ \mathbf{W}_l}f_l(\mathbf{h}_i^{l-1}\mathbf{W}_l + \mathbf{b}_l)$$

Model Fitting

- **Backpropagation**: The *backpropagation* algorithm [Rumelhart et al., 1986] is just an efficient way to compute the chain rule through the network.

$$\frac{\partial}{\partial \mathbf{W}_l}\mathcal{L} = \frac{\partial}{\partial \mathbf{h}_i^L}\log p(\mathbf{y}_i|\mathbf{h}_i^L, \mathbf{w}_o, b_o)\ \frac{\partial \mathbf{h}_i^L}{\partial \mathbf{h}_i^{L-1}} \cdots \frac{\partial \mathbf{h}_i^{l+1}}{\partial \mathbf{h}_i^l}\ \frac{\partial}{\partial \mathbf{W}_l} f_l(\mathbf{h}_i^{l-1}\mathbf{W}_l + \mathbf{b}_l)$$

- **Automatic Differentiation**: Various 'autodiff' software libraries can compute these gradients for you, just needing definition of the forward model.

torch	theano	TensorFlow	PYTORCH
RONAN COLLOBERT (2000)	UNIVERSITY OF MONTREAL (2011)	GOOGLE BRAIN (2009/2015)	FACEBOOK AI (2016)

Model Fitting

- **Backpropagation**: The *backpropagation* algorithm [Rumelhart et al., 1986] is just an efficient way to compute the chain rule through the network.

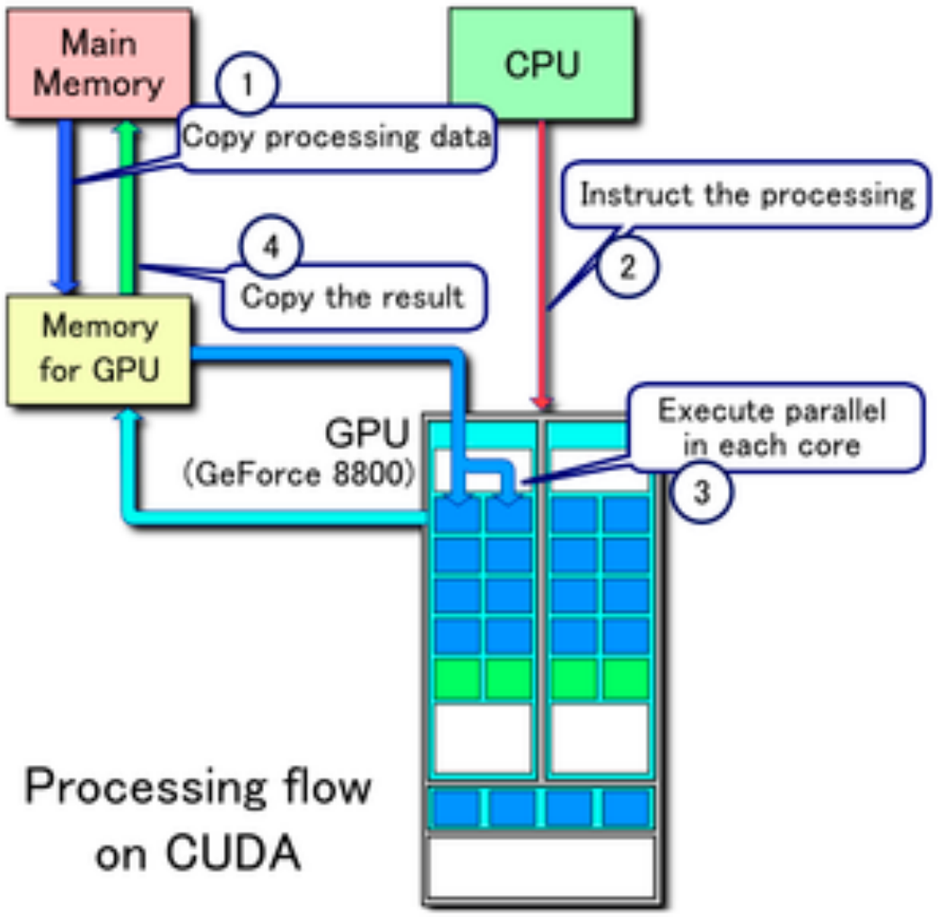
$$\frac{\partial}{\partial\ \mathbf{W}_l}\mathcal{L} = \frac{\partial}{\partial\ \mathbf{h}_i^L}\log p(\mathbf{y}_i|\mathbf{h}_i^L, \mathbf{w}_o, b_o)\ \frac{\partial\ \mathbf{h}_i^L}{\partial\ \mathbf{h}_i^{L-1}} \cdots \frac{\partial\ \mathbf{h}_i^{l+1}}{\partial\ \mathbf{h}_i^l}\ \frac{\partial}{\partial\ \mathbf{W}_l} f_l(\mathbf{h}_i^{l-1}\mathbf{W}_l + \mathbf{b}_l)$$

- **Automatic Differentiation**: Various 'autodiff' software libraries can compute these gradients for you, just needing definition of the forward model.

MOST POPULAR TODAY

torch	theano	TensorFlow	PYTORCH
RONAN COLLOBERT (2000)	UNIVERSITY OF MONTREAL (2011)	GOOGLE BRAIN (2009/2015)	FACEBOOK AI (2016)

Graphics Processing Unit (GPU)



- GPU cards have been essential to scaling deep learning models.
- Perform fast matrix multiplications via parallel, but low precision, computation

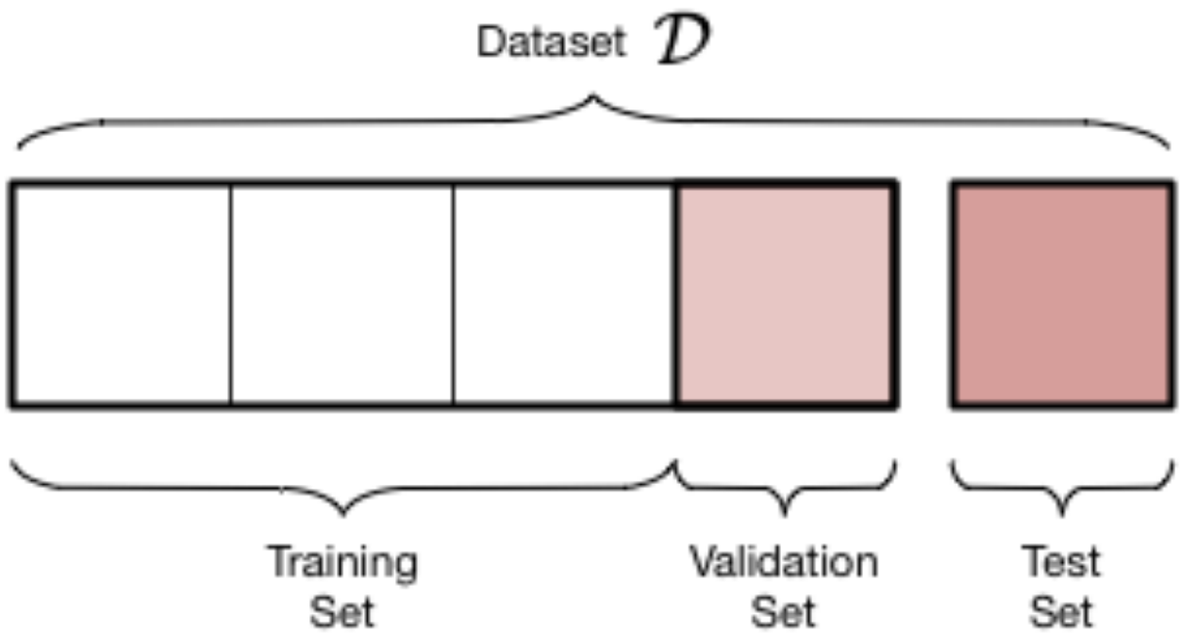
Hyperparameter Tuning

- [ ] Tuning the network architecture (e.g. number of hidden units, layers) and optimization parameters (e.g. step size) is needed for best performance.

Hyperparameter Tuning

- [ ] Tuning the network architecture (e.g. number of hidden units, layers) and optimization parameters (e.g. step size) is needed for best performance.

- [ ] Usually, a **random or grid search** over hyper parameters is performed, and **performance on a validation set** is used to make the final selection.

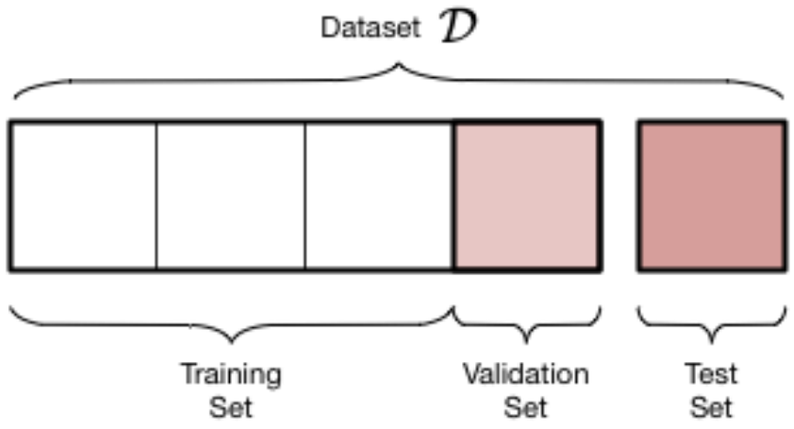


Hyperparameter Tuning

- Tuning the network architecture (e.g. number of hidden units, layers) and optimization parameters (e.g. step size) is needed for best performance.

- Usually, a **random or grid search** over hyper parameters is performed, and **performance on a validation set** is used to make the final selection.

- Improving the model selection process is an on-going area of research. Bayesian optimization, bandit algorithms, and reinforcement learning methods have been proposed.

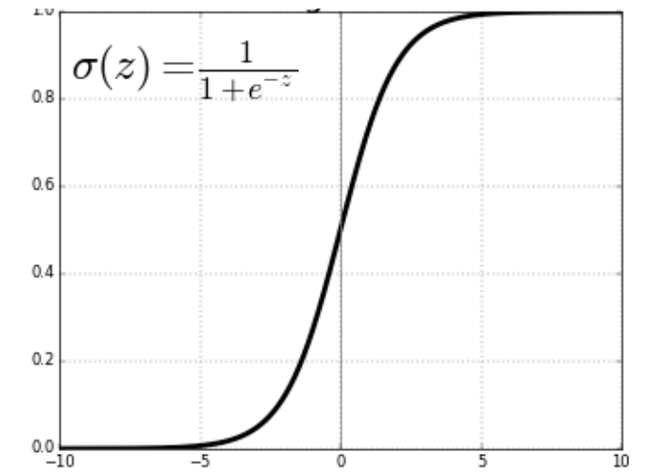


So What's Changed Since the 90's?

IN TERMS OF THE NN MODEL

Model Changes

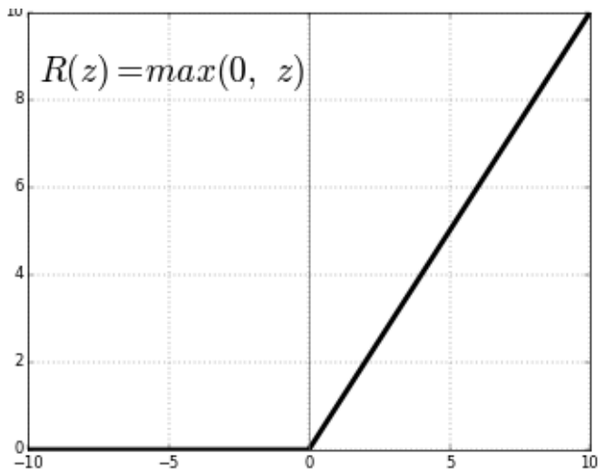
1. **ReLU Activations**: Scale-free non-linearities [Nair and Hinton, 2010].



LOGISTIC FUNCTION

OLD

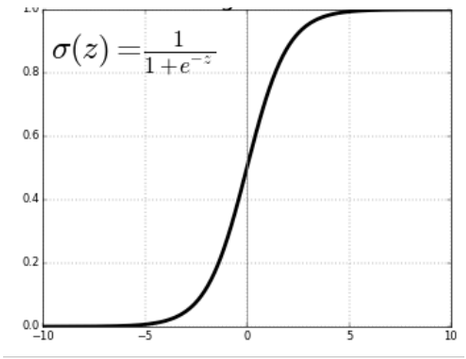
VS



RECTIFIED LINEAR UNIT (ReLU)

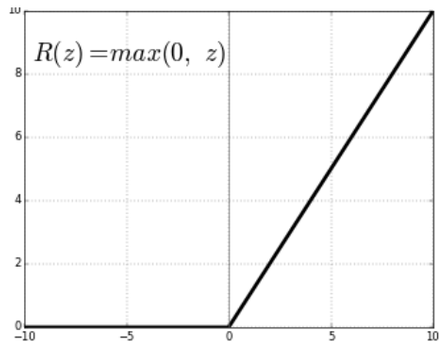
NEW

Model Changes



LOGISTIC FUNCTION

VS



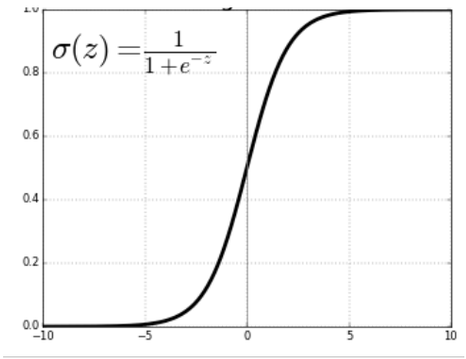
RECTIFIED LINEAR UNIT (ReLU)

CONNECTION

ReLUs are essentially the 'hockey stick' function used in piecewise linear spline regression:

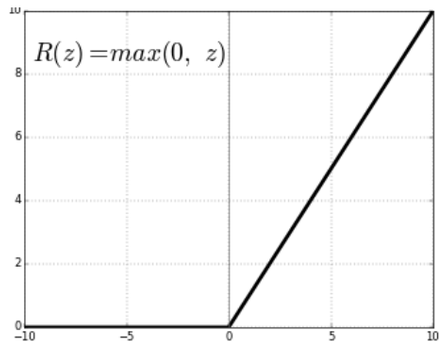
$$\gamma(\mathbf{x}; \boldsymbol{\beta}, \{\boldsymbol{\xi}\}_H) = \sum_{m=1}^{H} \beta_h (\mathbf{x} - \boldsymbol{\xi}_m)_+$$

Model Changes



LOGISTIC FUNCTION

VS



RECTIFIED LINEAR UNIT (ReLU)

CONNECTION

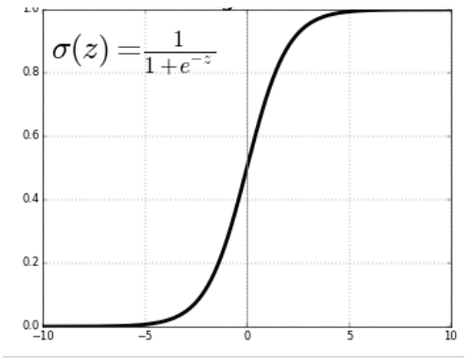
ReLUs are essentially the 'hockey stick' function used in piecewise linear spline regression:

$$\gamma(\mathbf{x}; \boldsymbol{\beta}, \{\boldsymbol{\xi}\}_H) = \sum_{m=1}^{H} \beta_h (\mathbf{x} - \boldsymbol{\xi}_m)_+$$

We can think of ReLU units in neural networks as having angular knots:

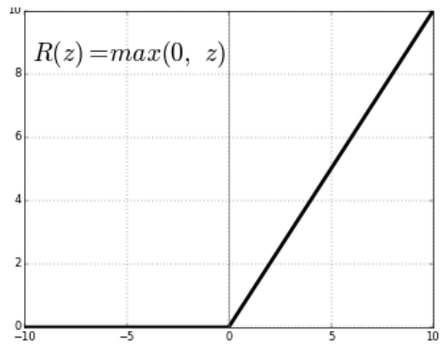
$$\mathbf{x} > \xi_m \quad \text{VS} \quad \cos(\mathbf{x}, \boldsymbol{\beta}_{1,m}) > \frac{\xi_m}{||\mathbf{x}||\ ||\boldsymbol{\beta}_{1,m}||}$$

Model Changes



LOGISTIC FUNCTION

VS



RECTIFIED LINEAR UNIT (ReLU)

CONNECTION

ReLUs are essentially the 'hockey stick' function used in piecewise linear spline regression:

$$\gamma(\mathbf{x}; \boldsymbol{\beta}, \{\boldsymbol{\xi}\}_H) = \sum_{m=1}^{H} \beta_h (\mathbf{x} - \boldsymbol{\xi}_m)_+$$

We can think of ReLU units in neural networks as having angular knots:

$$\mathbf{x} > \xi_m \quad \text{VS} \quad \cos(\mathbf{x}, \boldsymbol{\beta}_{1,m}) > \frac{\xi_m}{||\mathbf{x}||\ ||\boldsymbol{\beta}_{1,m}||}$$

BIAS VARIABLES = KNOTS

Model Changes

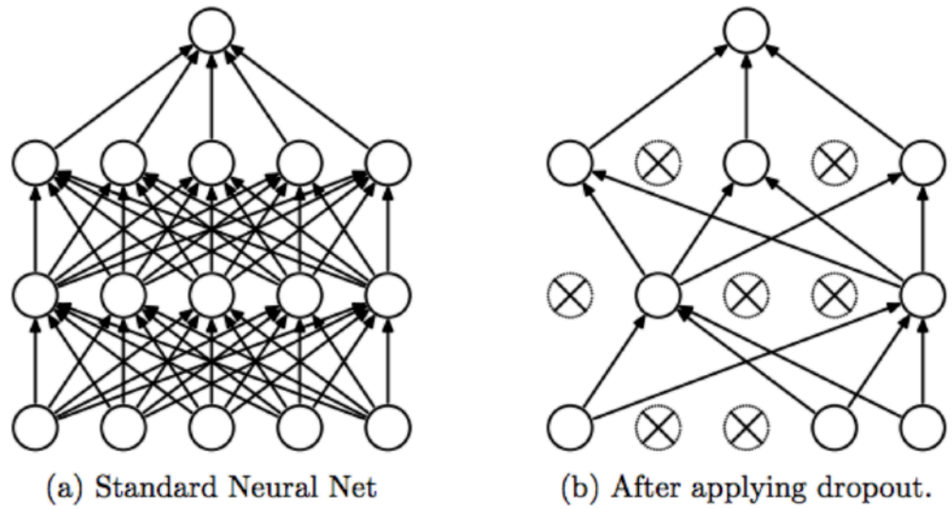
2. Better Regularization: Stochastic regularization strategies [Srivastava et al., 2014].

$$\sum_{l=1}^{L+1} ||\mathbf{W}_l||_2^2$$

RIDGE PENALTY

OLD

VS



(a) Standard Neural Net

(b) After applying dropout.

DROPOUT

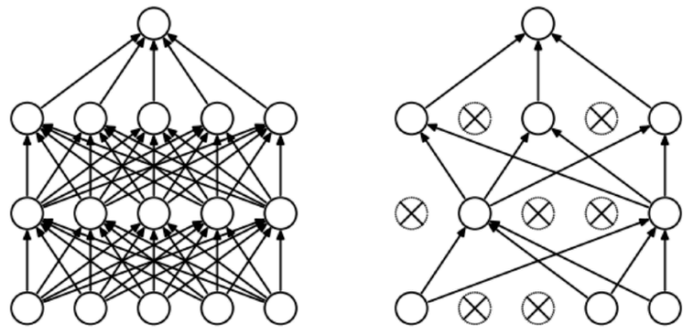
NEW

Model Changes

$$\sum_{l=1}^{L+1} ||\mathbf{W}_l||_2^2$$

RIDGE PENALTY

VS



(a) Standard Neural Net (b) After applying dropout.

DROPOUT

CONNECTION

Can think of dropout as spike and slab variable selection

[Kuo and Mallick, 1998]:

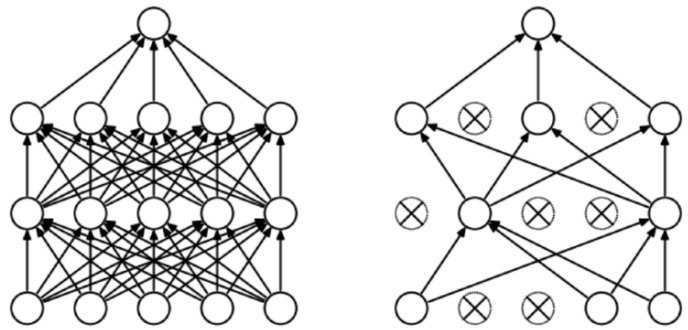
$$\mathbf{h}_i^l = f_l\left((\mathbf{h}_i^{l-1} \odot \boldsymbol{\xi})\mathbf{W}_l + \mathbf{b}_l\right), \boldsymbol{\xi} \sim \text{Bernoulli}(p)$$

Model Changes

$$\sum_{l=1}^{L+1} ||\mathbf{W}_l||_2^2$$

RIDGE PENALTY

vs



(a) Standard Neural Net (b) After applying dropout.

DROPOUT

CONNECTION

Can think of dropout as spike and slab variable selection

[Kuo and Mallick, 1998]:

$$\mathbf{h}_i^l = f_l\left((\mathbf{h}_i^{l-1} \odot \boldsymbol{\xi})\mathbf{W}_l + \mathbf{b}_l\right), \boldsymbol{\xi} \sim \text{Bernoulli}(p)$$

Noise Model $p(\lambda)$	Variance Prior $p(\lambda^2)$	Marginal Prior $p(w)$
Bernoulli	Bernoulli	Spike-and-Slab
Gaussian	χ^2	Unnamed
Rayleigh	Exponential	Laplace
Inverse Nagakami	Γ^{-1}	Student-t
Half-Cauchy	Unnamed	Horseshoe

[Nalisnick et al., 2015]

Model Changes

3. Skip Connections: Identity connections to previous layer [He et al., 2015].

$$\mathbf{h}_i^l = f_l(\mathbf{h}_i^{l-1}\mathbf{W}_l + \mathbf{b}_l)$$

NONLINEAR TRANSFORM

OLD

VS

$$\mathbf{h}_i^l = f_l(\mathbf{h}_i^{l-1}\mathbf{W}_l + \mathbf{b}_l) + \mathbf{h}_i^{l-1}$$

SKIP CONNECTION

NEW

Model Changes

3. Skip Connections: Identity connections to previous layer [He et al., 2015].

$$\mathbf{h}_i^l = f_l(\mathbf{h}_i^{l-1}\mathbf{W}_l + \mathbf{b}_l)$$

NONLINEAR TRANSFORM

OLD

VS

$$\mathbf{h}_i^l = f_l(\mathbf{h}_i^{l-1}\mathbf{W}_l + \mathbf{b}_l) + \mathbf{h}_i^{l-1}$$

SKIP CONNECTION

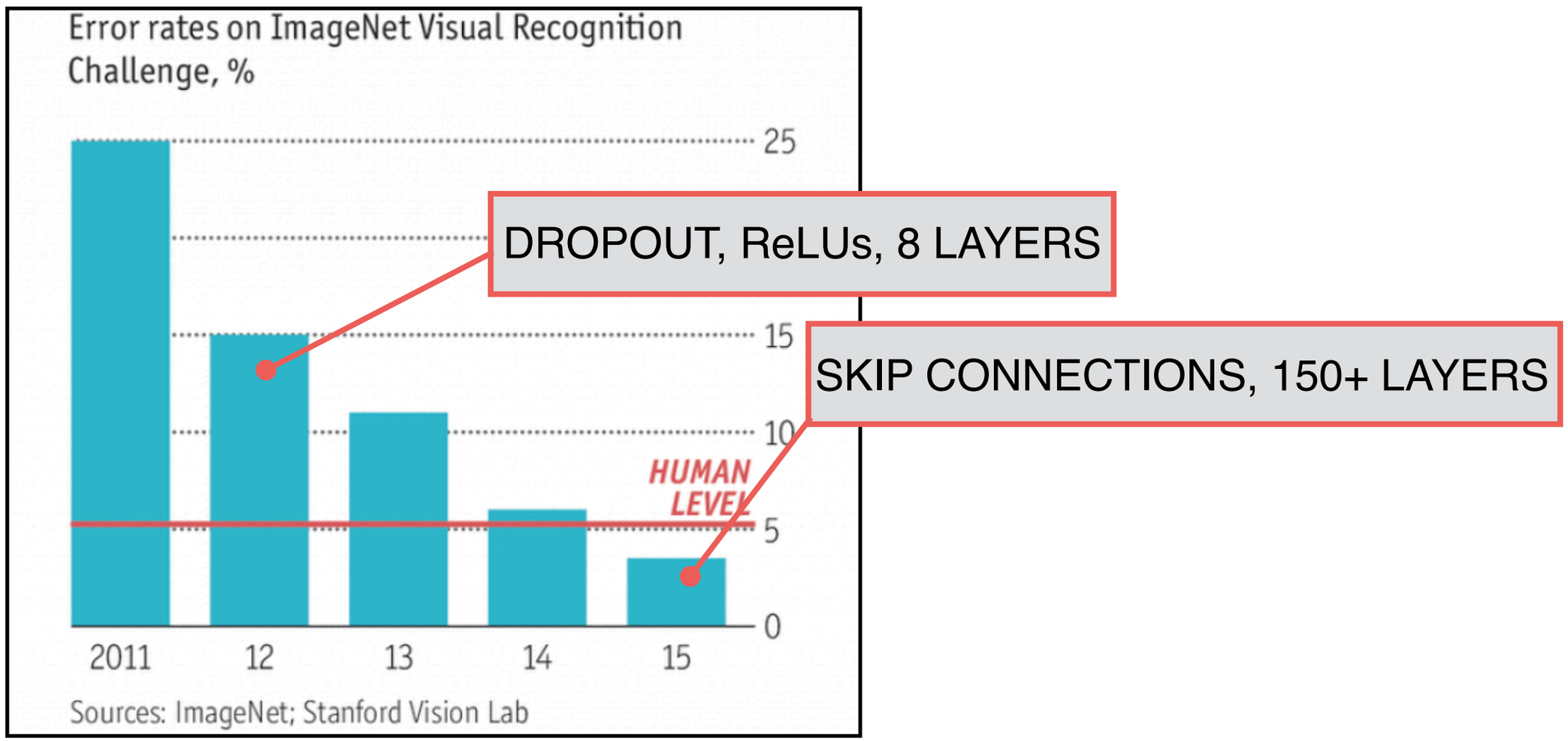
NEW

Can think of skip-connections as modeling the residual transformation:

$$\mathbf{h}_i^l - \mathbf{h}_i^{l-1} = f_l(\mathbf{h}_i^{l-1}\mathbf{W}_l + \mathbf{b}_l)$$

Deep Learning Results

Computer Vision: Results on ImageNet object classification dataset.



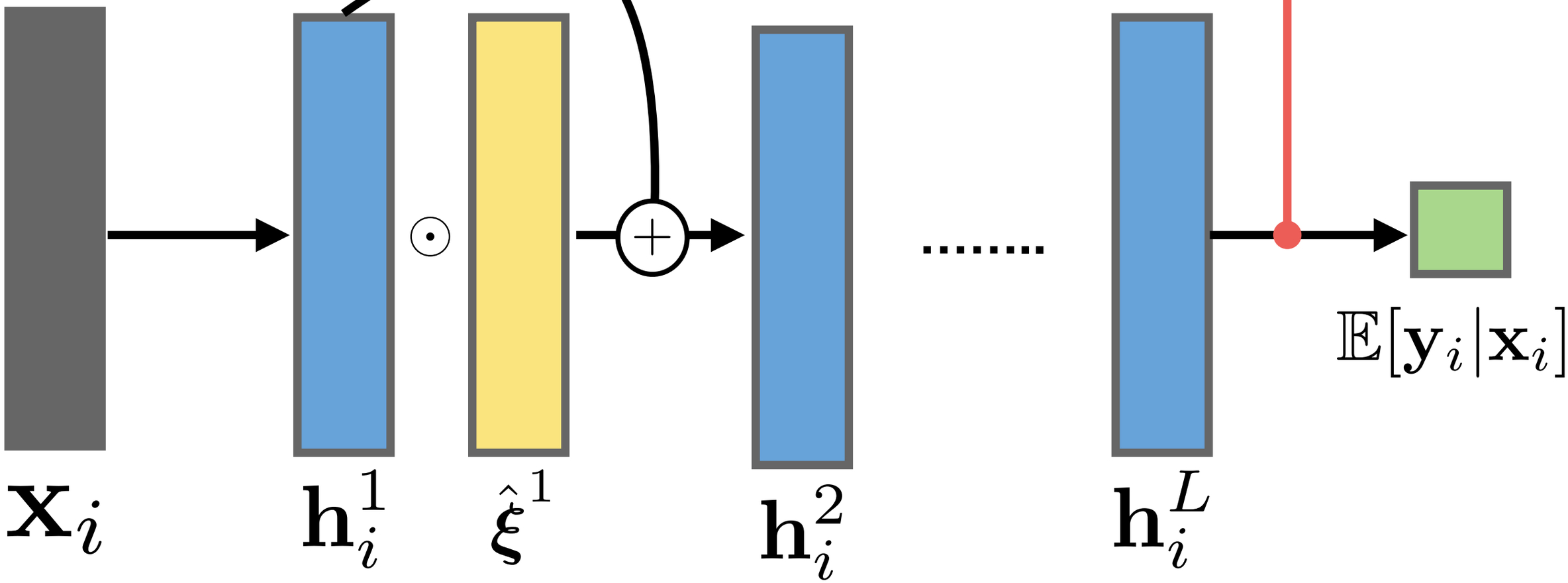
https://www.economist.com/news/special-report/21700756-artificial-intelligence-boom-based-old-idea-modern-twist-not

Modern Deep Neural Networks

ReLU

$$\mathbf{h}_i^l = f_l\left((\mathbf{h}_i^{l-1} \odot \hat{\boldsymbol{\xi}}^l)\mathbf{W}_l + \mathbf{b}_l\right) + \mathbf{h}_i^{l-1}$$

$$\mathbb{E}[\mathbf{y}_i|\mathbf{x}_i] = g^{-1}(\mathbf{h}_i^L \mathbf{w}_o + b_o)$$



$\hat{\boldsymbol{\xi}}^1 \sim \text{Bernoulli}(p)$

Factor Analysis and Autoencoders

Factor Analysis and Autoencoders

- **Factor Analysis**:

$$\mathbf{x}_i = \mathbf{z}_i \mathbf{A} + \mathbf{b} + \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathrm{N}(\mathbf{0}, \boldsymbol{\Sigma}_0)$$

loading matrix

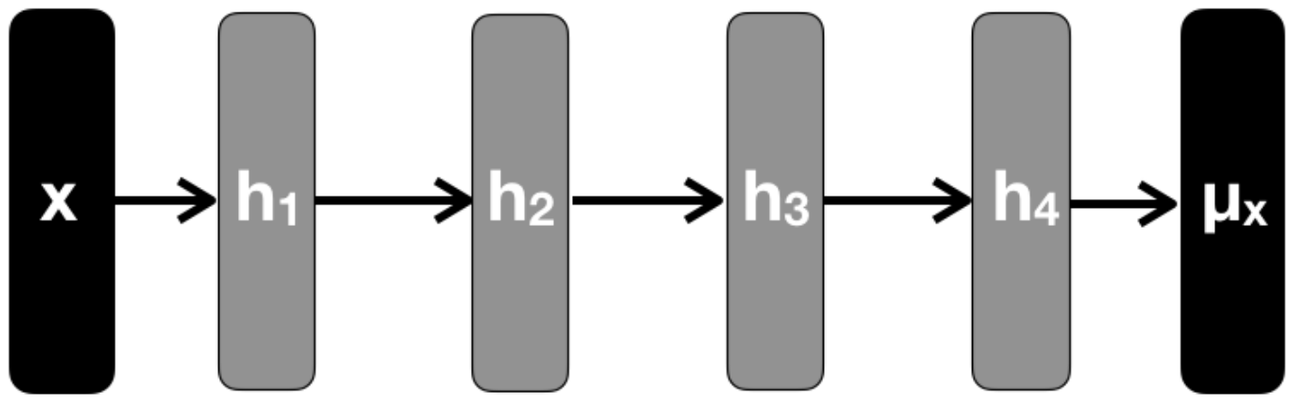
Factor Analysis and Autoencoders

Factor Analysis:

$$\mathbf{x}_i = \mathbf{z}_i \mathbf{A} + \mathbf{b} + \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathrm{N}(\mathbf{0}, \boldsymbol{\Sigma}_0)$$

loading matrix

Autoencoder:



$$\mathbb{E}[\mathbf{x}_i|\mathbf{x}_i] = g^{-1}(\mathbf{h}_{i,L}\mathbf{W}_{L+1} + \mathbf{b}_{L+1}), \quad \mathbf{h}_{i,l} = f_L(\mathbf{h}_{i,l-1}\mathbf{W}_l + \mathbf{b}_l), \quad \mathbf{h}_{i,0} = \mathbf{x}_i$$

Equivalence

- **Principal Components Analysis (PCA):** Both models can be shown equivalent to PCA under certain modifications. [Tipping & Bishop, 1999] [Baldi & Hornik, 1989]

Equivalence

- **Principal Components Analysis (PCA):** Both models can be shown equivalent to PCA under certain modifications. [Tipping & Bishop, 1999] [Baldi & Hornik, 1989]

- **Nonlinear Factor Analysis:** Factor analysis can be made non-linear in the factors and parameters [Amemiya, 1993].

$$\mathbf{x}_i = \psi(\mathbf{z}_i, \mathbf{A}) + \mathbf{b} + \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathrm{N}(\mathbf{0}, \boldsymbol{\Sigma}_0)$$

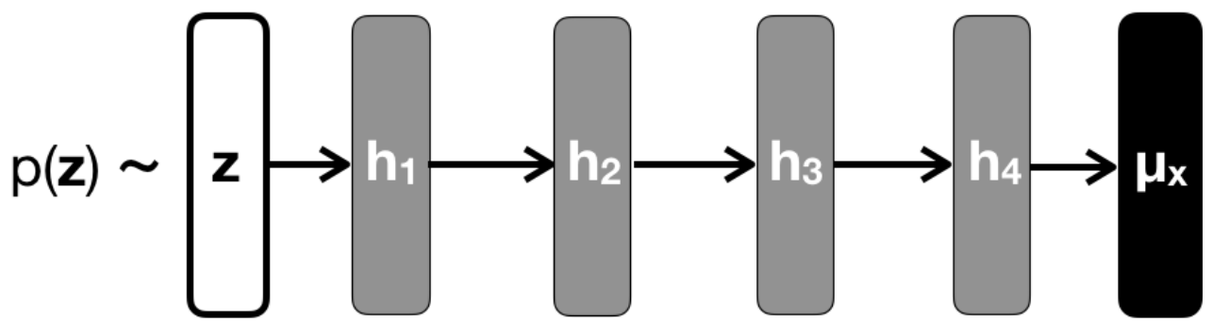
Equivalence

- **Principal Components Analysis (PCA):** Both models can be shown equivalent to PCA under certain modifications. [Tipping & Bishop, 1999] [Baldi & Hornik, 1989]

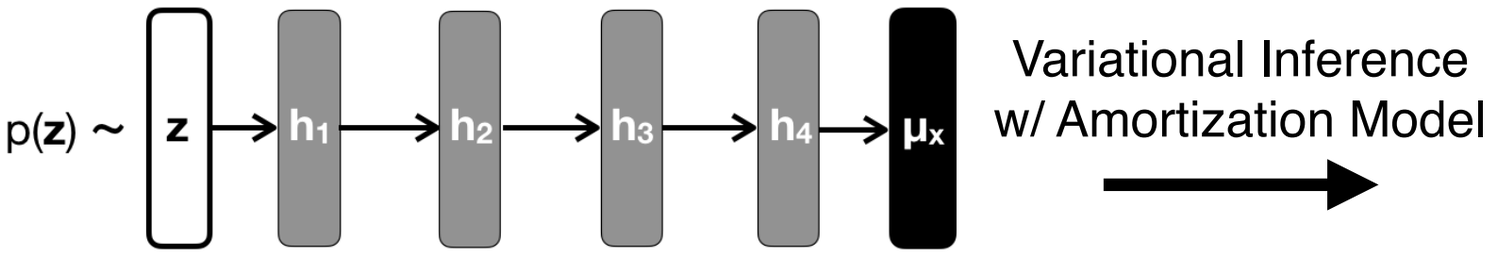
- **Nonlinear Factor Analysis:** Factor analysis can be made non-linear in the factors and parameters [Amemiya, 1993].

$$\mathbf{x}_i = \psi(\mathbf{z}_i, \mathbf{A}) + \mathbf{b} + \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathrm{N}(\mathbf{0}, \boldsymbol{\Sigma}_0)$$

If we assume the non-linearity is defined by a neural network, then we have what's known as a **density network** [MacKay & Gibbs, 1999].



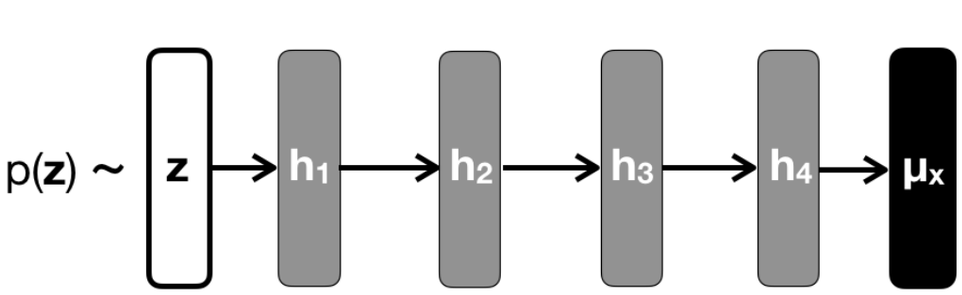
Equivalence



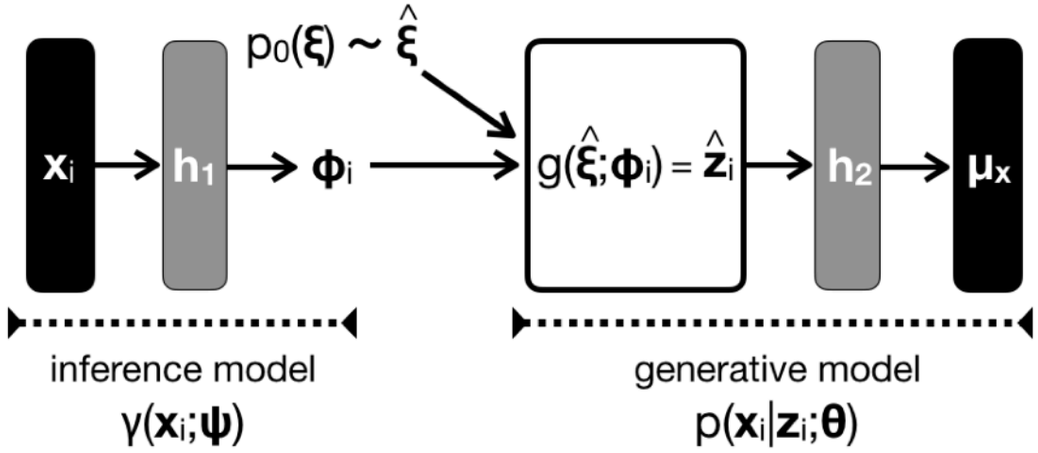
Variational Inference
w/ Amortization Model

Nonlinear Factor Analysis /
Density Network

Equivalence



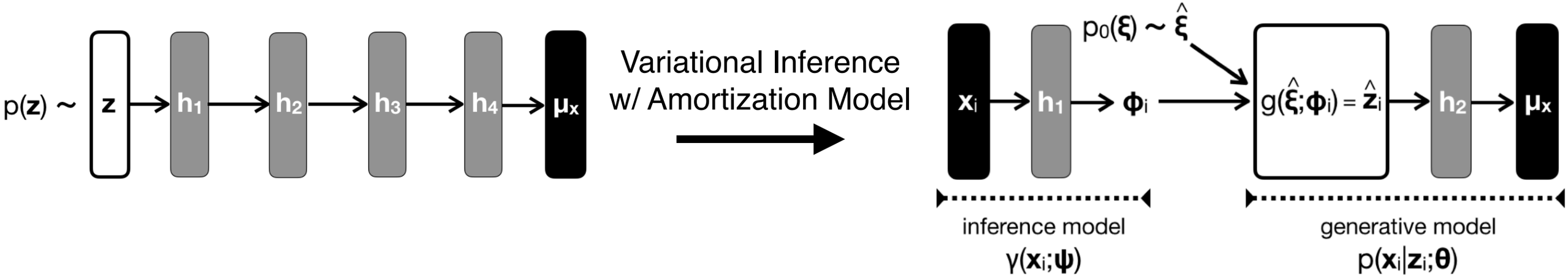
Variational Inference
w/ Amortization Model



Nonlinear Factor Analysis /
Density Network

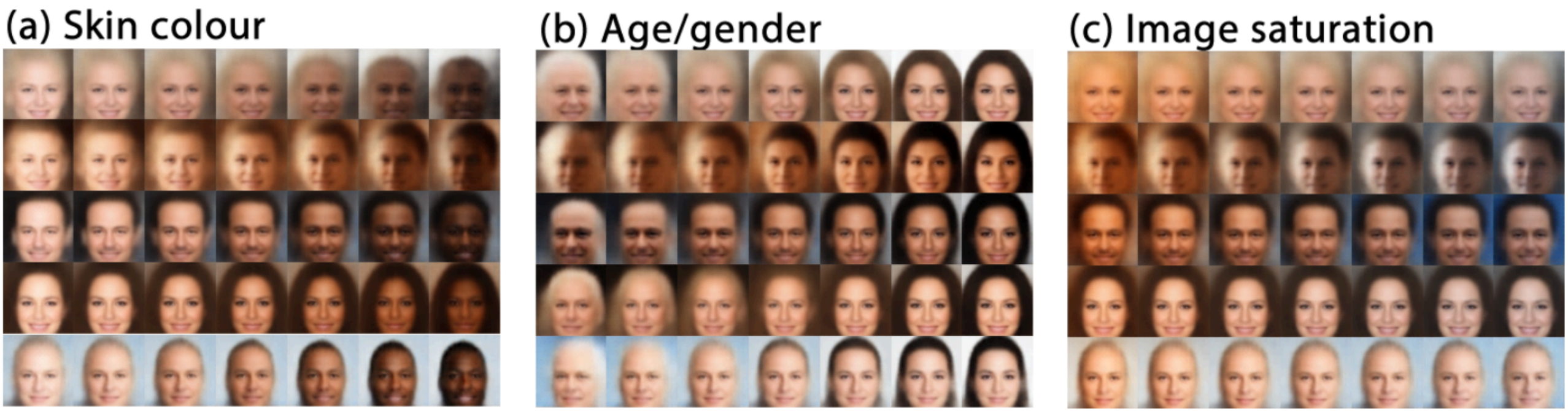
Variational Autoencoder
[Kingma & Welling, 2014]

Equivalence



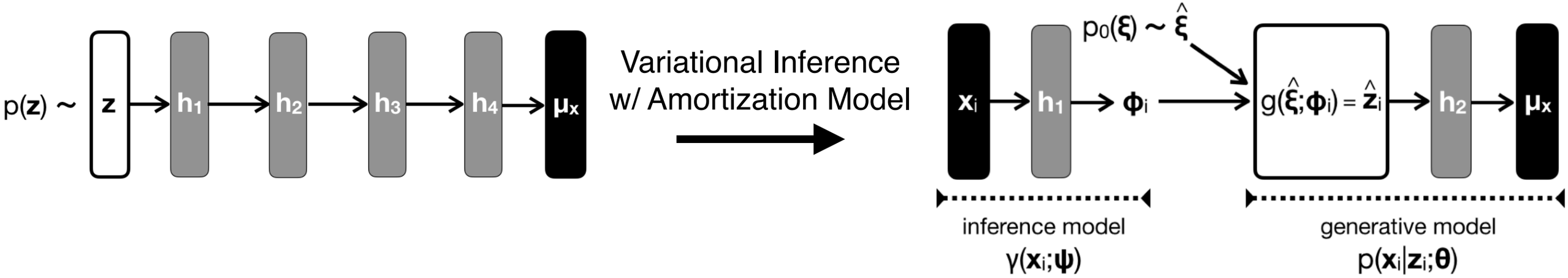
Nonlinear Factor Analysis /
Density Network

Variational Autoencoder
[Kingma & Welling, 2014]



[Higgins et al., 2017]

Equivalence



Nonlinear Factor Analysis /
Density Network

Variational Autoencoder
[Kingma & Welling, 2014]

Interpretable VAEs for nonlinear group factor analysis

Samuel K. Ainsworth[1], Nicholas J. Foti[1], Adrian KC Lee[2], and Emily B. Fox[1]

[1]School of Computer Science and Engineering, University of Washington
[2]Institute for Learning & Brain Sciences, University of Washington

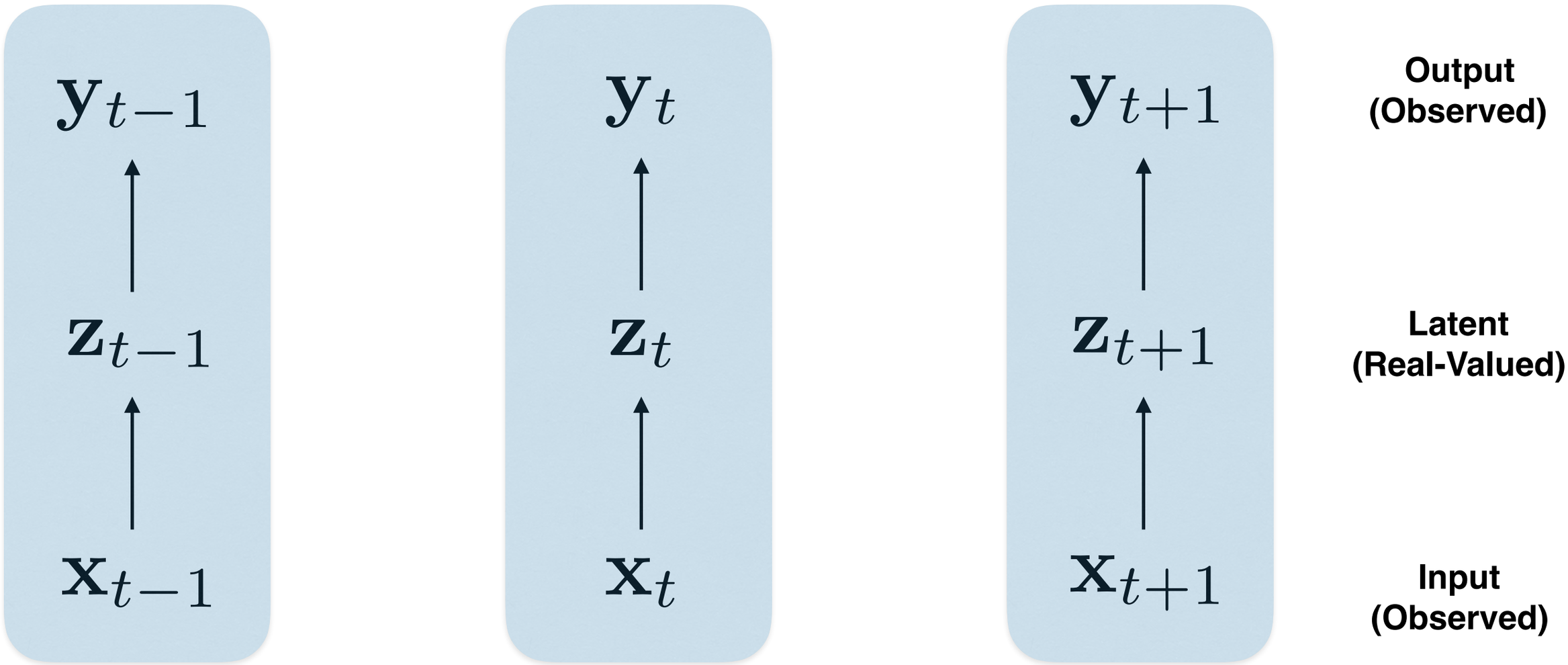
February 15, 2018

Abstract

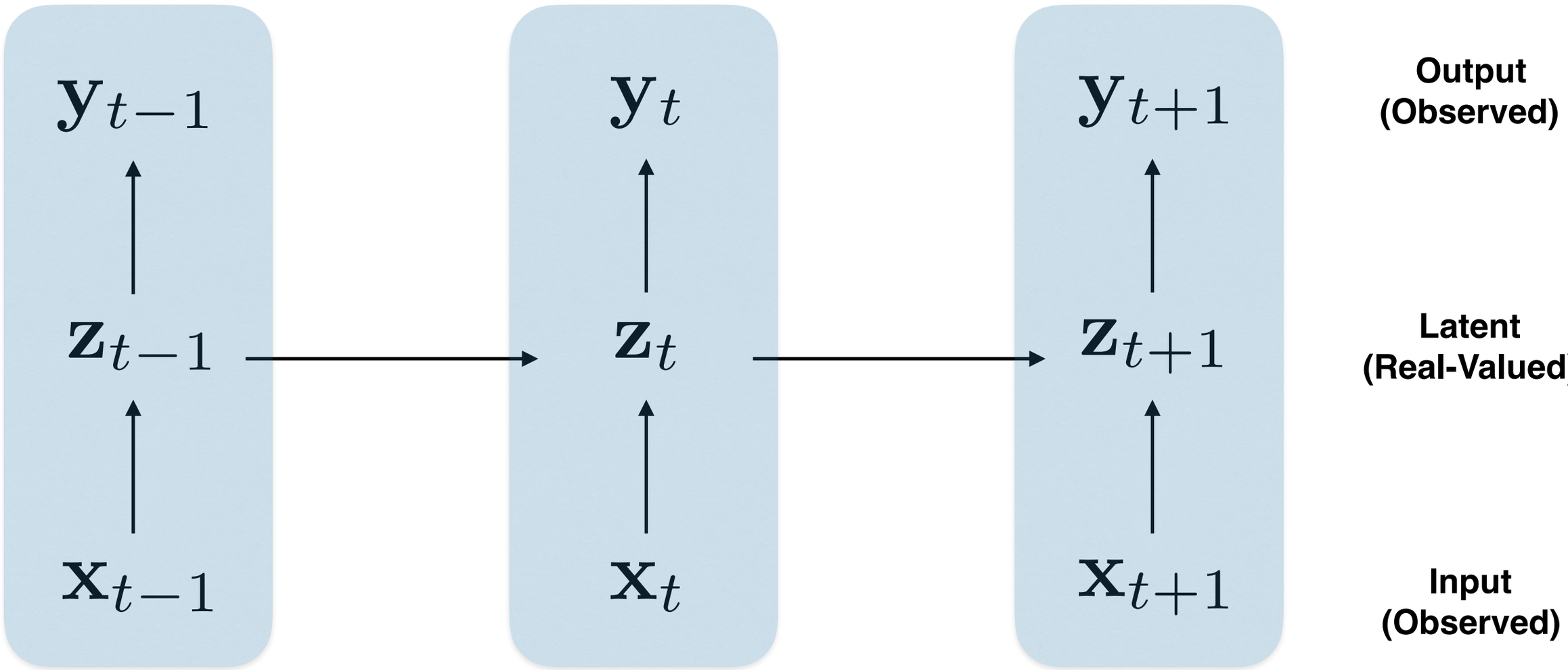
Deep generative models have recently yielded encouraging results in producing subjectively realistic samples of complex data. Far less attention has been paid to making these generative models interpretable. In many scenarios, ranging from scientific applications to finance, the observed variables have a natural grouping. It is often of interest to understand systems of interaction amongst these groups, and latent factor models (LFMs) are an attractive approach. However, traditional LFMs are limited by assuming a linear correlation structure. We present an output interpretable VAE (oi-VAE) for grouped data that models complex, nonlinear latent-to-observed relationships.

State-Space Models and Recurrent NNs

Prediction with Sequential Data

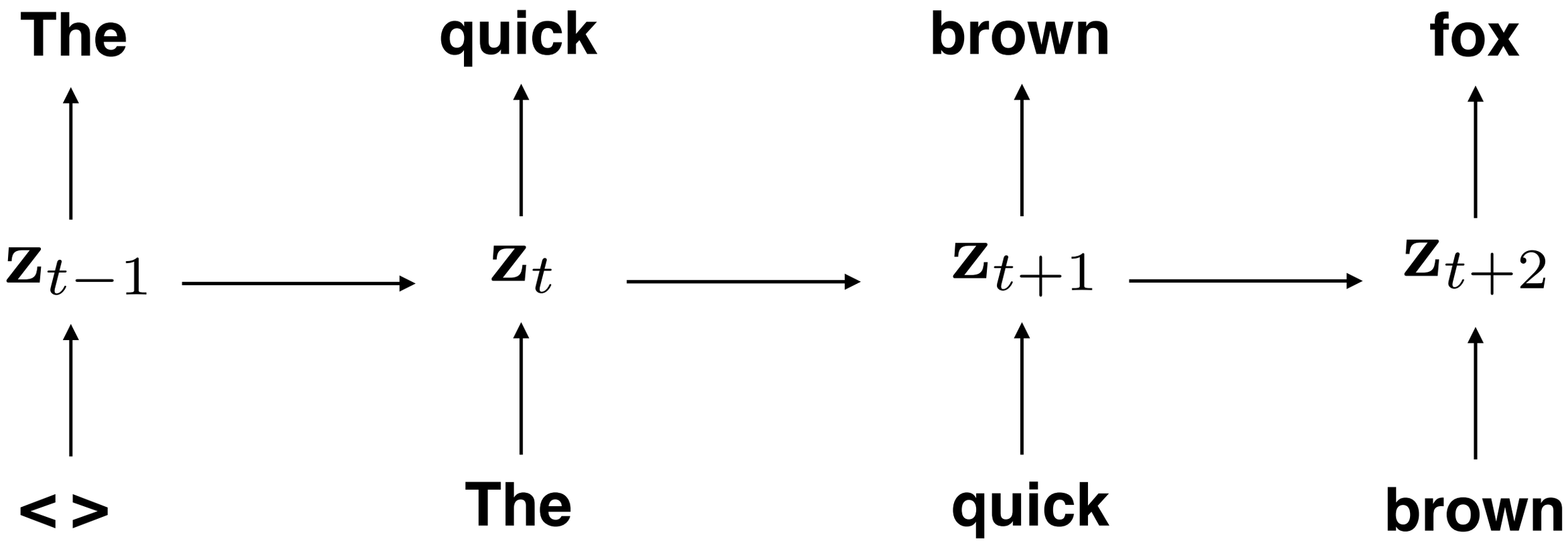


Recurrent Neural Network (RNN) Models



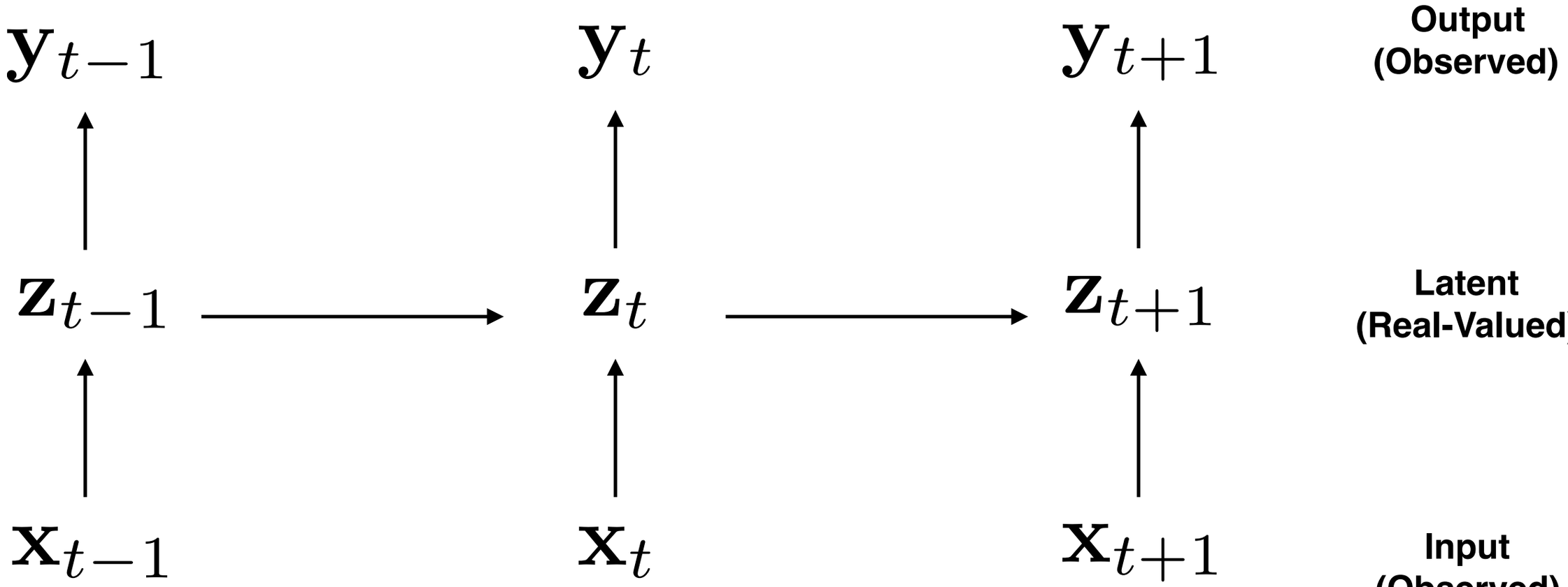
Each time-slice can be a deep network
….hidden (latent) vectors are coupled together over time

Language Modeling with RNNs



Many extensions, e.g,
Long-short-term-memory (LSTM) models for longer memory

State-Space Models in Statistics



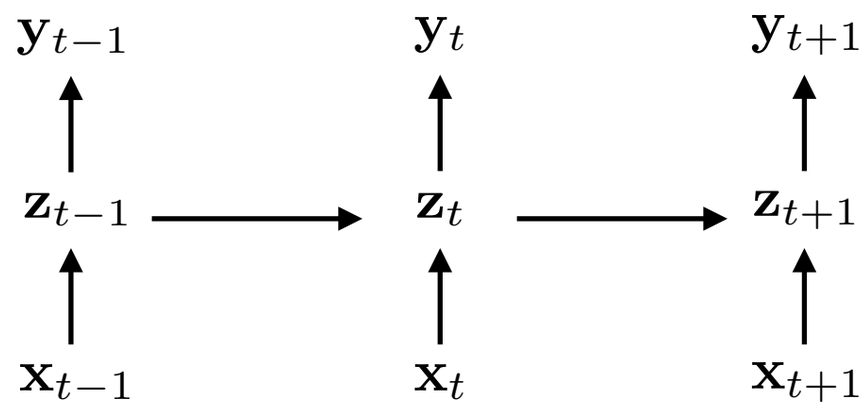
Kalman (1960)
Durbin and Koopman (2001)
Shumway and Stoffer (2016)

e.g., Linear-Gaussian assumptions
for dependencies in graph

Recurrent Neural Network (RNN) Models

State dynamics equation:

$$\mathbf{z}_t = f\Big(\mathbf{W}^{zz}\mathbf{z}_{t-1} \; + \; \mathbf{W}^{xz}\mathbf{x}_t \; + \; \mathbf{b}_z \Big)$$



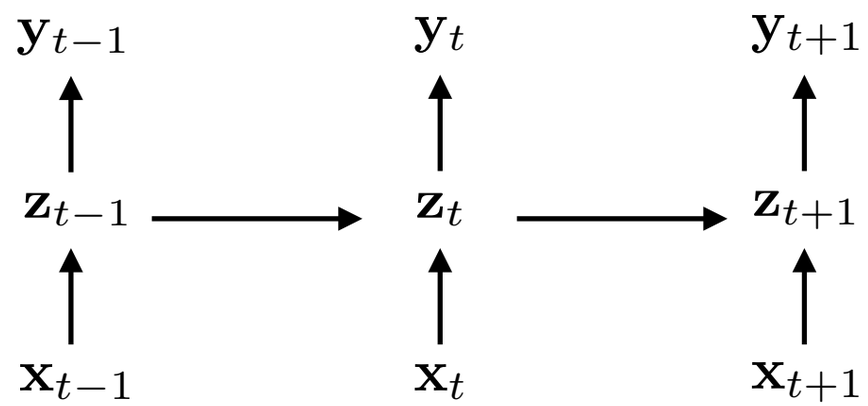
Observation equation:

$$\mathbf{y}_t = g\Big(\mathbf{W}^{zy}\mathbf{z}_t \; + \; \mathbf{b}_y \Big)$$

Recurrent Neural Network (RNN) Models

State dynamics equation:

$$\mathbf{z}_t = f\Big(\mathbf{W}^{zz}\mathbf{z}_{t-1} + \mathbf{W}^{xz}\mathbf{x}_t + \mathbf{b}_z \Big)$$

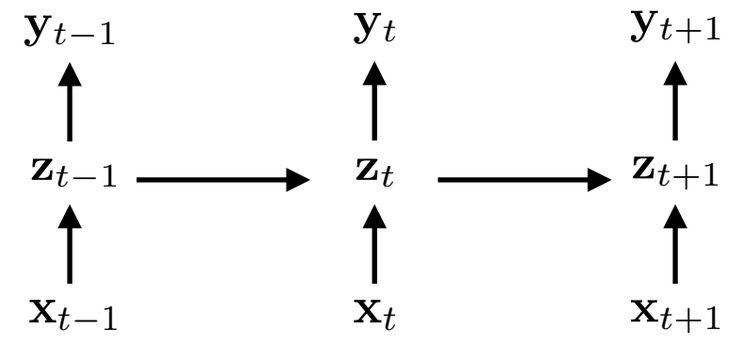


Observation equation:

$$\mathbf{y}_t = g\Big(\mathbf{W}^{zy}\mathbf{z}_t + \mathbf{b}_y \Big)$$

Key difference with state-space models:
Latent variables z are computed as deterministic (non-linear) functions
No distributional assumptions (e.g., Gaussian)
Both an advantage and a drawback

Comparing RNNs and State Space Models



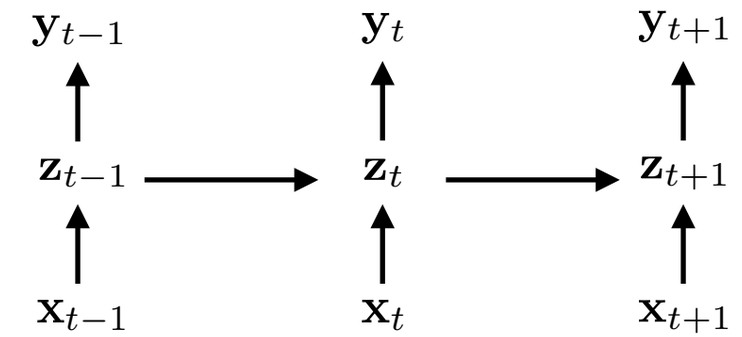
Similarities:

Use of latent state representation (e.g., for dimensionality reduction)

Independence of output at time t, given state at time t

Parameter estimation with gradient methods

Comparing RNNs and State Space Models



Similarities:

Use of latent state representation (e.g., for dimensionality reduction)

Independence of output at time t, given state at time t

Parameter estimation with gradient methods

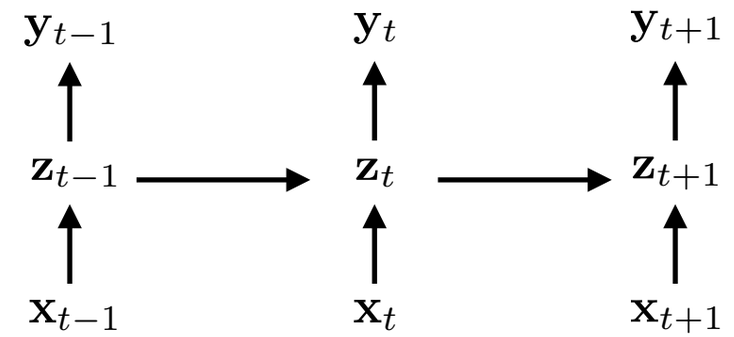
Differences:

RNN is more flexible (does not require distribution on z)

Harder for an RNN to handle missing data, do forecasting, be Bayesian

RNN is less interpretable

Comparing RNNs and State Space Models



Similarities:

Use of latent state representation (e.g., for dimensionality reduction)

Independence of output at time t, given state at time t

Parameter estimation with gradient methods

Differences:

RNN is more flexible (does not require distribution on z)

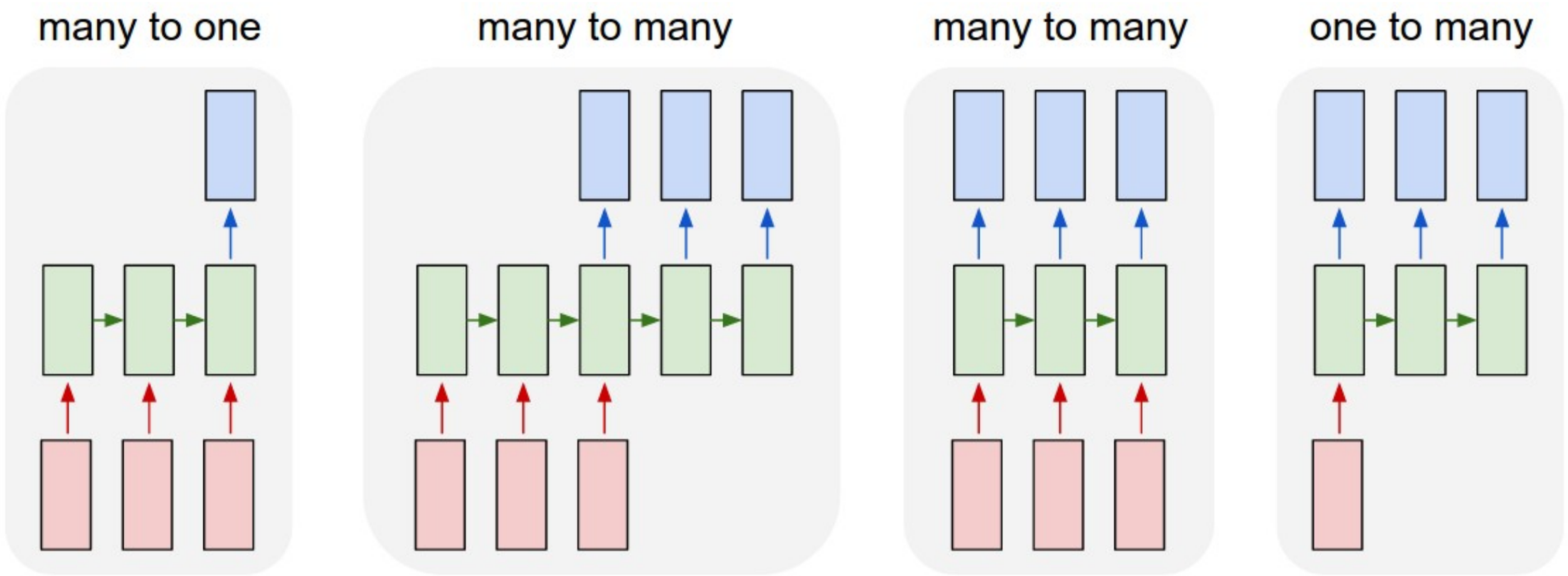
Harder for an RNN to handle missing data, do forecasting, be Bayesian

RNN is less interpretable

Exploration of hybrid RNN/state-space models: active area of research

e.g., see Fraccaro et al., 2016, Krishnan et al 2017, Zheng et al, 2017

The Many Variants of Recurrent Network Models



Figures from http://cs231n.stanford.edu/slides/2017/

Open Problems

Open Problems

- **Better understanding of NN optimization and regularization:** *Why don't they overfit more often? Why does stochastic gradient work unexpectedly well? Are there alternative optimization strategies?*

Open Problems

- **Better understanding of NN optimization and regularization:** *Why don't they overfit more often? Why does stochastic gradient work unexpectedly well? Are there alternative optimization strategies?*

- **Make NNs less 'data hungry':** *Better regularization methods? Using Bayesian formulations to better regularize / incorporate inductive biases / leverage prior information.*

Open Problems

- **Better understanding of NN optimization and regularization:** *Why don't they overfit more often? Why does stochastic gradient work unexpectedly well? Are there alternative optimization strategies?*

- **Make NNs less 'data hungry':** *Better regularization methods? Using Bayesian formulations to better regularize / incorporate inductive biases / leverage prior information.*

- **More exploration of non-image, non-iid data:** *What if data are correlated, censored, biased, etc., how do NNs behave?*

Open Problems

- **Better understanding of NN optimization and regularization:** *Why don't they overfit more often? Why does stochastic gradient work unexpectedly well? Are there alternative optimization strategies?*

- **Make NNs less 'data hungry':** *Better regularization methods? Using Bayesian formulations to better regularize / incorporate inductive biases / leverage prior information.*

- **More exploration of non-image, non-iid data:** *What if data are correlated, censored, biased, etc., how do NNs behave?*

- **Model interpretation and uncertainty quantification:** *How can we understand what the model has learned? When will it fail? Can we get calibrated uncertainty estimates?*

Questions Left Unanswered in a Previous Seminar

Questions Left Unanswered in a Previous Seminar

1. **Bayesian Methods:** *What is the role of Bayesian methods in deep learning? Can they help with architecture selection, for instance?*

Questions Left Unanswered in a Previous Seminar

1. **Bayesian Methods:** *What is the role of Bayesian methods in deep learning? Can they help with architecture selection, for instance?*

- [ ] **Selecting the Number of Nodes:** [Neal, 1994], [MacKay, 1994], [Lawrence, 2002], [Adams et al., 2010], [Nalisnick & Smyth, 2017], [Ghosh & Doshi-Velez, 2017]

Questions Left Unanswered in a Previous Seminar

1. **Bayesian Methods:** *What is the role of Bayesian methods in deep learning? Can they help with architecture selection, for instance?*

- [ ] **Selecting the Number of Nodes:** [Neal, 1994], [MacKay, 1994], [Lawrence, 2002], [Adams et al., 2010], [Nalisnick & Smyth, 2017], [Ghosh & Doshi-Velez, 2017]

- [ ] **Selecting the Number of Layers:** [Adams et al., 2010], [Graves, 2016]

Questions Left Unanswered in a Previous Seminar

1. **Bayesian Methods:** *What is the role of Bayesian methods in deep learning? Can they help with architecture selection, for instance?*

- [ ] **Selecting the Number of Nodes:** [Neal, 1994], [MacKay, 1994], [Lawrence, 2002], [Adams et al., 2010], [Nalisnick & Smyth, 2017], [Ghosh & Doshi-Velez, 2017]

- [ ] **Selecting the Number of Layers:** [Adams et al., 2010], [Graves, 2016]

- [ ] **Uncertainty Quantification for Safe Automation:** [Gal & Ghahramani, 2016], [Li & Gal, 2017], [Louizos & Welling, 2017], [Lakshminarayanan et al., 2017]

Questions Left Unanswered in a Previous Seminar

2. Inference ($\hat{\theta}$) vs Prediction ($\hat{y}$): *Statisticians 'make their money' answering inferential questions. How can deep learning help with that?*

Questions Left Unanswered in a Previous Seminar

2. Inference ($\hat{\theta}$) vs Prediction ($\hat{y}$): *Statisticians 'make their money' answering inferential questions. How can deep learning help with that?*

☐ **Using Deep Learning to Improve the Speed / Accuracy of Inference**

- **MCMC:** [Zhang et al., 2017], [L. Li et al., 2017], [Y. Li et al., 2017], [Levy et al., 2018]

Questions Left Unanswered in a Previous Seminar

2. Inference ($\hat{\theta}$) vs Prediction ($\hat{y}$): *Statisticians 'make their money' answering inferential questions. How can deep learning help with that?*

☐ **Using Deep Learning to Improve the Speed / Accuracy of Inference**

- **MCMC:** [Zhang et al., 2017], [L. Li et al., 2017], [Y. Li et al., 2017], [Levy et al., 2018]

- **Variational Inference:** [Kingma & Welling, 2014], [Burda et al., 2015], [Li & Turner, 2016], [Ranganath et al., 2016], [Mescheder et al., 2017], [Tran et al., 2017], [Feng & Liu, 2017] (+ many more)

Questions Left Unanswered in a Previous Seminar

3. **Non-IID Data:** *Can deep learning methods still be useful for non-iid data (e.g. observational studies)?*

Questions Left Unanswered in a Previous Seminar

3. Non-IID Data: *Can deep learning methods still be useful for non-iid data (e.g. observational studies)?*

- [ ] **Rubin-Neyman Potential Outcomes:** [Johansson et al., 2016]

Questions Left Unanswered in a Previous Seminar

3. Non-IID Data: *Can deep learning methods still be useful for non-iid data (e.g. observational studies)?*

- ☐ **Rubin-Neyman Potential Outcomes:** [Johansson et al., 2016]
- ☐ **Causal Graphs:** [Hartford et al., 2017], [Louizos et al., 2017], [Tran & Blei, 2018]

Questions Left Unanswered in a Previous Seminar

3. Non-IID Data: *Can deep learning methods still be useful for non-iid data (e.g. observational studies)?*

- [ ] **Rubin-Neyman Potential Outcomes:** [Johansson et al., 2016]
- [ ] **Causal Graphs:** [Hartford et al., 2017], [Louizos et al., 2017], [Tran & Blei, 2018]
- [ ] **Deep Survival Analysis:** [Ranganath et al., 2016]

Questions Left Unanswered in a Previous Seminar

3. Non-IID Data: *Can deep learning methods still be useful for non-iid data (e.g. observational studies)?*

- [ ] **Rubin-Neyman Potential Outcomes:** [Johansson et al., 2016]
- [ ] **Causal Graphs:** [Hartford et al., 2017], [Louizos et al., 2017], [Tran & Blei, 2018]
- [ ] **Deep Survival Analysis:** [Ranganath et al., 2016]
- [ ] **Relational Data / Networks:** [Duvenaud et al., 2016], [Kipf & Welling, 2017] (+ many more)

Conclusions and Summary

PROBLEM: Prior to 2000's, NNs with 4+ layers were hard to train. SVMs worked just as well as shallow NNs.

SOLUTIONS:

- **More Data**: Many of the datasets being tested were too small, resulting in the NNs overfitting.
  - → **ImageNet Dataset**: 15 million images, 22,000 classes
- **Better Hardware**: And when the datasets were large, the hardware was not fast enough to make NN training practical.
  - → **Graphics Processing Units (GPUs)**: Can perform fast matrix multiplications (but with loss of precision).
- **Model Changes**: Small but crucial changes to the NN model architecture.
- **Better Understanding of Gradient Ascent**: Gradient-based optimization is better in non-convex settings than previously thought.

Summary: Key Points from This Talk

- [ ] Deep neural networks are very flexible non-linear models
- [ ] Models tend to be very complex, require lots of labeled data
- [ ] Focus is on prediction, not on model interpretation
- [ ] “Learning” is often regularized maximum likelihood via gradient methods

Many links to statistical thinking:

There is much that statistics can contribute to deep learning.

Statistics will benefit from an awareness about deep learning.

Further Reading from a Statistical Perspective

- [ ] L. Breiman
Statistical Modeling: The Two Cultures
Statistical Science, 2001

- [ ] B. Efron and T. Hastie
Chapter 18: Neural Networks and Deep Learning
Computer Age Statistical Inference
Cambridge University Press, 2016

- [ ] S. Mohamed
A Statistical View of Deep Learning
Blog post: http://blog.shakirm.com/wp-content/uploads/2015/07/SVDL.pdf, 2015

- [ ] E. Nalisnick, D. Tran, D. Blei, and P. Smyth
Deep Learning: A Synthesis from Probabilistic Foundations
coming soon…

Thank you. Questions?

SLIDES: `http://www.ics.uci.edu/~enalisni/DLforStats.pdf`

References

Ryan Adams, Hanna Wallach, and Zoubin Ghahramani. Learning the structure of deep sparse graphical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 1–8, 2010.

Samuel Ainsworth, Nicholas J. Foti, Adrian K. C. Lee, and Emily B. Fox. Interpretable vaes for nonlinear group factor analysis. *ArXiv Preprint*, 2018.

Yasuo Amemiya. On nonlinear factor analysis. In *Proceedings of the ASA Social Statistics Section*, pages 290–294, 1993.

Pierre Baldi and Kurt Hornik. Neural networks and principal component analysis: Learning from examples without local minima. *Neural networks*, 2(1):53–58, 1989.

Leo Breiman. Statistical modeling: The two cultures. *Statistical science*, 16(3):199–231, 2001.

Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. *International Conference on Learning Representations (ICLR)*, 2016.

Yann N Dauphin, Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Surya Ganguli, and Yoshua Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *Advances in neural information processing systems*, pages 2933–2941, 2014.

J. Deng, K. Li, M. Do, H. Su, and L. Fei-Fei. Construction and Analysis of a Large Scale Image Ontology. Vision Sciences Society, 2009.

J Durbin and SJ Koopman. Time series analysis by state space methods, 2001.

David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. Convolutional networks on graphs for learning molecular fingerprints. In *Advances in neural information processing systems*, pages 2224–2232, 2015.

References

Bradley Efron and Trevor Hastie. *Computer age statistical inference*, volume 5. Cambridge University Press, 2016.

Yihao Feng, Dilin Wang, and Qiang Liu. Learning to draw samples with amortized stein variational gradient descent. In *Proceedings of the Thirty-Third Conference on Uncertainty in Artificial Intelligence*. AUAI Press, 2017.

Marco Fraccaro, Søren Kaae Sønderby, Ulrich Paquet, and Ole Winther. Sequential neural models with stochastic layers. In *Advances in neural information processing systems*, pages 2199–2207, 2016.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of The 33rd International Conference on Machine Learning (ICML)*, pages 1050–1059, 2016.

Soumya Ghosh and Finale Doshi-Velez. Model selection in bayesian neural networks via horseshoe priors. *NIPS Workshop on Bayesian Deep Learning*, 2017.

Alex Graves. Adaptive computation time for recurrent neural networks. *ArXiv Preprint*, 2016.

Moritz Hardt, Ben Recht, and Yoram Singer. Train faster, generalize better: Stability of stochastic gradient descent. In *International Conference on Machine Learning*, pages 1225–1234, 2016.

Jason Hartford, Greg Lewis, Kevin Leyton-Brown, and Matt Taddy. Deep iv: A flexible approach for counterfactual prediction. In *International Conference on Machine Learning*, pages 1414–1423, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. β-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. *International Conference on Learning Representations (ICLR)*, 2017.

References

Fredrik Johansson, Uri Shalit, and David Sontag. Learning representations for counterfactual inference. In *International Conference on Machine Learning*, pages 3020–3029, 2016.

Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

Diederik Kingma and Max Welling. Auto-encoding variational Bayes. *International Conference on Learning Representations (ICLR)*, 2014.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *International Conference on Learning Representations*, 2017.

Rahul G Krishnan, Uri Shalit, and David Sontag. Structured inference networks for nonlinear state space models. In *AAAI*, pages 2101–2109, 2017.

Lynn Kuo and Bani Mallick. Variable Selection for Regression Models. *Sankhyā: The Indian Journal of Statistics, Series B*, pages 65–81, 1998.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, pages 6405–6416, 2017.

Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th annual international conference on machine learning*, pages 609–616. ACM, 2009.

Daniel Levy, Matthew D Hoffman, and Jascha Sohl-Dickstein. Generalizing hamiltonian monte carlo with neural networks. *International Conference on Learning Representations (ICLR)*, 2018.

Chunyuan Li, Rosanne Liu, and Jason Yosinski. Measuring the Intrinsic Dimension of Objective Landscapes. *International Conference on Learning Representations (ICLR)*, 2018.

References

Lingge Li, Andrew Holbrook, Babak Shahbaba, and Pierre Baldi. Neural network gradient hamiltonian monte carlo. *arXiv preprint arXiv:1711.05307*, 2017.

Yingzhen Li and Yarin Gal. Dropout inference in bayesian neural networks with alpha-divergences. In *International Conference on Machine Learning*, pages 2052–2061, 2017.

Yingzhen Li and Richard E Turner. Variational inference with Renyi divergence. *Neural Information Processing Systems (NIPS)*, 2016.

Christos Louizos and Max Welling. Multiplicative normalizing flows for variational bayesian neural networks. In *International Conference on Machine Learning*, pages 2218–2227, 2017.

Christos Louizos, Uri Shalit, Joris M Mooij, David Sontag, Richard Zemel, and Max Welling. Causal Effect Inference with Deep Latent Variable Models. In *Advances in Neural Information Processing Systems*, pages 6449–6459, 2017.

David JC MacKay. Bayesian non-linear modeling for the prediction competition. In *Maximum Entropy and Bayesian Methods*, pages 221–234. Springer, 1994.

David JC MacKay and Mark N Gibbs. Density networks. *Statistics and neural networks: advances at the interface. Oxford University Press, Oxford*, pages 129–144, 1999.

Lars M. Mescheder, Sebastian Nowozin, and Andreas Geiger. Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks. *Proceedings of the 34th International Conference on Machine Learning*, 2017.

Andriy Mnih and Karol Gregor. Neural variational inference and learning in belief networks. In *International Conference on Machine Learning*, pages 1791–1799, 2014.

Shakir Mohamed. A statistical view of deep learning. *The Spectator: Shakir's Machine Learning Blog*, 2015.

References

Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

Eric Nalisnick and Padhraic Smyth. Stick-Breaking Variational Autoencoders. *International Conference on Learning Representations (ICLR)*, 2017.

Eric Nalisnick, Anima Anandkumar, and Padhraic Smyth. A Scale Mixture Perspective of Multiplicative Noise in Neural Networks. *ArXiv Preprint*, 2015.

Radford M Neal. *Bayesian Learning for Neural Networks*. PhD thesis, University of Toronto, 1994.

John Ashworth Nelder and R Jacob Baker. *Generalized Linear Models*. Wiley Online Library, 1972.

Rajesh Ranganath, Adler Perotte, Noémie Elhadad, and David Blei. Deep survival analysis. *Machine Learning for Healthcare*, 2016a.

Rajesh Ranganath, Dustin Tran, Jaan Altosaar, and David Blei. Operator variational inference. In *Advances in Neural Information Processing Systems*, pages 496–504, 2016b.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533, 1986.

Robert H Shumway and David S Stoffer. State-space models. In *Time series analysis and its applications*, pages 319–404. Springer, 2011.

Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of machine learning research*, 15(1):1929–1958, 2014.

References

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Michael E Tipping and Christopher M Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(3):611–622, 1999.

Dustin Tran and David Blei. Implicit Causal Models for Genome-wide Association Studies. *International Conference on Learning Representations (ICLR)*, 2018.

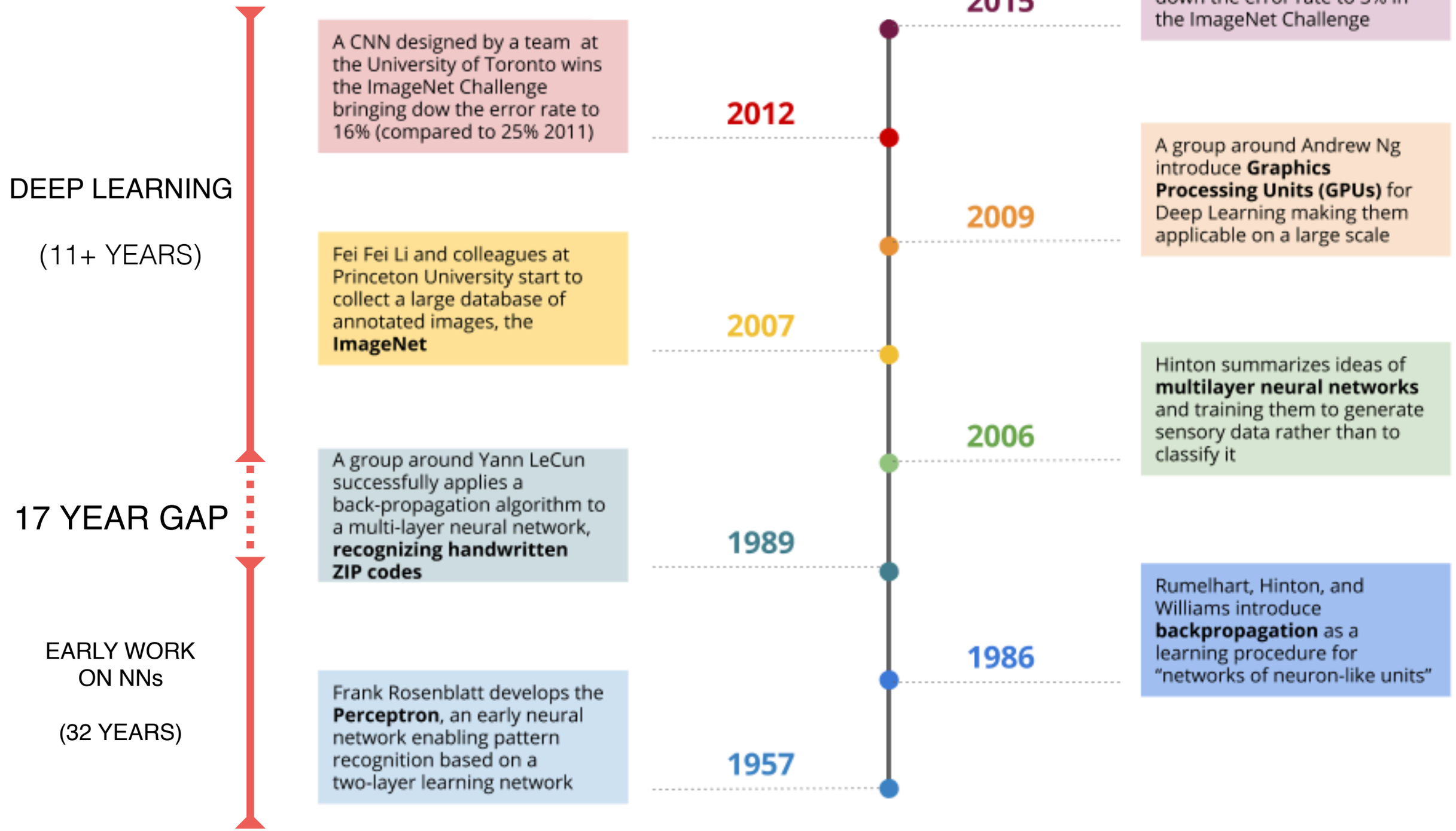
Dustin Tran, Rajesh Ranganath, and David M Blei. Deep and hierarchical implicit models. *Advances in Neural Information Processing Systems*, 2017.

Cheng Zhang, Babak Shahbaba, and Hongkai Zhao. Hamiltonian monte carlo acceleration using surrogate functions with random bases. *Statistics and computing*, 27(6):1473–1490, 2017.

Xun Zheng, Manzil Zaheer, Amr Ahmed, Yuan Wang, Eric P Xing, and Alexander J Smola. State space lstm models with particle mcmc inference. *arXiv preprint arXiv:1711.11179*, 2017.

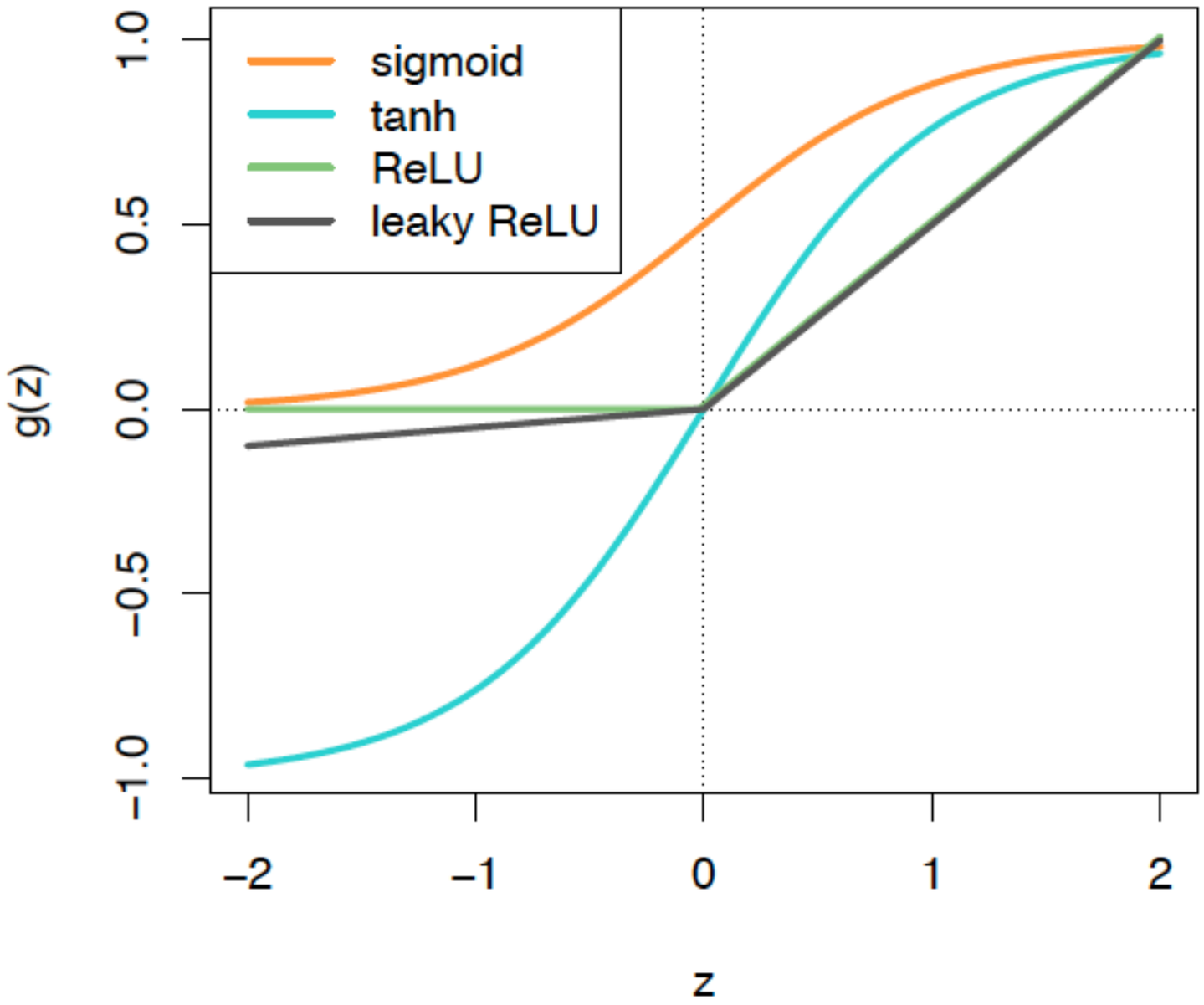
Appendix

Historical Perspective



https://micro-dimensions.com/blog/2017/2/13/whats-all-the-fuss-about-deep-learning-and-how-will-it-impact-pathology

Non-Linearities at Hidden Units



From Efron and Hastie, Computer-Aided Statistical Inference, 2016