Deep Learning for Natural Language Processing and Machine Translation

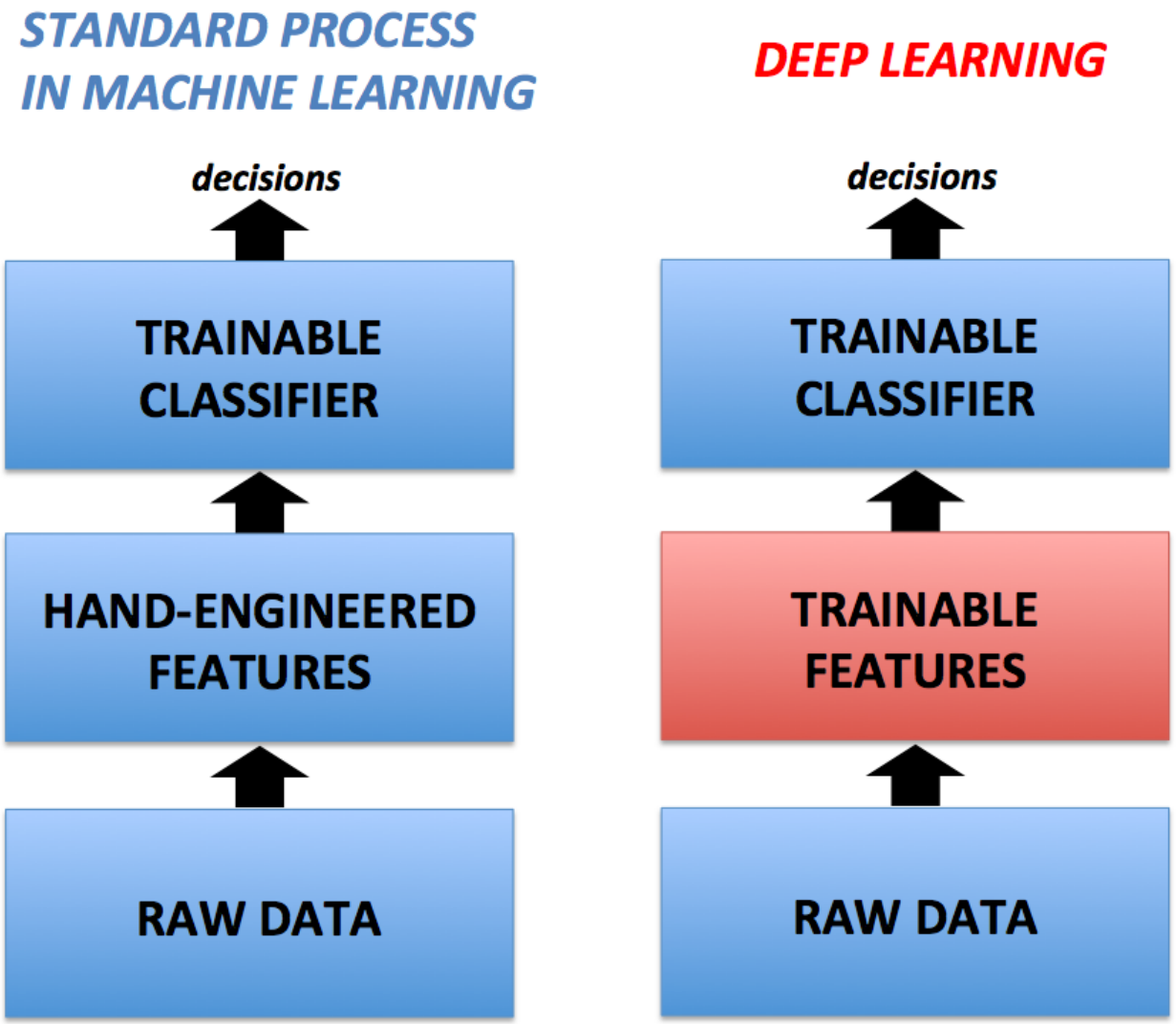
Kevin Duh

Nara Institute of Science and Technology, Japan

2014/11/04

What is Deep Learning?

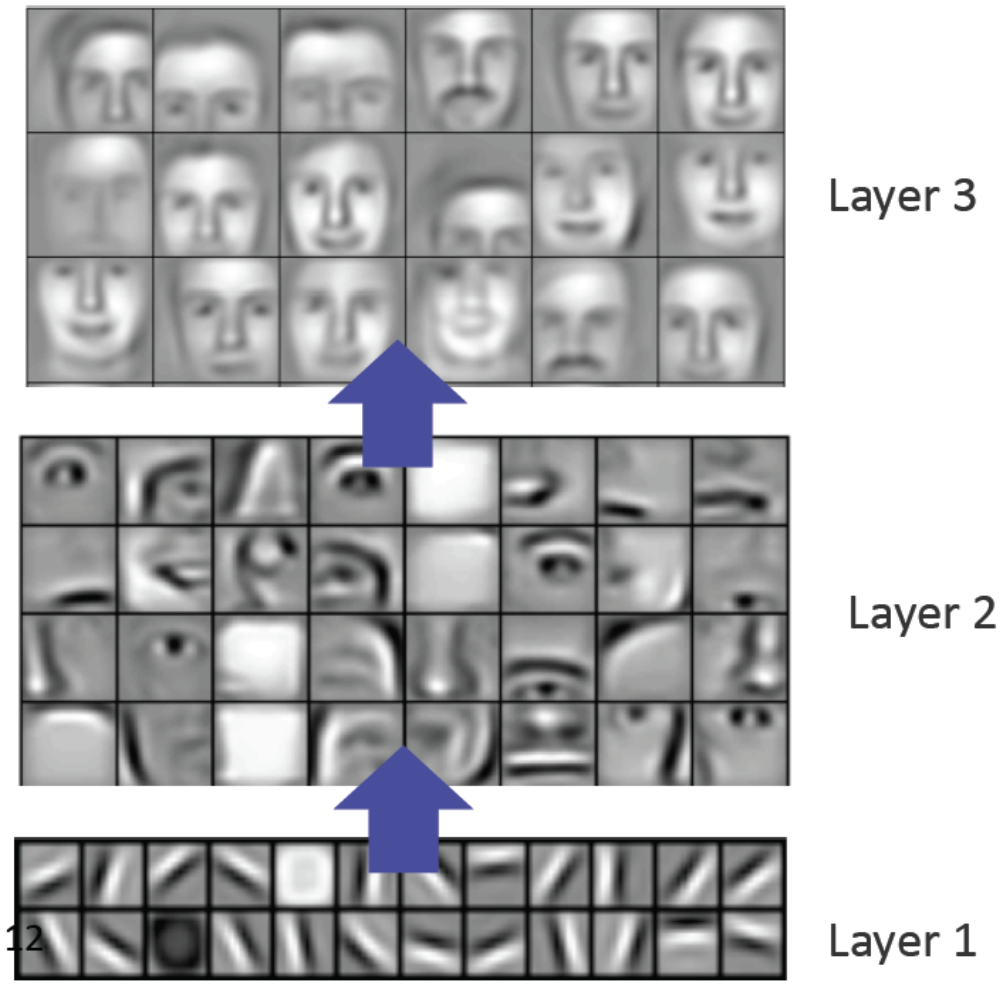
A family of methods that uses deep architectures to learn high-level feature representations



Example of Trainable Features [Lee et al., 2009]

Input: Images (raw pixels)

$\rightarrow$ Output: Features representing Edges, Body Parts, Full Faces



Outline

1. Deep Learning Background
   - Neural Networks (1-layer, 2-layer)
   - Potentials and Difficulties of Deep Architecture
   - The Breakthrough in 2006
2. Two Main Types of Deep Architectures
   - Deep Belief Nets (DBN) [Hinton et al., 2006]
   - Stacked Auto-Encoders (SAE) [Bengio et al., 2006]
   - Current Status of Deep Learning
3. Applications in Natural Language Processing and Machine Translation
   - Use as Non-linear classifier
   - Use as Distributed representation
   - Survey of Machine Translation Research

Outline

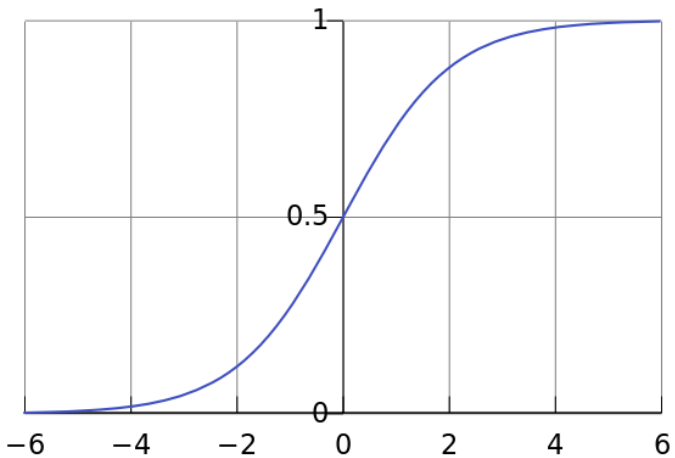
1. Deep Learning Background
   - Neural Networks (1-layer, 2-layer)
   - Potentials and Difficulties of Deep Architecture
   - The Breakthrough in 2006
2. Two Main Types of Deep Architectures
   - Deep Belief Nets (DBN) [Hinton et al., 2006]
   - Stacked Auto-Encoders (SAE) [Bengio et al., 2006]
   - Current Status of Deep Learning
3. Applications in Natural Language Processing and Machine Translation
   - Use as Non-linear classifier
   - Use as Distributed representation
   - Survey of Machine Translation Research

Problem Setup

- Training Data: a set of $(x^{(m)}, y^{(m)})_{m=\{1,2,..M\}}$ pairs
  - Input $x^{(m)} \in R^d$
  - Output $y^{(m)} = \{0, 1\}$
- Goal: Learn function $f : x \rightarrow y$ to predict correctly on new inputs x.
  - Step 1: Choose a function model family:
    - ★ e.g. logistic regression, support vector machines, neural networks
  - Step 2: Optimize parameters w on the Training Data
    - ★ e.g. minimize loss function $\min_w \sum_{m=1}^{M} (f_w(x^{(m)}) - y^{(m)})^2$

Logistic Regression (1-layer net)

- Function model: $f(x) = \sigma(w^T \cdot x)$
  - Parameters: vector $w \in R^d$
  - σ is a non-linearity, e.g. sigmoid: $\sigma(z) = 1/(1 + \exp(-z))$



- Non-linearity will be important in expressiveness multi-layer nets. Other non-linearities, e.g., $tanh(z) = (e^z - e^{-z})/(e^z + e^{-z})$

Gradient Descent for Logistic Regression

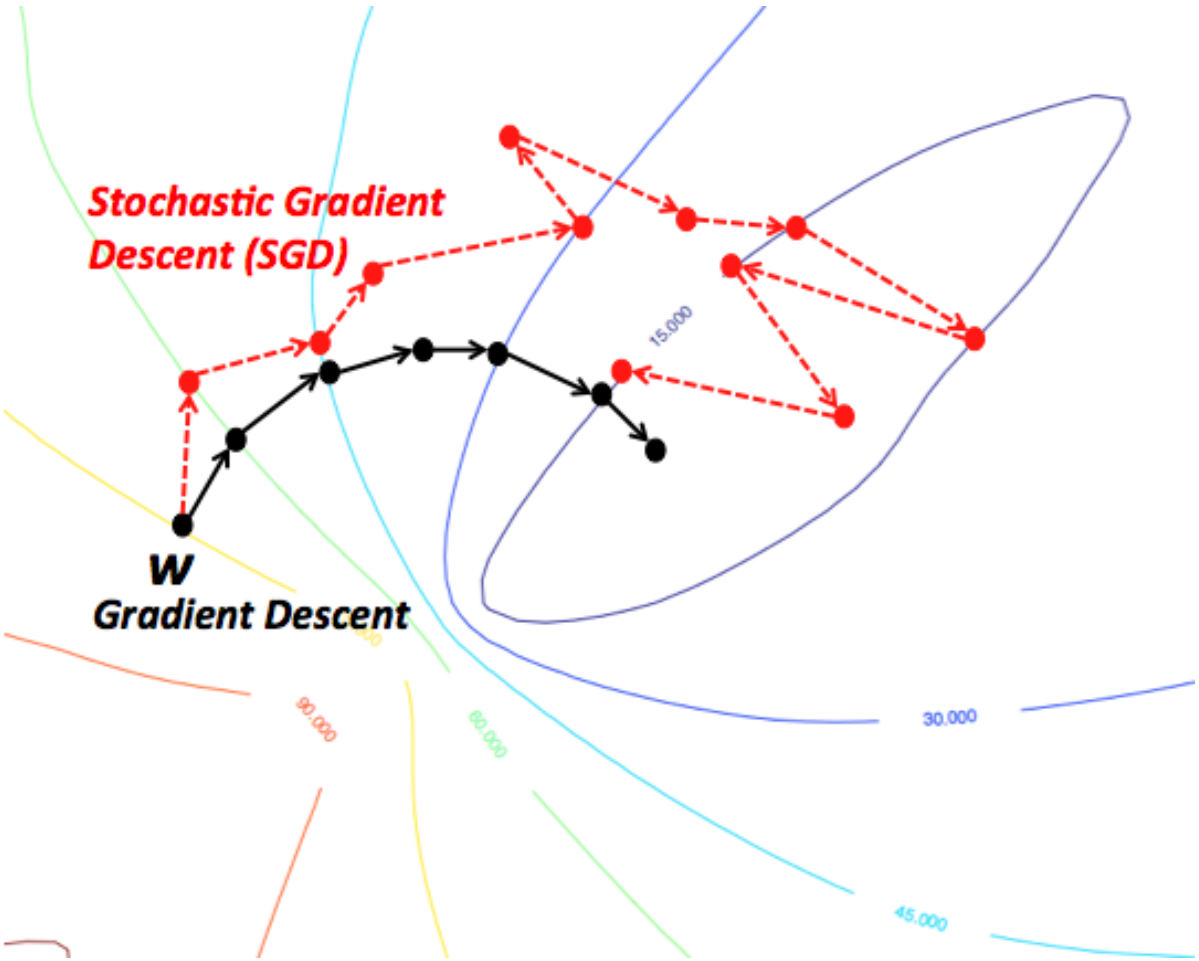
- Assume Squared-Error* $Loss(w) = \frac{1}{2}\sum_m(\sigma(w^T x^{(m)}) - y^{(m)})^2$
- Gradient: $\nabla_w Loss = \sum_m \left[\sigma(w^T x^{(m)}) - y^{(m)}\right]\sigma'(w^T x^{(m)})x^{(m)}$
  - Define input into non-linearity $in^{(m)} = w^T x^{(m)}$
  - General form of gradient: $\sum_m Error^{(m)} * \sigma'(in^{(m)}) * x^{(m)}$
  - Derivative of sigmoid $\sigma'(z) = \sigma(z)(1 - \sigma(z))$
- Gradient Descent Algorithm ():
  1. Initialize w randomly
  2. Update until convergence: $w \leftarrow w - \gamma(\nabla_w Loss)$
- Stochastic gradient descent (SGD) algorithm:
  1. Initialize w randomly
  2. Update until convergence: $w \leftarrow w - \gamma(Error^{(m)} * \sigma'(in^{(m)}) * x^{(m)})$

---

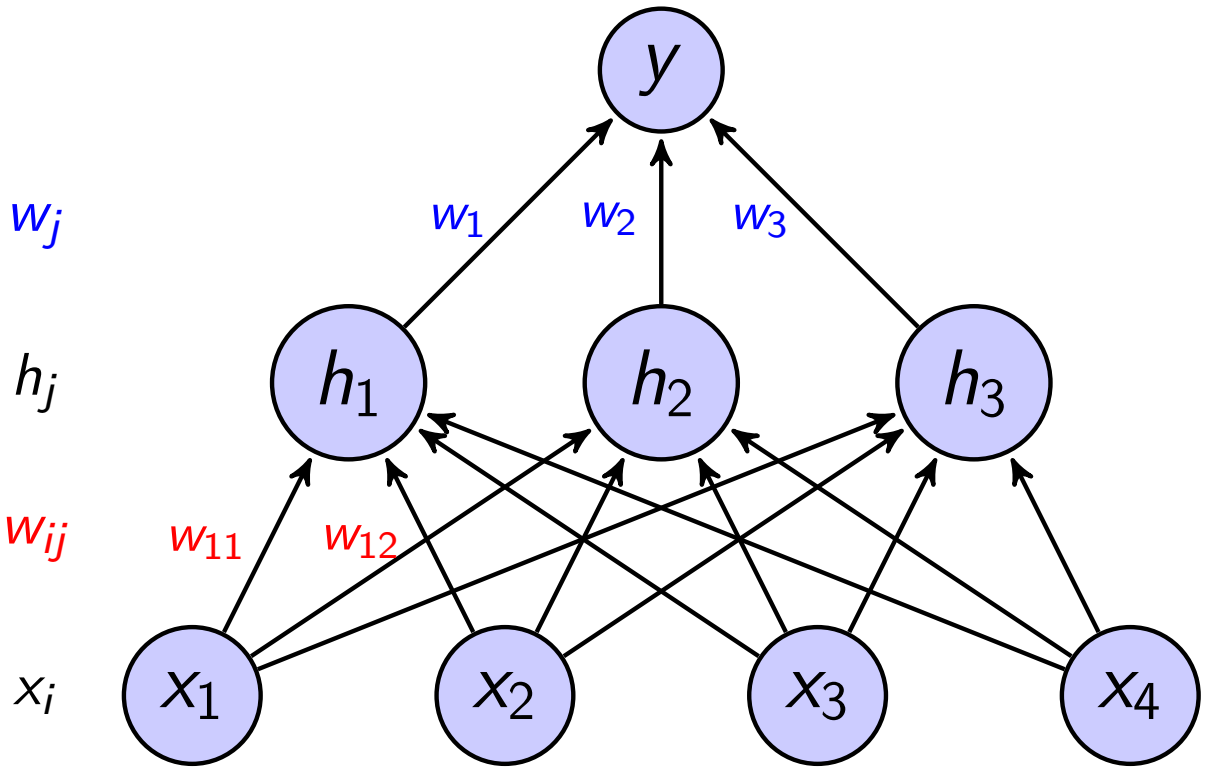
*An alternative is Cross-Entropy loss:
$\sum_m y^{(m)} \log(\sigma(w^T x^{(m)})) + (1 - y^{(m)}) \log(1 - \sigma(w^T x^{(m)}))$

SGD Pictorial View

- Loss objective contour plot: $\frac{1}{2}\sum_m(\sigma(w^T x^{(m)}) - y^{(m)})^2 + ||w||$
  - Gradient descent goes in steepest descent direction
  - SGD is noisy descent (but faster per iteration)



2-layer Neural Networks



$f(x) = \sigma(\sum_j w_j \cdot h_j) = \sigma(\sum_j w_j \cdot \sigma(\sum_i w_{ij} x_i))$

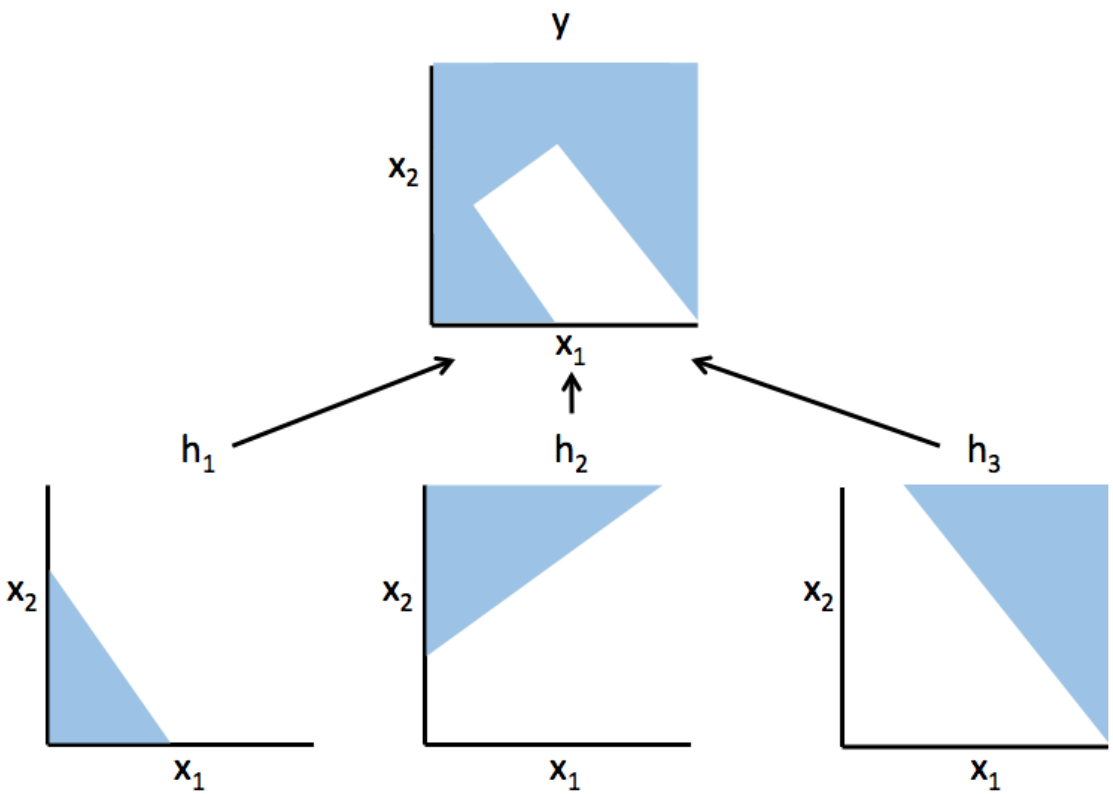
Hidden units h_j's can be viewed as new "features" from combining x_i's

---

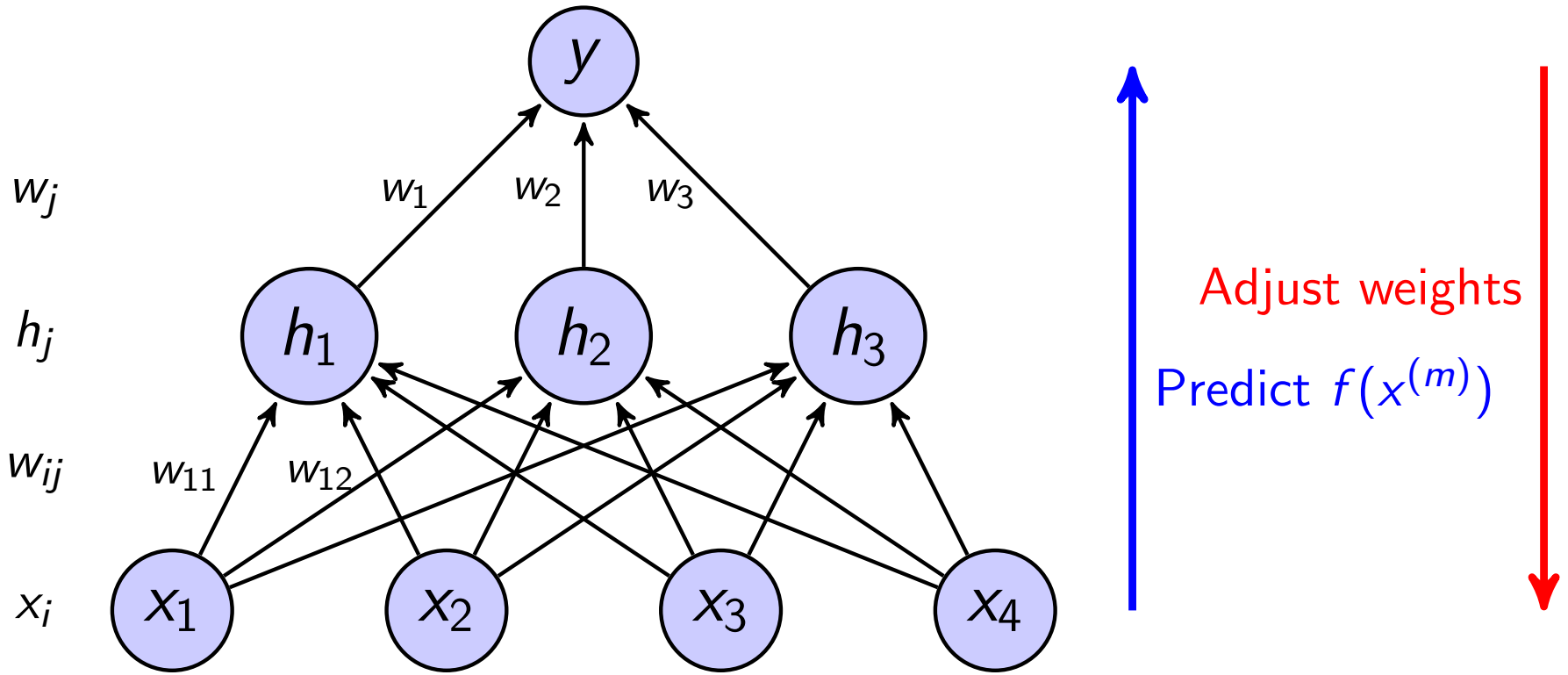
Called Multilayer Perceptron (MLP), but more like multilayer logistic regression

Expressive Power of Non-linearity

- A deeper architecture is more expressive than a shallow one given same number of nodes [Bishop, 1995]
  - 1-layer nets only model linear hyperplanes
  - 2-layer nets can model any continuous function (given sufficient nodes)
  - >3-layer nets can do so with fewer nodes



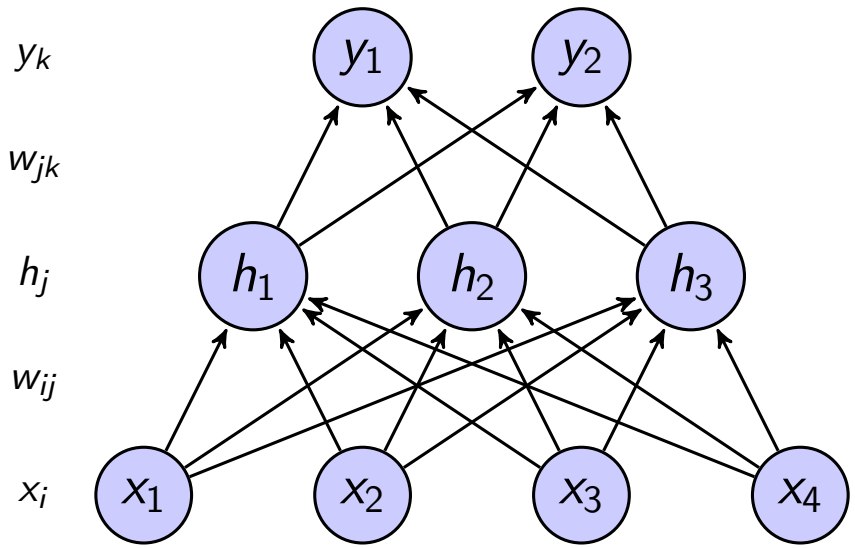
Training Neural Nets: Back-propagation



1. For each sample, compute $f(x^{(m)}) = \sigma(\sum_j w_j \cdot \sigma(\sum_i w_{ij} x_i^{(m)}))$
2. If $f(x^{(m)}) \neq y^{(m)}$, back-propagate error and adjust weights $\{w_{ij}, w_j\}$.

Derivatives of the weights

Assume two outputs (y_1, y_2) per input x,
and loss per sample: $Loss = \sum_k \frac{1}{2} [\sigma(in_k) - y_k]^2$



$$\frac{\partial Loss}{\partial w_{jk}} = \frac{\partial Loss}{\partial in_k} \frac{\partial in_k}{\partial w_{jk}} = \delta_k \frac{\partial (\sum_j w_{jk} h_j)}{\partial w_{jk}} = \delta_k h_j$$

$$\frac{\partial Loss}{\partial w_{ij}} = \frac{\partial Loss}{\partial in_j} \frac{\partial in_j}{\partial w_{ij}} = \delta_j \frac{\partial (\sum_j w_{ij} x_i)}{\partial w_{ij}} = \delta_j x_i$$

$$\delta_k = \frac{\partial}{\partial in_k} \left(\sum_k \frac{1}{2} [\sigma(in_k) - y_k]^2 \right) = [\sigma(in_k) - y_k] \, \sigma'(in_k)$$

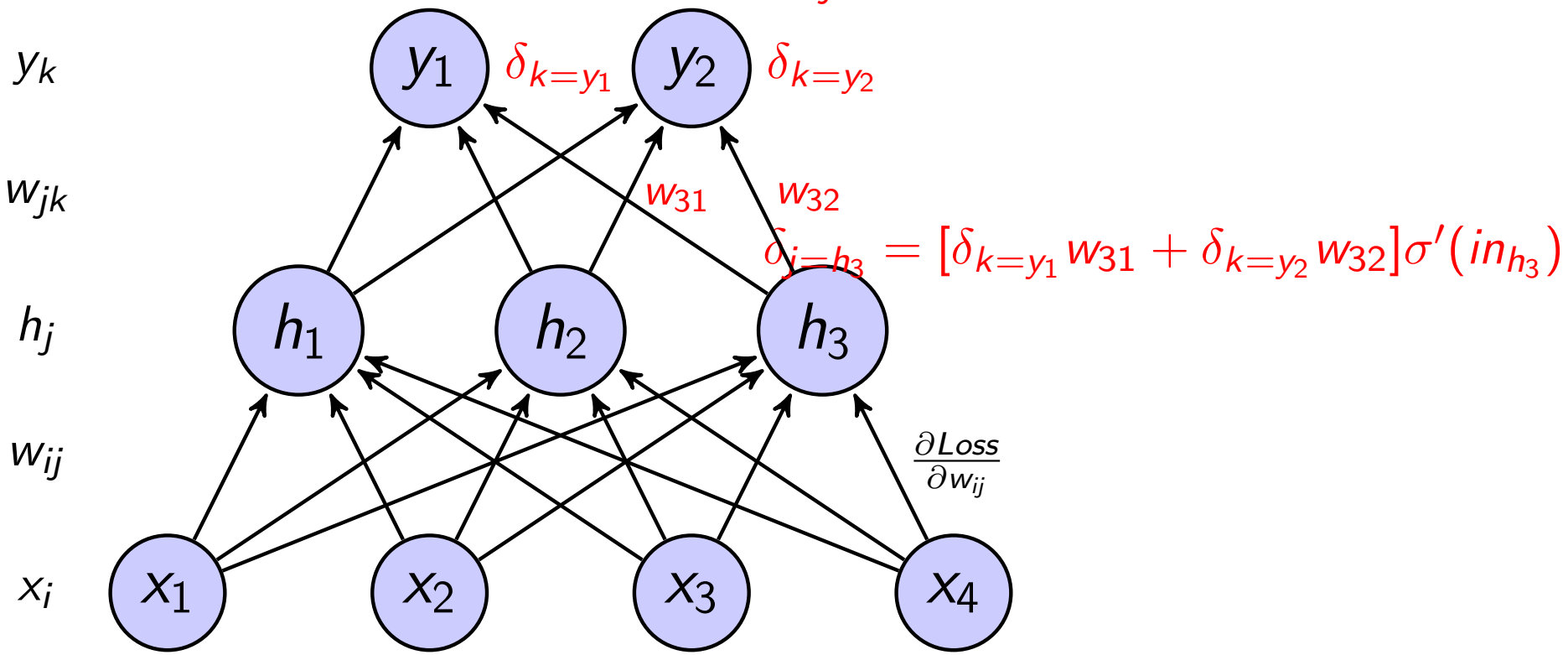
$$\delta_j = \sum_k \frac{\partial Loss}{\partial in_k} \frac{\partial in_k}{\partial in_j} = \sum_k \delta_k \cdot \frac{\partial}{\partial in_j} \left(\sum_j w_{jk} \sigma(in_j) \right) = [\sum_k \delta_k w_{jk}] \, \sigma'(in_j)$$

Training Neural Nets: Back-propagation

All updates involve some scaled error from output $*$ input feature:

- $\frac{\partial Loss}{\partial w_{jk}} = \delta_k h_j$ where $\delta_k = [\sigma(in_k) - y_k]\,\sigma'(in_k)$
- $\frac{\partial Loss}{\partial w_{ij}} = \delta_j x_i$ where $\delta_j = [\sum_k \delta_k w_{jk}]\,\sigma'(in_j)$

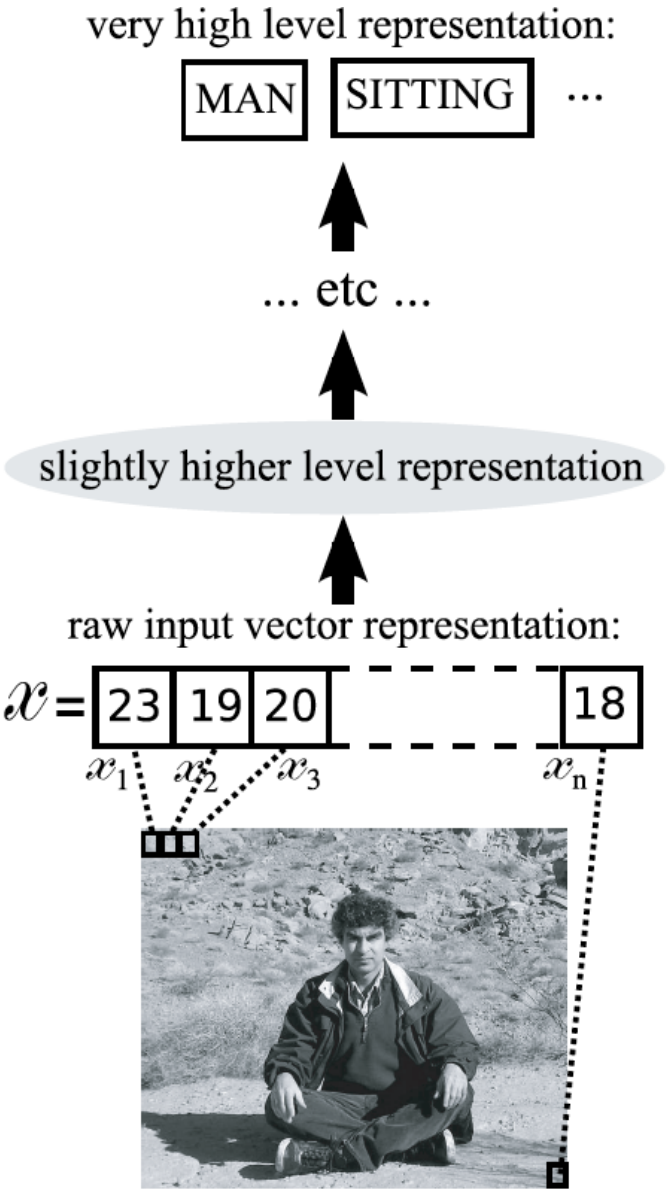
First compute δ_k from final layer, then δ_j for previous layer and iterate.



Outline

1. Deep Learning Background
   - Neural Networks (1-layer, 2-layer)
   - Potentials and Difficulties of Deep Architecture
   - The Breakthrough in 2006
2. Two Main Types of Deep Architectures
   - Deep Belief Nets (DBN) [Hinton et al., 2006]
   - Stacked Auto-Encoders (SAE) [Bengio et al., 2006]
   - Current Status of Deep Learning
3. Applications in Natural Language Processing and Machine Translation
   - Use as Non-linear classifier
   - Use as Distributed representation
   - Survey of Machine Translation Research

Potential of Deep Architecture



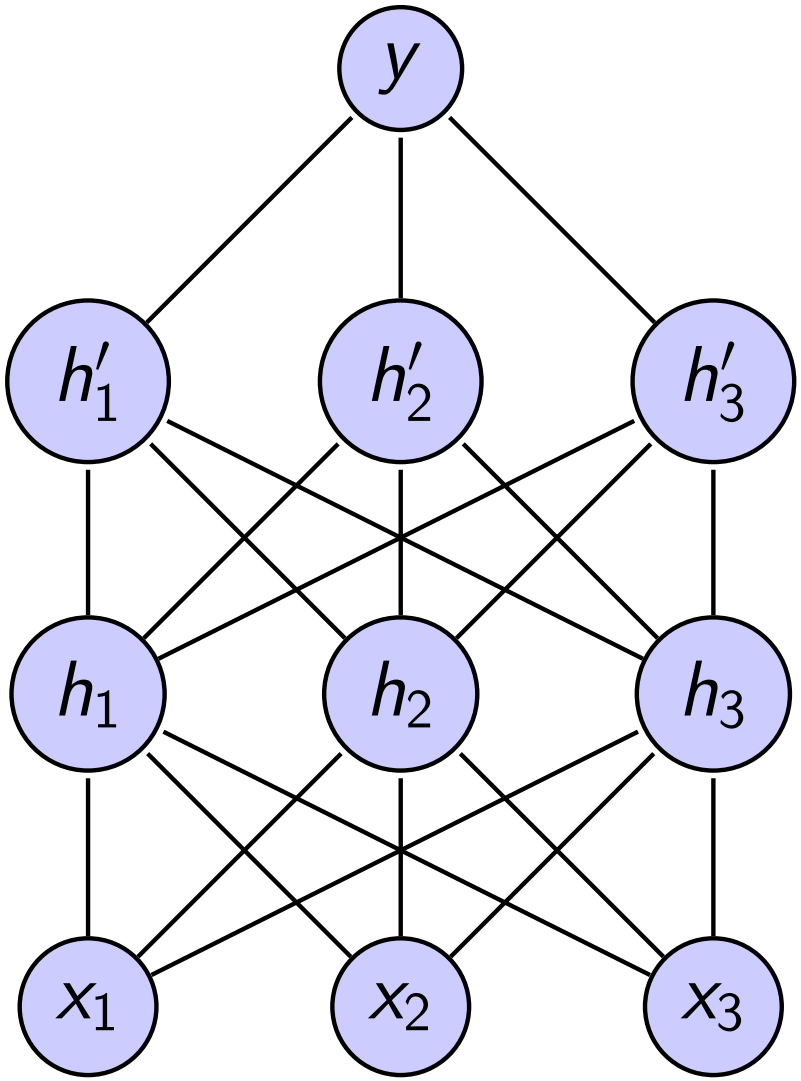
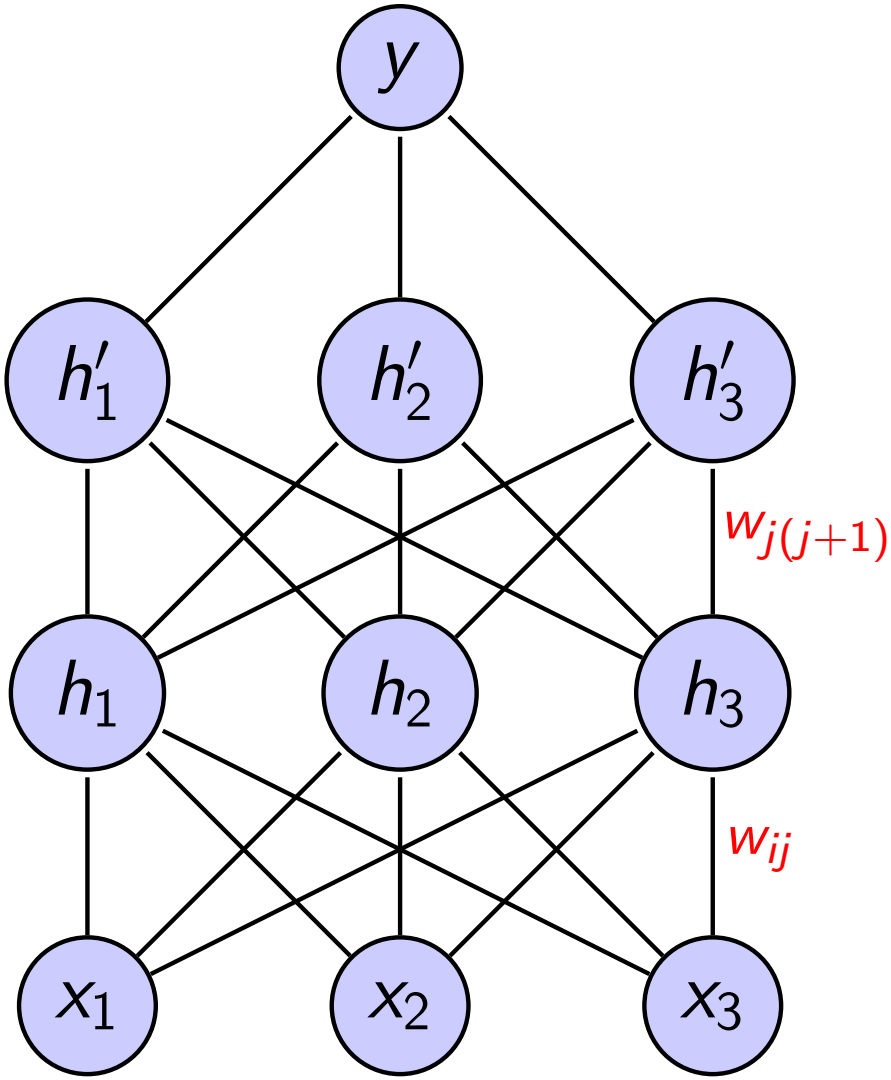


*Figure from [Bengio, 2009]

Difficulties of Deep Architecture

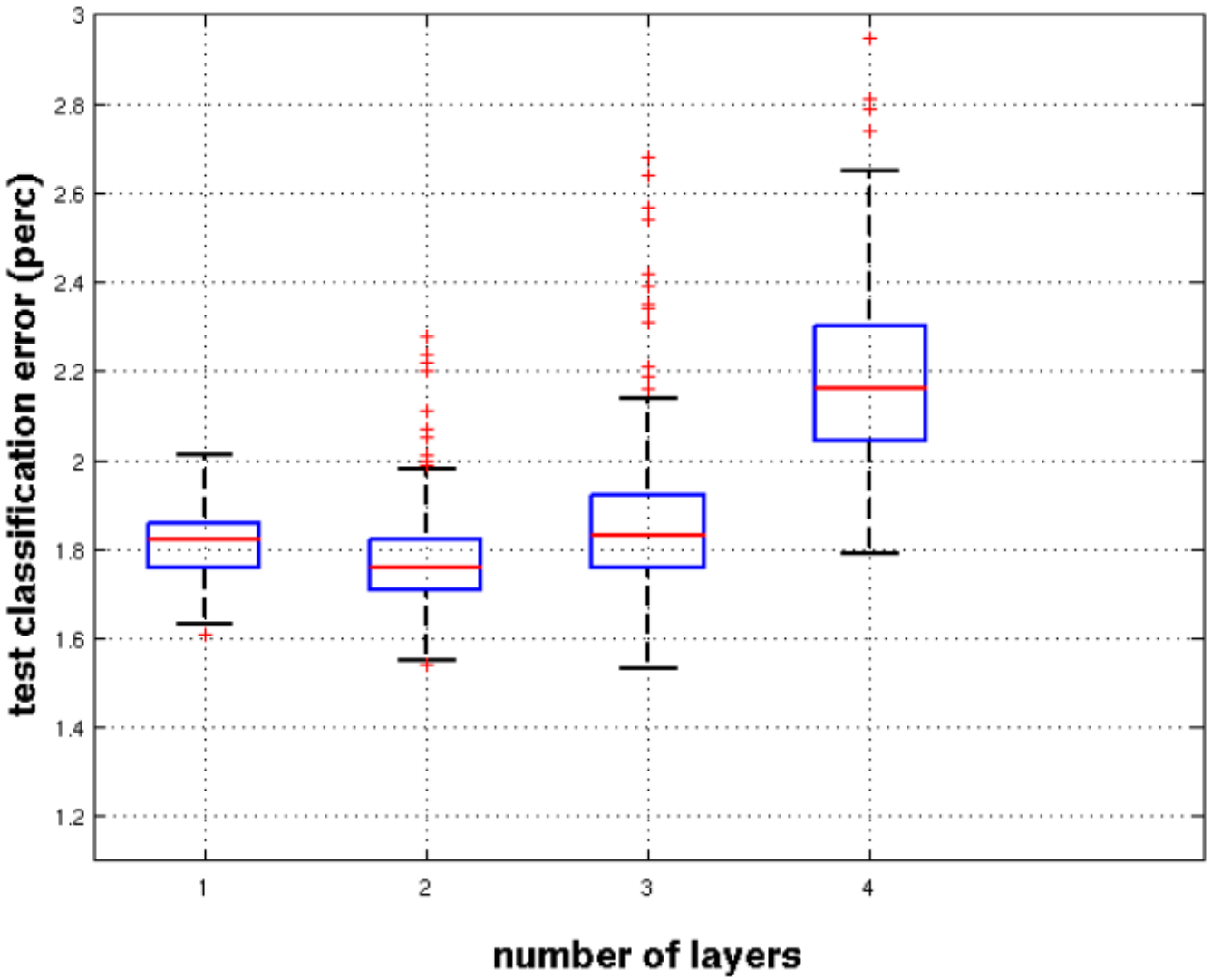
Vanishing gradient problem in Backpropagation

- $\frac{\partial Loss}{\partial w_{ij}} = \frac{\partial Loss}{\partial in_j}\frac{\partial in_j}{\partial w_{ij}} = \delta_j x_i$
- $\delta_j = \left[\sum_{j+1} \delta_{j+1} w_{j(j+1)}\right] \sigma'(in_j)$
- δ_j may vanish after repeated multiplication
- Also, exploding gradient problem!



Analysis of Training Difficulties [Erhan et al., 2009]

- MNIST digit classification task
- Train neural net by Backpropagation (random initialization of w_{ij})

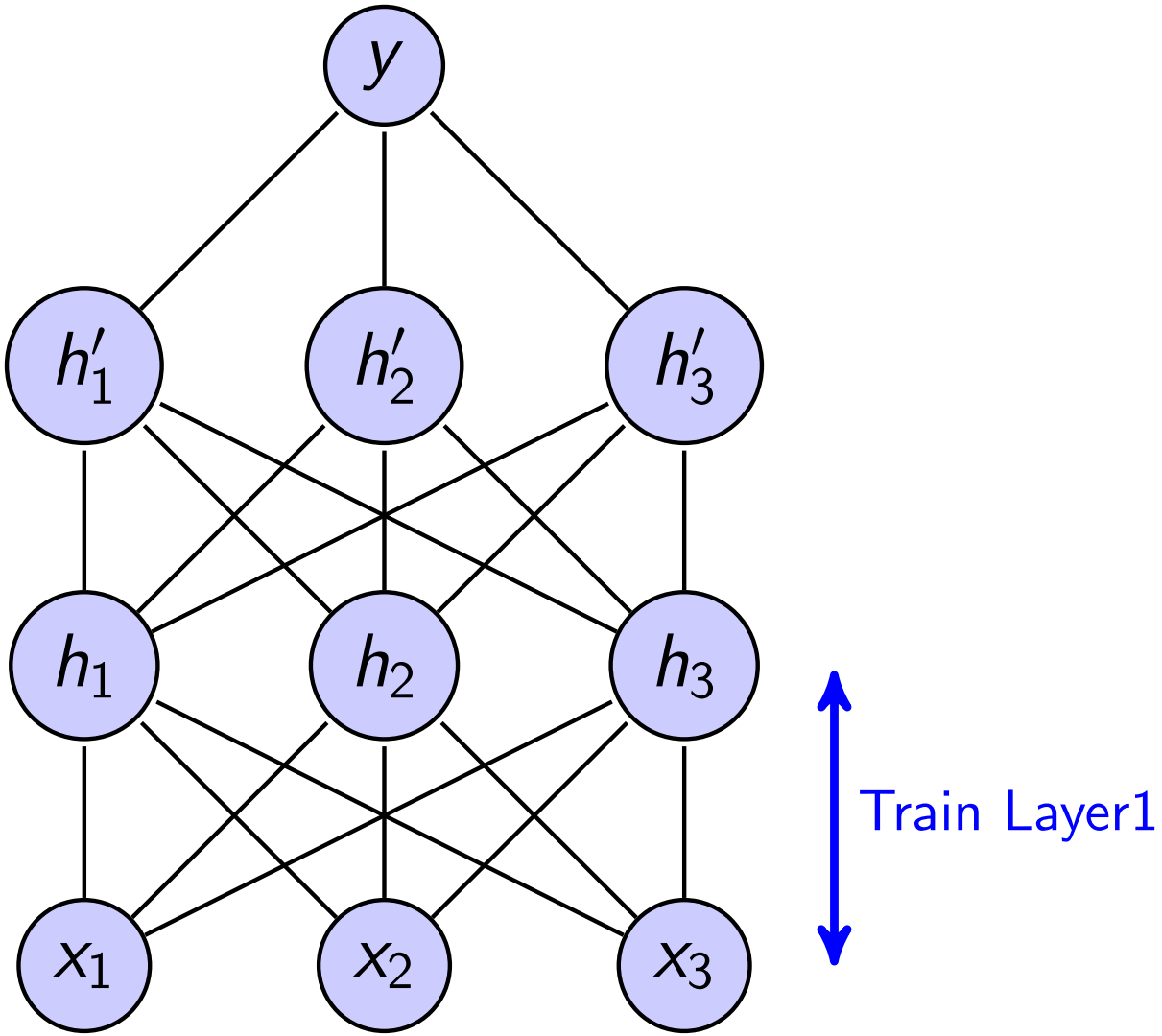


Outline

1. Deep Learning Background
   - Neural Networks (1-layer, 2-layer)
   - Potentials and Difficulties of Deep Architecture
   - The Breakthrough in 2006
2. Two Main Types of Deep Architectures
   - Deep Belief Nets (DBN) [Hinton et al., 2006]
   - Stacked Auto-Encoders (SAE) [Bengio et al., 2006]
   - Current Status of Deep Learning
3. Applications in Natural Language Processing and Machine Translation
   - Use as Non-linear classifier
   - Use as Distributed representation
   - Survey of Machine Translation Research

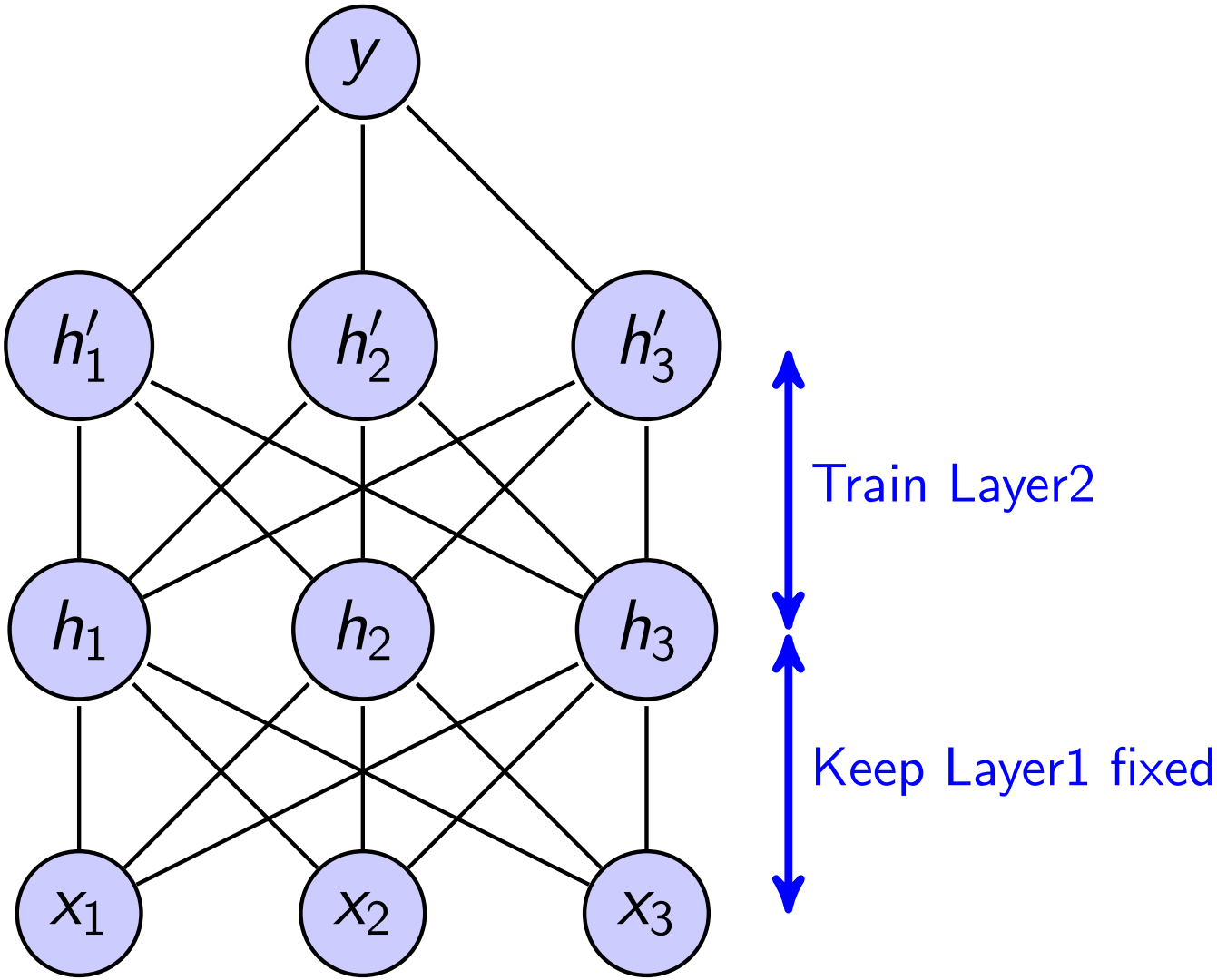
Layer-wise Pre-training [Hinton et al., 2006]

First, train one layer at a time, optimizing data-likelihood objective $P(x)$



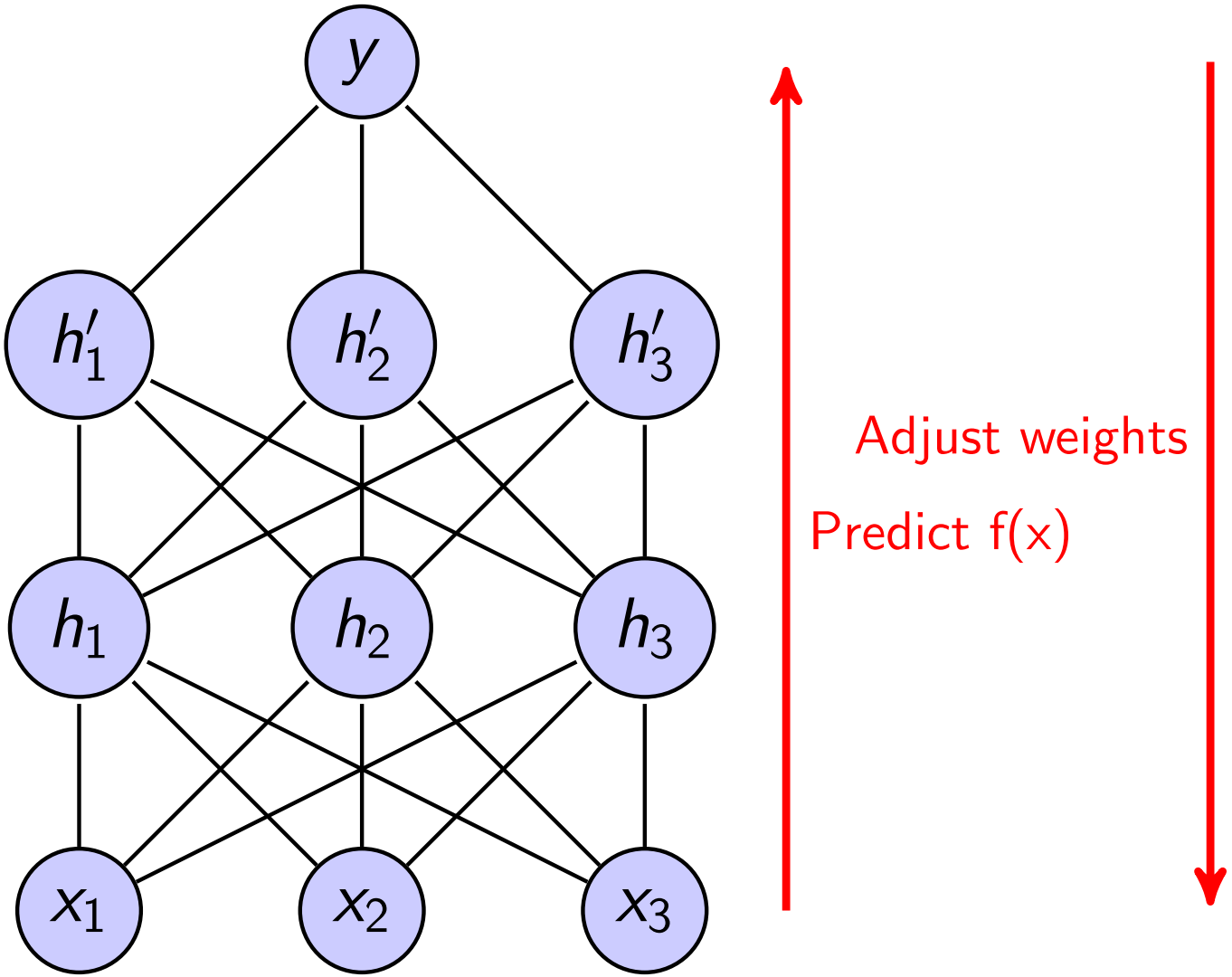
Layer-wise Pre-training [Hinton et al., 2006]

First, train one layer at a time, optimizing data-likelihood objective $P(x)$



Layer-wise Pre-training [Hinton et al., 2006]

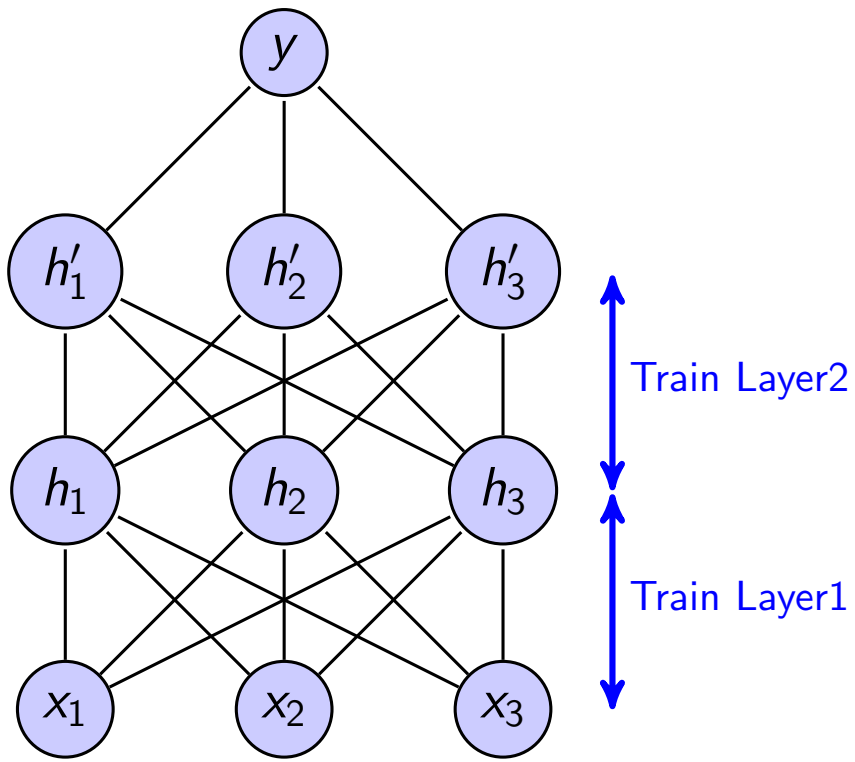
Finally, fine-tune labeled objective $P(y|x)$ by Backpropagation



Layer-wise Pre-training [Hinton et al., 2006]

Key Idea:
Focus on modeling the input $P(X)$ better with each successive layer.
Worry about optimizing the task $P(Y|X)$ later.

> *"If you want to do computer vision, first learn computer graphics." – Geoff Hinton*



Extra advantage:
Can exploit large amounts of unlabeled data!

Outline

1. Deep Learning Background
   - Neural Networks (1-layer, 2-layer)
   - Potentials and Difficulties of Deep Architecture
   - The Breakthrough in 2006

2. Two Main Types of Deep Architectures
   - Deep Belief Nets (DBN) [Hinton et al., 2006]
   - Stacked Auto-Encoders (SAE) [Bengio et al., 2006]
   - Current Status of Deep Learning

3. Applications in Natural Language Processing and Machine Translation
   - Use as Non-linear classifier
   - Use as Distributed representation
   - Survey of Machine Translation Research

Deep Learning Paradigm

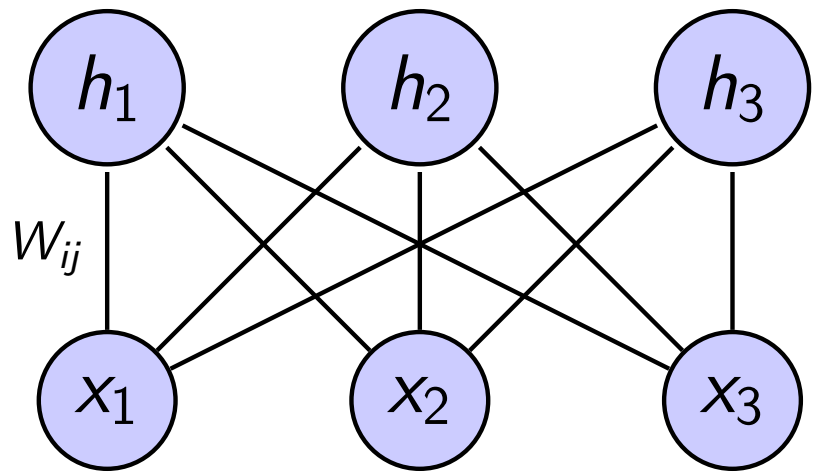
- Recall problem setup: Learn function $f : x \rightarrow y$
- First learn hidden features h that model input, i.e. $x \rightarrow h \rightarrow y$
- How do we discover useful latent features h from data x?
  - e.g. use Restricted Boltzmann Machines (RBMs)

Restricted Boltzmann Machine (RBM)

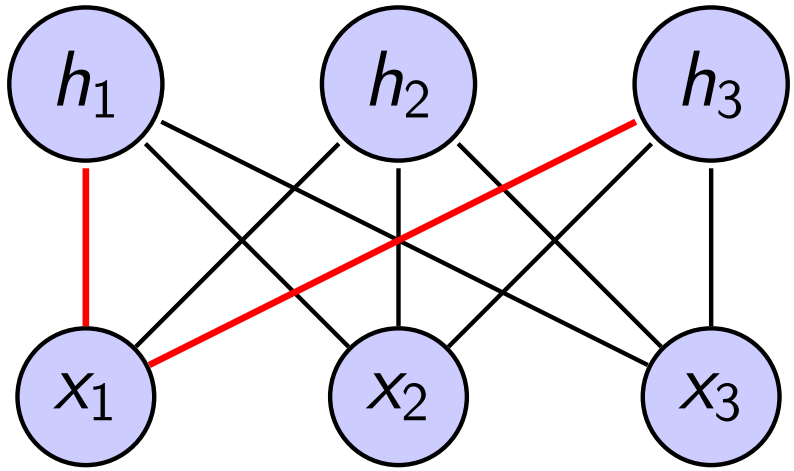
- RBM is a probabilistic model on binary variables h_j and x_i:

$$\begin{aligned} p(x,h) &= \frac{1}{Z_\theta}\exp\left(-E_\theta(x,h)\right) \\ &= \frac{1}{Z_\theta}\exp\left(x^T W h + b^T x + d^T h\right) \end{aligned}$$

  - W is a matrix; elements W_{ij} models correlation between x_i and h_j
  - b and d are bias terms; we'll assume $b = d = 0$ here.
  - normalizer (partition function): $Z_\theta = \sum_{(x,h)} \exp(-E_\theta(x,h))$



Restricted Boltzmann Machine (RBM): Example



Let weights W_{ij} on edges $(h_1, x_1), (h_1, x_3)$ be positive, others be near 0.

x_1	x_2	x_3	h_1	h_2	h_3	$p(x,h) = \frac{1}{Z_\theta} \exp(x^T W h)$
1	0	0	1	0	1	highest
1	0	0	0	0	1	high
1	0	0	1	0	0	high
0	0	0	1	0	1	low
0	1	0	0	0	0	low
0	0	1	0	0	0	low

etc

RBM Posterior Distribution (Easy!)

- Computing $p(h|x)$ is easy due to factorization:

$$
\begin{aligned}
p(h|x) &= \frac{p(x,h)}{\sum_h p(x,h)} = \frac{1/Z_\theta \exp(-\mathrm{E}(\mathrm{x},\mathrm{h}))}{\sum_h 1/Z_\theta \exp(-\mathrm{E}(\mathrm{x},\mathrm{h}))} \\
&= \frac{\exp(x^T W h + b^T x + d^T h)}{\sum_h \exp(x^T W h + b^T x + d^T h)} \\
&= \frac{\prod_j \exp(x^T W_j h_j + d_j h_j) \cdot \exp(b^T x)}{\sum_{h_1 \in \{0,1\}} \sum_{h_2 \in \{0,1\}} \cdots \sum_{h_j} \prod_j \exp(x^T W_j h_j + d_j h_j) \cdot \exp(b^T x)} \\
&= \frac{\prod_j \exp(x^T W_j h_j + d_j h_j)}{\prod_j \sum_{h_j \in \{0,1\}} \exp(x^T W_j h_j + d_j h_j)} \\
&= \prod_j \frac{\exp(x^T W_j h_j + d_j h_j)}{\sum_{h_j \in \{0,1\}} \exp(x^T W_j h_j + d_j h_j)} = \prod_j p(h_j|x)
\end{aligned}
$$

- $p(h_j = 1|x) = \exp(x^T W_j + d_j)/Z = \sigma(x^T W_j + d_j)$ is Logistic Regression!
- Similarly, computing $p(x|h) = \prod_i p(x_i|h)$ is easy

RBM Max-Likelihood Training (Hard!)

Derivative of the Log-Likelihood: $\partial_{w_{ij}} \log P_w(x = x^{(m)})$

$$= \partial_{w_{ij}} \log \sum_h P_w(x = x^{(m)}, h) \tag{1}$$

$$= \partial_{w_{ij}} \log \sum_h \frac{1}{Z_w} \exp\left(-E_w(x^{(m)}, h)\right) \tag{2}$$

$$= -\partial_{w_{ij}} \log Z_w + \partial_{w_{ij}} \log \sum_h \exp\left(-E_w(x^{(m)}, h)\right) \tag{3}$$

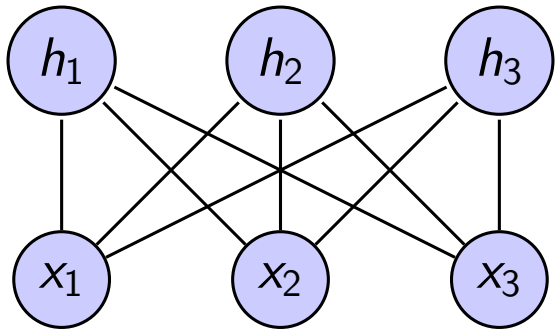
$$= \frac{1}{Z_w} \sum_{h,x} e^{(-E_w(x,h))} \partial_{w_{ij}} E_w(x, h) - \frac{1}{\sum_h e^{(-E_w(x^{(m)},h))}} \sum_h e^{(-E_w(x^{(m)},h))} \partial_{w_{ij}} E_w(x^{(m)}, h)$$

$$= \sum_{h,x} P_w(x, h)[\partial_{w_{ij}} E_w(x, h)] - \sum_h P_w(x^{(m)}, h)[\partial_{w_{ij}} E_w(x^{(m)}, h)] \tag{4}$$

$$= -\mathbb{E}_{p(x,h)}[x_i \cdot h_j] + \mathbb{E}_{p(h|x=x^{(m)})}[x_i^{(m)} \cdot h_j] \tag{5}$$

Second term (positive phase) increases probability of $x^{(m)}$; First term (negative phase) decreases probability of samples generated by the model

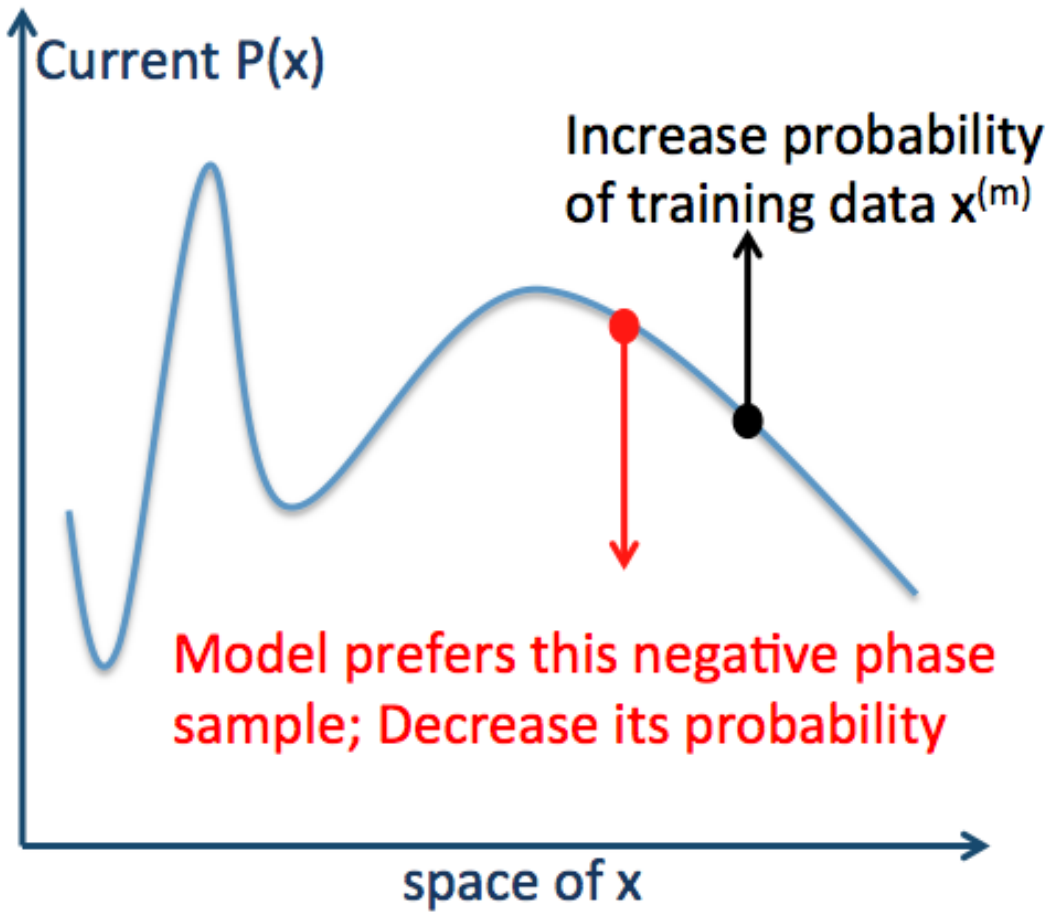
Contrastive Divergence Algorithm



- The negative phase term ($\mathbb{E}_{p(x,h)}[x_i \cdot h_j]$) is expensive because it requires sampling (x,h) from the model
- Gibbs Sampling (sample x then h iteratively) works, but waiting for convergence at each gradient step is slow.
- Contrastive Divergence is a faster but biased method: initialize with training point and wait only a few (usu. 1) sampling steps
  1. Let $x^{(m)}$ be training point, $W = [w_{ij}]$ be current model weights
  2. Sample $\hat{h}_j \in \{0, 1\}$ from $p(h_j | x = x^{(m)}) = \sigma(\sum_i w_{ij} x_i^{(m)} + d_j)\ \forall j$.
  3. Sample $\tilde{x}_i \in \{0, 1\}$ from $p(x_i | h = \hat{h}) = \sigma(\sum_j w_{ij} \hat{h}_j + b_i)\ \forall i$.
  4. Sample $\tilde{h}_j \in \{0, 1\}$ from $p(h_j | x = \tilde{x}) = \sigma(\sum_i w_{ij} \tilde{x}_i + d_j)\ \forall j$.
  5. $w_{ij} \leftarrow w_{ij} + \gamma(x_i^{(m)} \cdot \hat{h}_j - \tilde{x}_i \cdot \tilde{h}_j)$

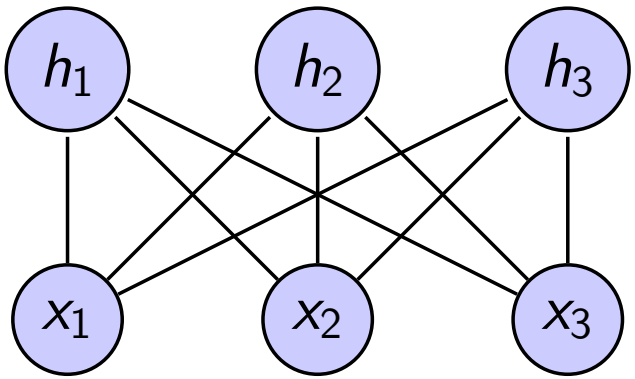
Contrastive Divergence Pictorial View

- Goal: Make RBM $p(x, h)$ have high probability on training samples
- To do so, we'll "steal" probability mass from nearby samples that incorrectly preferred by the model
- For detailed analysis, see [Carreira-Perpinan and Hinton, 2005]

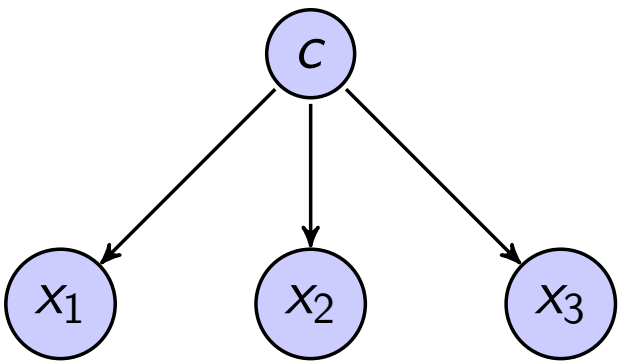


Distributed Representations learned by RBM

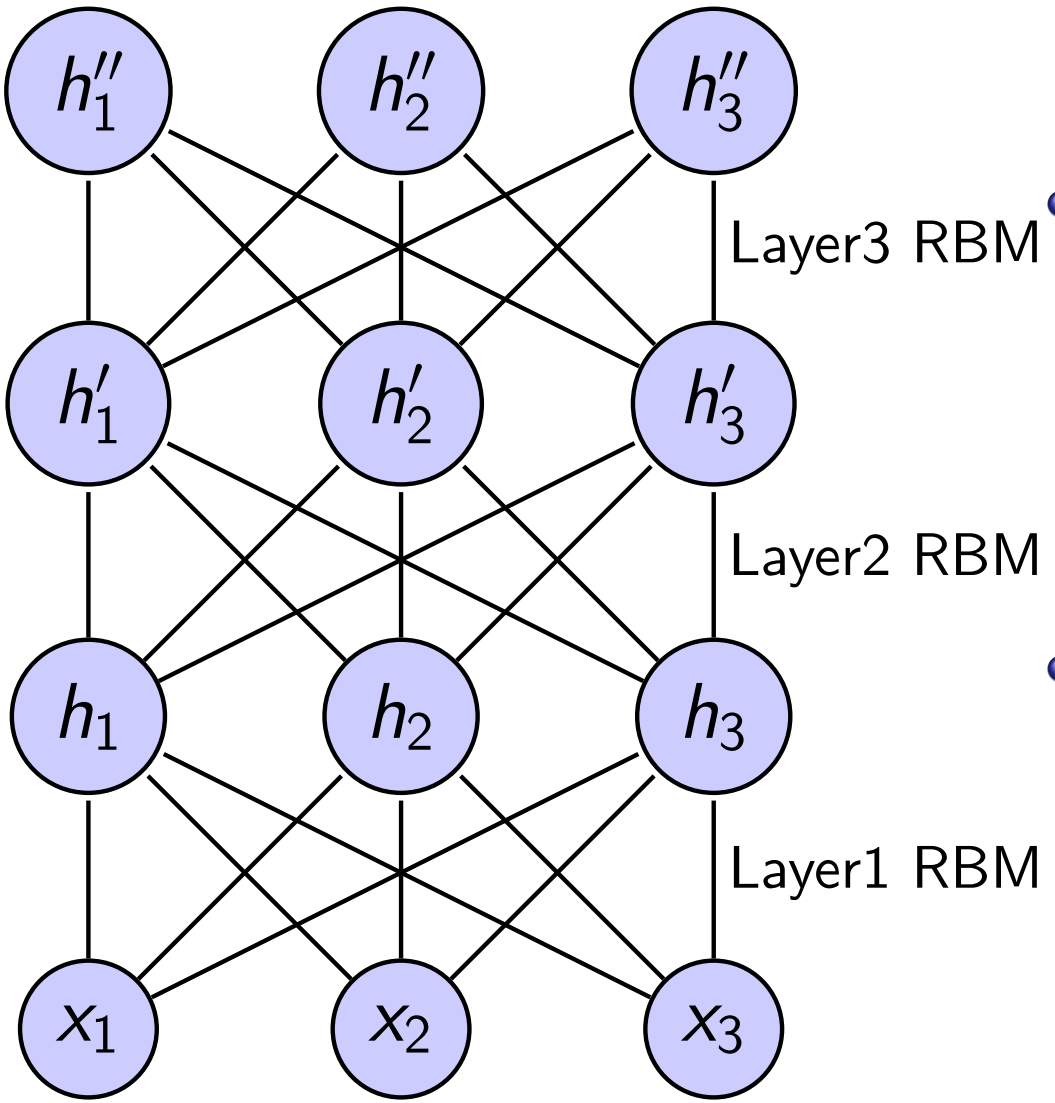
- Vector h act as **Distributed Representation** of data
  - Multiple h_j may be active simultaneously for a given x. (Multi-clustering)
  - $2^{|h|}$ possible representations with $|h| \times |x|$ parameters.



- An equivalent mixture model $p(x) = \sum_h p(c)p(x|c)$ needs $2^{|h|} \times |x|$ parameters:



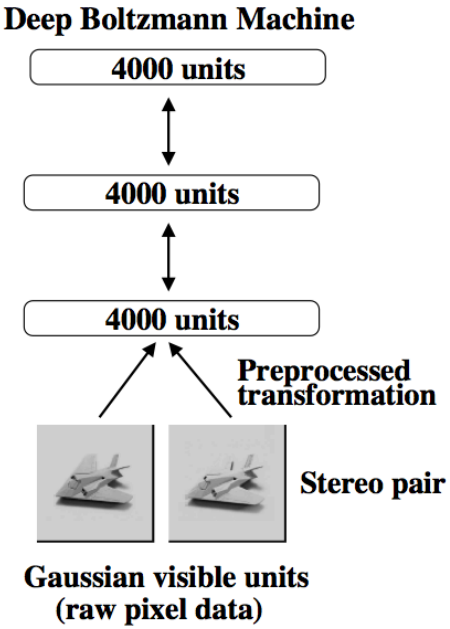
Deep Belief Nets (DBN) = Stacked RBM

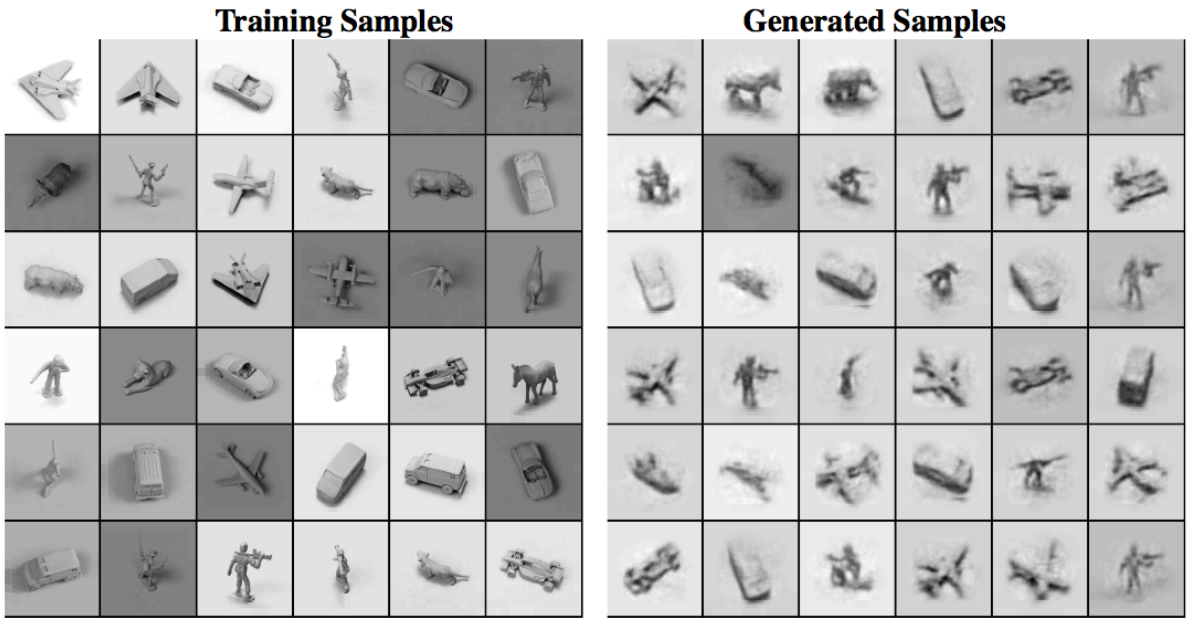


- DBN defines a probabilistic generative model $p(x) = \sum_{h,h',h''} p(x|h)p(h|h')p(h',h'')$ (top 2 layers is interpreted as a RBM; lower layers are directed sigmoids)
- Stacked RBMs can also be used to initialize a Deep Neural Network (DNN)

Example of what a Deep Generative Model can do

After training on 20k images, the generative model of [Salakhutdinov and Hinton, 2009]* can generate random images (dimension=8976) that are amazingly realistic!





This model is a Deep Boltzmann Machine (DBM), different from Deep Belief Nets (DBN) but also built by stacking RBMs.

Deep Belief Nets (DBN) Summary

1. Layer-wise pre-training is the innovation that rekindled interest in deep architectures.
2. Pre-training focuses on optimizing likelihood on the data, not the target label. First model $p(x)$ to do better $p(y|x)$.
3. Why RBM? $p(h|x)$ is tractable, so it's easy to stack.
4. RBM training can be expensive. Solution: contrastive divergence
5. We can stack RBMs to form a deep probabilistic generative model (DBN), or to initialize deep neural network (DNN)

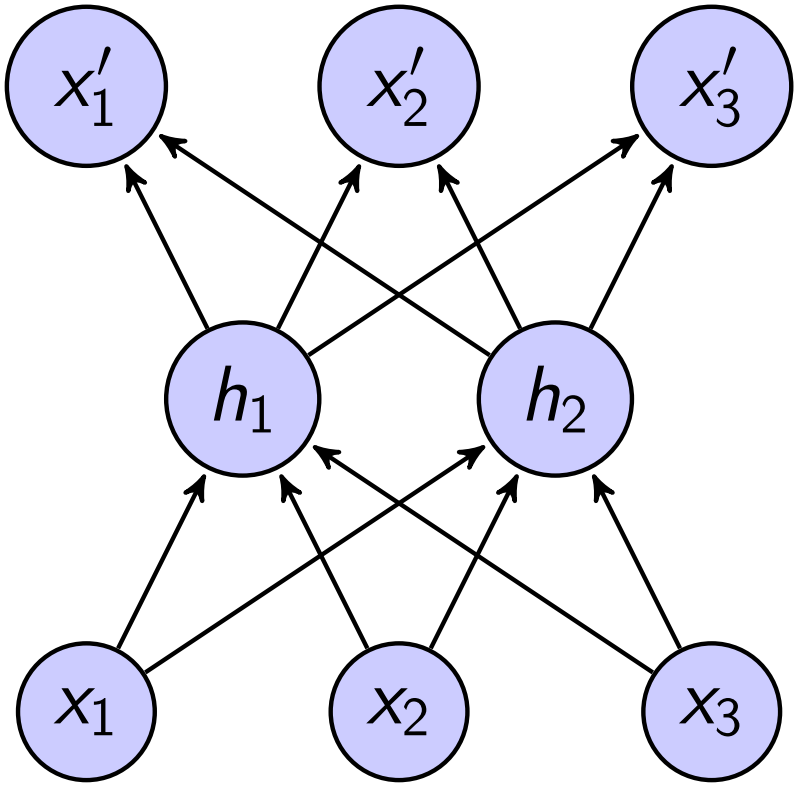
Outline

1. Deep Learning Background
   - Neural Networks (1-layer, 2-layer)
   - Potentials and Difficulties of Deep Architecture
   - The Breakthrough in 2006

2. Two Main Types of Deep Architectures
   - Deep Belief Nets (DBN) [Hinton et al., 2006]
   - Stacked Auto-Encoders (SAE) [Bengio et al., 2006]
   - Current Status of Deep Learning

3. Applications in Natural Language Processing and Machine Translation
   - Use as Non-linear classifier
   - Use as Distributed representation
   - Survey of Machine Translation Research

Auto-Encoders: Efficient replacement for RBM



Decoder: $x' = \sigma(W'h + d)$

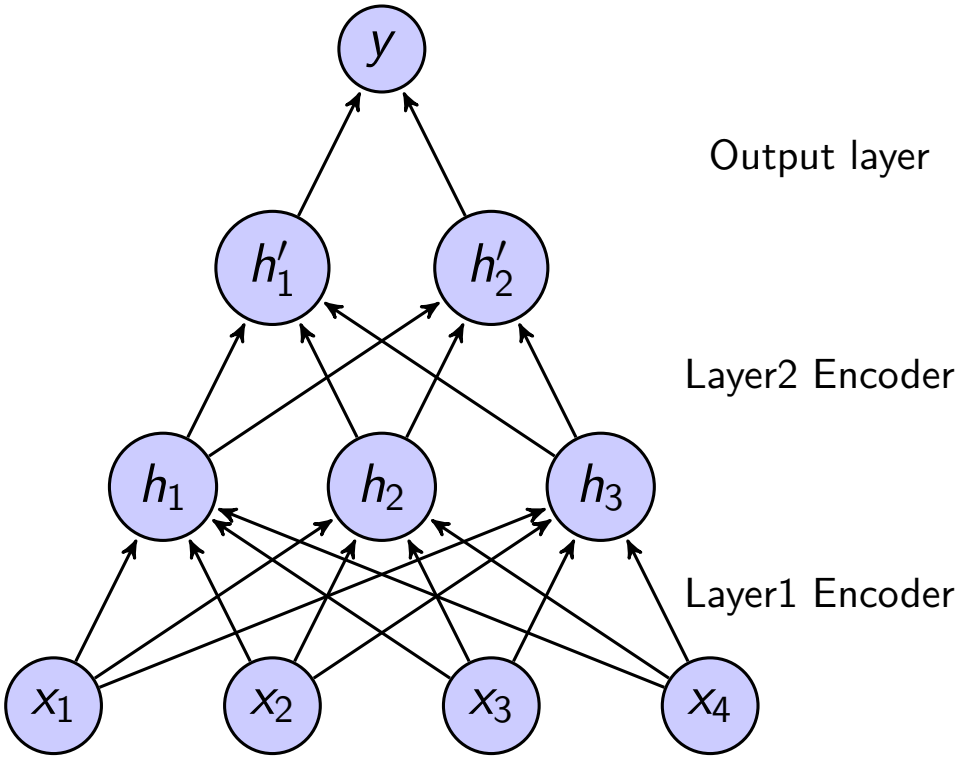
Encoder: $h = \sigma(Wx + b)$

Encourage h to give small reconstruction error:

- e.g. $Loss = \sum_m ||x^{(m)} - DECODER(ENCODER(x^{(m)}))||^2$
- Reconstruction: $x' = \sigma(W'\sigma(Wx + b) + d)$
- $|h|$ is small to enforce "compression" of data
- Training by Backpropagation for 2-layer nets, with $x^{(m)}$ as both input and output

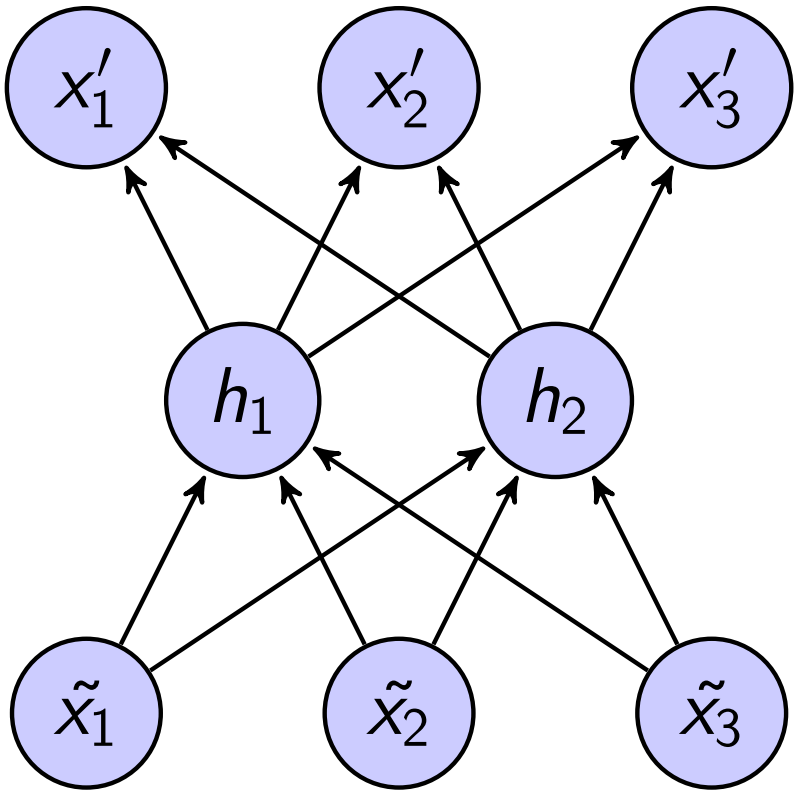
Stacked Auto-Encoders (SAE)

- The encoder/decoder gives same form $p(h|x)$, $p(x|h)$ as RBMs, so can be stacked in the same way to form Deep Architectures



- Unlike RBMs, Auto-encoders are deterministic.
  - $h = \sigma(Wx + b)$, not $p(h = \{0, 1\}) = \sigma(Wx + b)$
  - Disadvantage: Can't form deep generative model
  - Advantage: Fast to train, and useful still for Deep Neural Nets

Auto-Encoder Variants: e.g. Denoising Auto-Encoder



Decoder: $x' = \sigma(W'h + d)$

Encoder: $h = \sigma(W\tilde{x} + b)$

$\tilde{x} = x +$ noise

1. Perturb input data x to $\tilde{x}$ using invariance from domain knowledge.
2. Train weights to reduce reconstruction error with respect to original input: $||x - x'||$

Denoising Auto-Encoders

- Example: Randomly shift, rotate, and scale input image
- An image of "2" is a "2" no matter how you add noise, so the auto-encoder will try to cancel the variations that are not important.

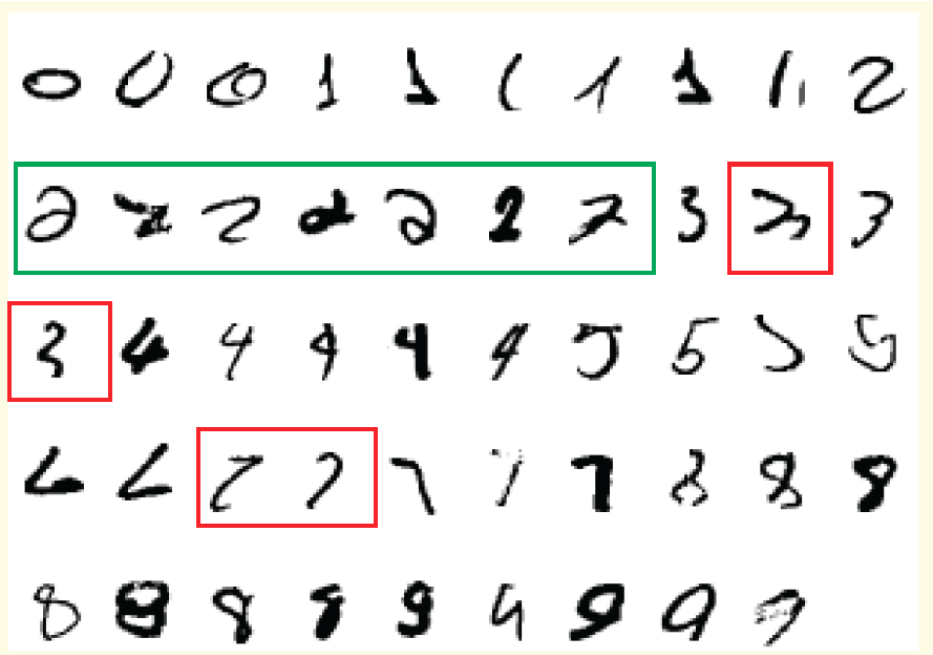


Figure from Geoff Hinton's 2012 Coursera course, lecture 1: `https://www.coursera.org/course/neuralnets`

Stacked Auto-Encoders (SAE): Summary

1. Auto-Encoders are cheaper alternatives to RBMs.
   - Not probabilistic, but fast to train using Backpropagation
   - Achieves similar accuracies as RBM [Bengio et al., 2006]
2. Auto-Encoders learn to "compress" and "re-construct" input data. Again, the focus is on modeling $p(x)$ first.
3. Many variants, some provide ways to incorporate domain knowledge.

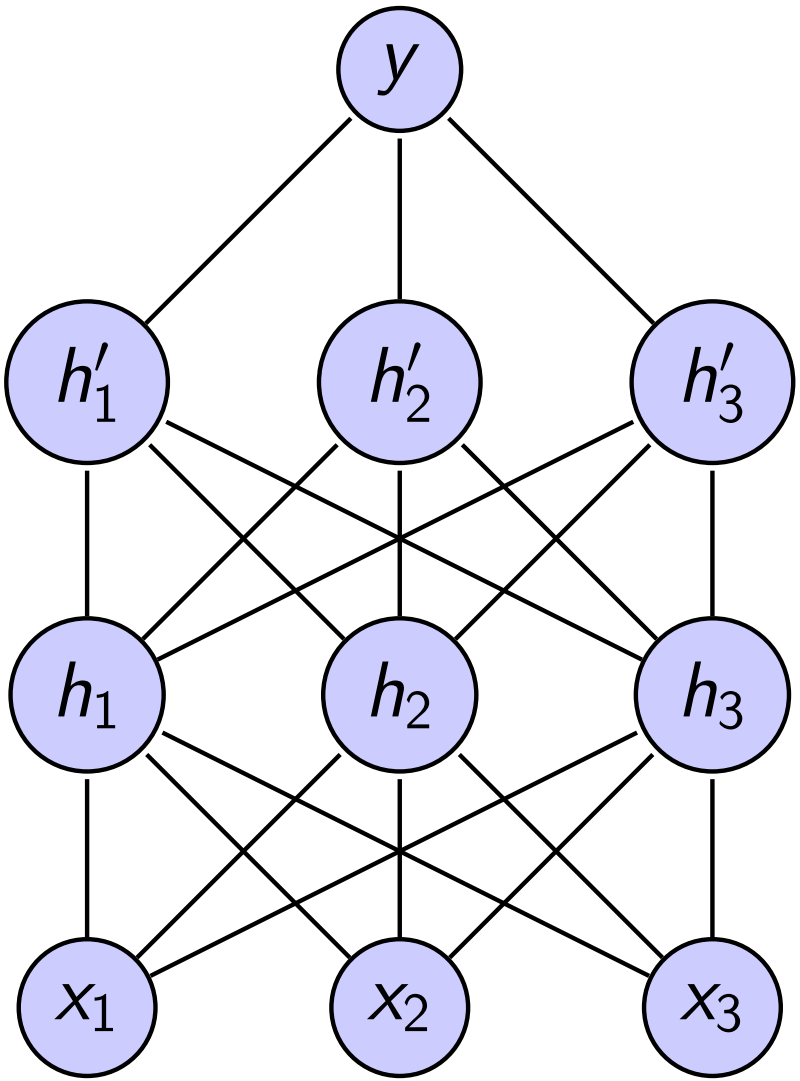
Outline

1. Deep Learning Background
   - Neural Networks (1-layer, 2-layer)
   - Potentials and Difficulties of Deep Architecture
   - The Breakthrough in 2006

2. Two Main Types of Deep Architectures
   - Deep Belief Nets (DBN) [Hinton et al., 2006]
   - Stacked Auto-Encoders (SAE) [Bengio et al., 2006]
   - Current Status of Deep Learning

3. Applications in Natural Language Processing and Machine Translation
   - Use as Non-linear classifier
   - Use as Distributed representation
   - Survey of Machine Translation Research

Why does Pre-Training work? [Erhan et al., 2010]



- A deep net can fit the training data in many ways (non-convex):
  1. By optimizing upper-layers really hard
  2. By optimizing lower-layers really hard
- Top-down vs. Bottom-up information
  1. Even if lower-layers are random weights, upper-layer may still fit well. But may not generalize to new data
  2. Pre-training with objective on $P(x)$ learns more generalizable features
- Pre-training seems to help put weights at a better local optimum

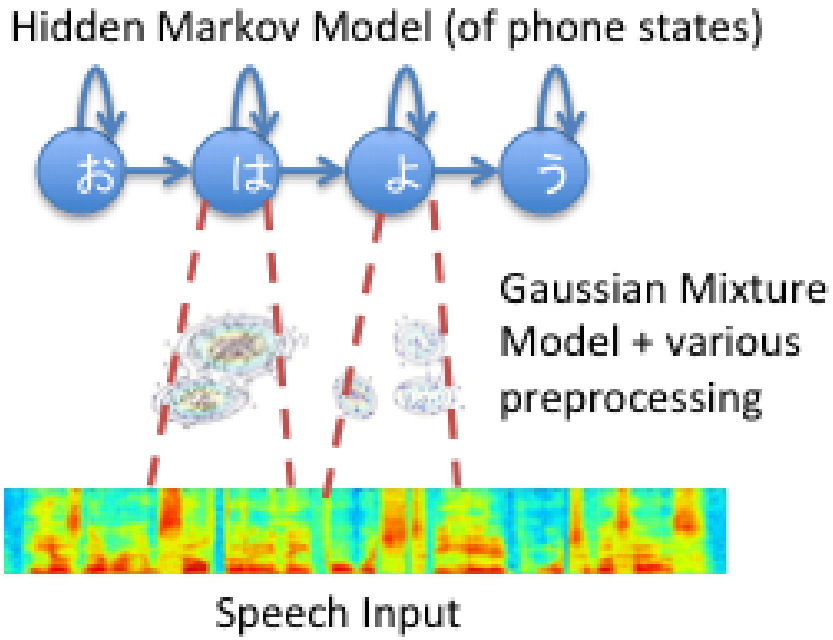
Is Pre-Training really necessary?

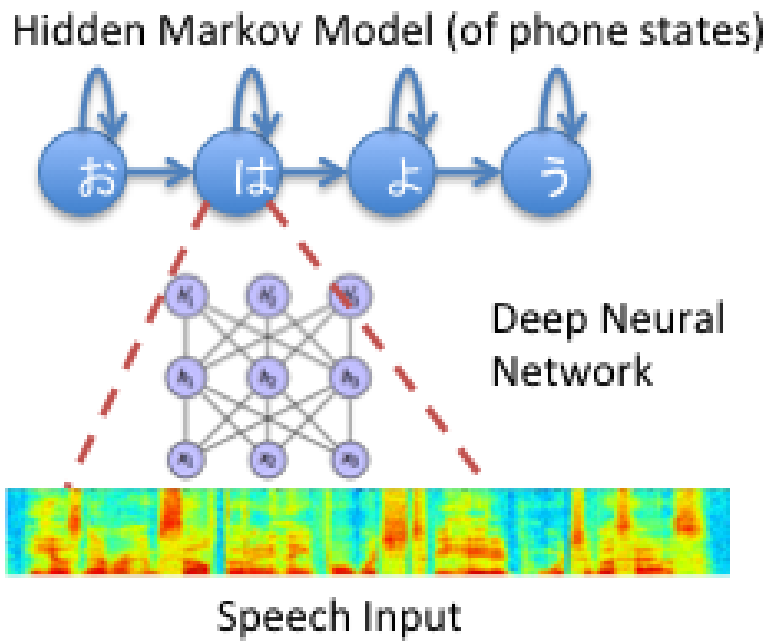
Answer in 2006: Yes!
Answer in 2014: No!

1. If initialization is done well by design (e.g. sparse connections and convolutional nets), maybe won't have vanishing gradient problem
2. If you have an extremely large datasets, maybe won't overfit. (But maybe that also means you want an ever deeper net)
3. New architectures are emerging: e.g. Stacked SVM's with random projections [Vinyals et al., 2012]

Success in Speech Recognition

Hybrid DNN-HMM system: (typically 3-8 layers, 2000 units/layer, 15 frames of input, 6000 output)





Success in Speech Recognition

Word Error Rate Results [Hinton et al., 2012a]:

TASK	HOURS OF TRAINING DATA	DNN-HMM	GMM-HMM WITH SAME DATA	GMM-HMM WITH MORE DATA
SWITCHBOARD (TEST SET 1)	309	18.5	27.4	18.6 (2,000 H)
SWITCHBOARD (TEST SET 2)	309	16.1	23.6	17.1 (2,000 H)
ENGLISH BROADCAST NEWS	50	17.5	18.8	
BING VOICE SEARCH (SENTENCE ERROR RATES)	24	30.4	36.2	
GOOGLE VOICE INPUT	5,870	12.3		16.0 (>> 5,870 H)
YOUTUBE	1,400	47.6	52.3	

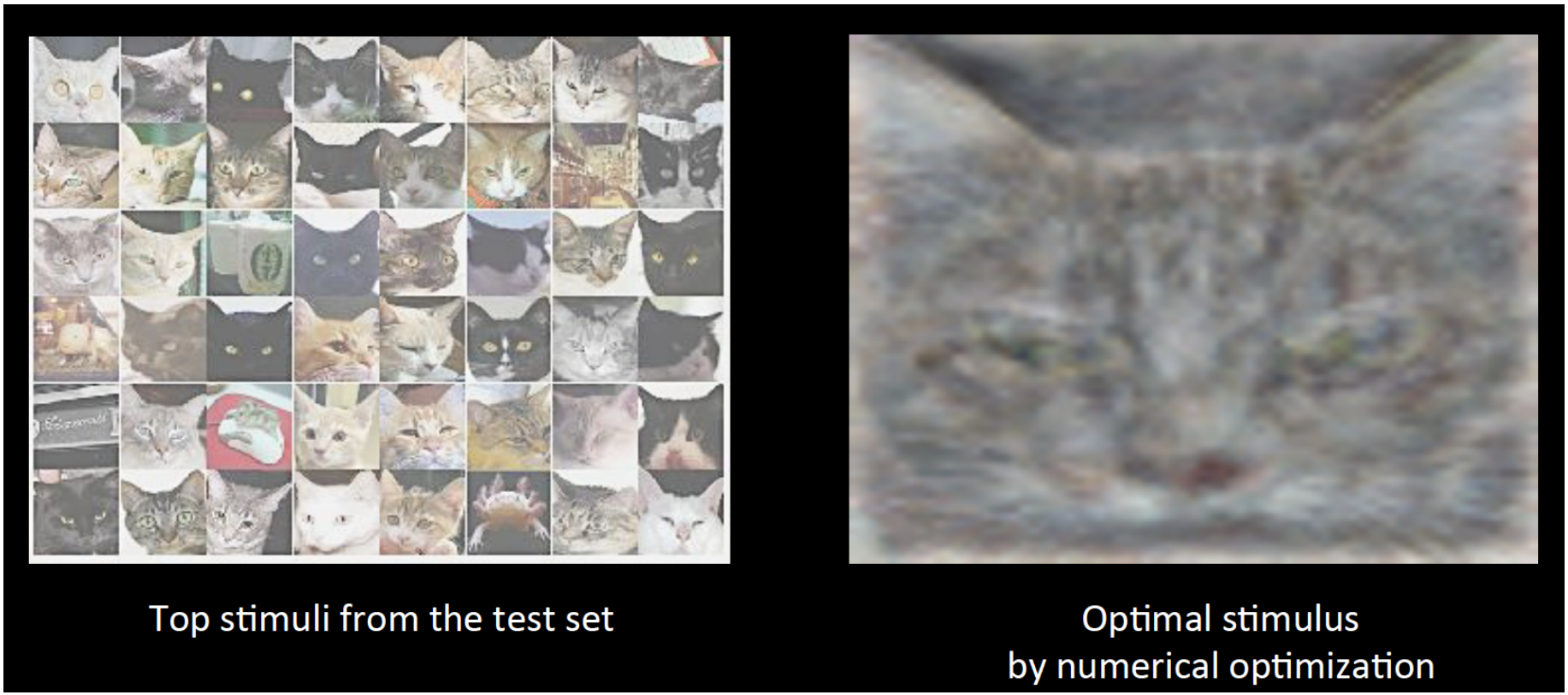
Success in Computer Vision [Le et al., 2012]

ImageNet Test Accuracy (22K categories):

Method	Accuracy
Random	0.005%
Previous State-of-the-art	9.3%
"9"-layer net, back-propagation without pre-training	13.6%
+ pre-training on 10 million Youtube images	15.8%

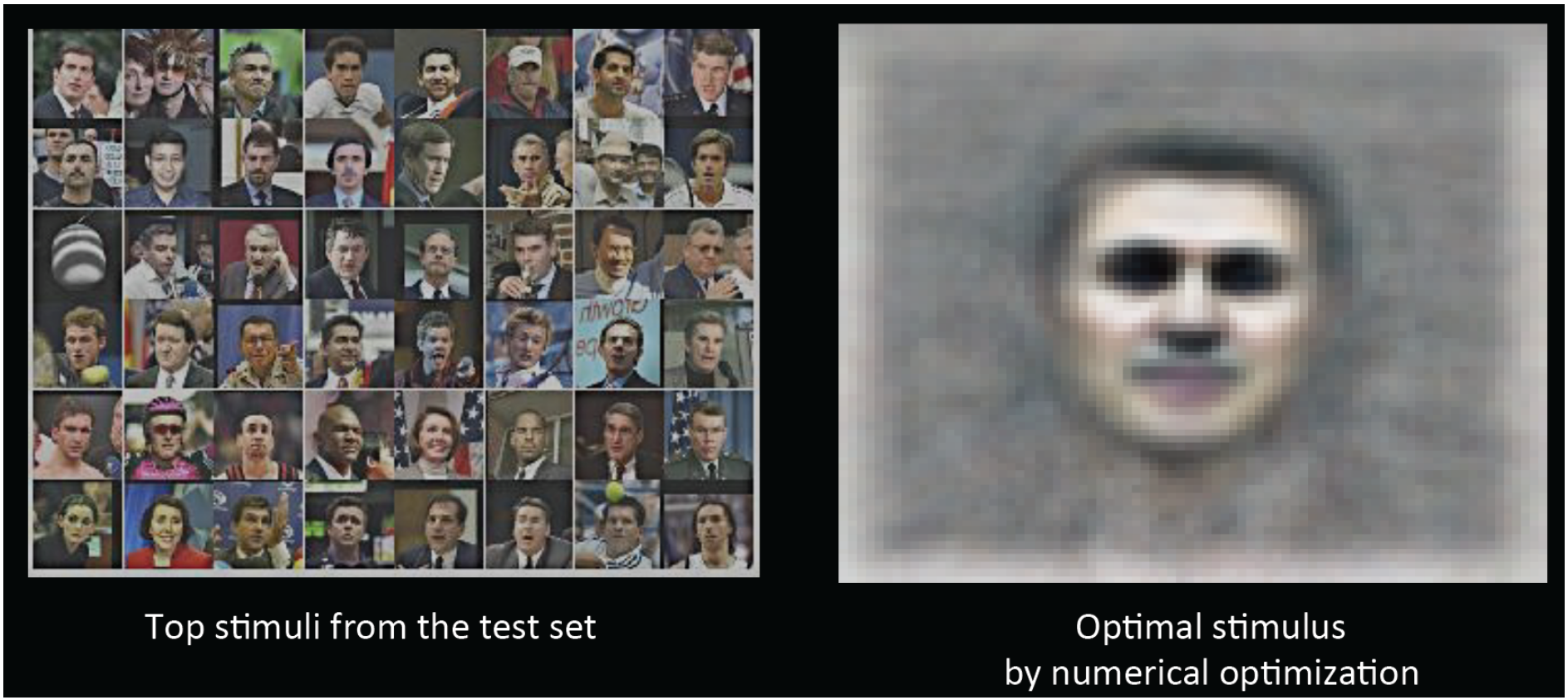
Deep network has 1 billion parameters, trained on 16k cores for a week

Cat neuron



*Graphics from [Le et al., 2012]

Face neuron



*Graphics from [Le et al., 2012]

Further enhancements worth knowing about

- SGD alternative, e.g. 2nd order methods [Martens, 2010], accelerated gradient [Sutskever et al., 2013]
- Better regularization, e.g. Dropout [Hinton et al., 2012b]
- Scaling to large data, e.g. [Dean et al., 2012, Coates et al., 2013]
- Hyper-parameter search, e.g. [Bergstra et al., 2011]
- Recent analyses on why things work or fail, e.g. [Szeged et al., 2014]

Outline

1. Deep Learning Background
   - Neural Networks (1-layer, 2-layer)
   - Potentials and Difficulties of Deep Architecture
   - The Breakthrough in 2006
2. Two Main Types of Deep Architectures
   - Deep Belief Nets (DBN) [Hinton et al., 2006]
   - Stacked Auto-Encoders (SAE) [Bengio et al., 2006]
   - Current Status of Deep Learning
3. Applications in Natural Language Processing and Machine Translation
   - Use as Non-linear classifier
   - Use as Distributed representation
   - Survey of Machine Translation Research

Recent Papers with keywords: Deep or Neural

Sequence labeling

- POS tagging & Name Entity Recognition [Turian et al., 2010, Collobert et al., 2011, Wang and Manning, 2013, Ma et al., 2014, Tsuboi, 2014, Guo et al., 2014, Qi et al., 2014]
- Word Segmentation [Zheng et al., 2013, Pei et al., 2014]

Syntax & Morphology

- Dependency Parsing [Stenetorp, 2013, Chen et al., 2014a, Levy and Goldberg, 2014, Bansal et al., 2014, Chen and Manning, 2014, Le and Zuidema, 2014]
- Constituency Parsing [Billingsley and Curran, 2012, Socher et al., 2013a, Andreas and Klein, 2014]
- CCG [Hermann and Blunsom, 2013], Selectional Preference [Van de Cruys, 2014], Morphology [Luong et al., 2013]

Semantics

- Word Representations [Tsubaki et al., 2013, Srivastava et al., 2013, Rocktäschel et al., 2014, Baroni et al., 2014, Hashimoto et al., 2014, Pennington et al., 2014, Neelakantan et al., 2014, Chen et al., 2014b, Milajevs et al., 2014]
- Semantic Role Labeling: [Hermann et al., 2014, Roth and Woodsend, 2014]
- Paraphrase [Socher et al., 2011]
- Grounding/Multi-modal [Fyshe et al., 2014, Kiela and Bottou, 2014]

Discourse

- [Ji and Eisenstein, 2014, Li et al., 2014a]

Question Answering, Knowledge Bases, & Relation Extraction

- [Hashimoto et al., 2013, Fu et al., 2014, Chang et al., 2014, Yih et al., 2014, Bordes et al., 2014, Iyyer et al., 2014, Yang et al., 2014, Gardner et al., 2014]

Sentiment Analysis

- [Glorot et al., 2011, Socher et al., 2013b, Irsoy and Cardie, 2014]

Summarization

- [Liu et al., 2012]

Novel Applications

- Poetry [Zhang and Lapata, 2014], Interestingness [Gao et al., 2014b], Hashtags [Weston et al., 2014]

Disclaimer!

- There's no consensus yet on how best to apply Deep Learning in NLP
  - Good results have been reported, but not yet revolutionary
  - Compared to Vision/Speech, methods for NLP seem to have less emphasis on "deep." Perhaps a single word is already an extremely informative feature, worth a thousand pixels?
- What we'll do: Summarize 2 ways Deep/Neural ideas can be used
  1. As non-linear classifier
  2. As distributed representation
- Show one successful and one unsuccessful case study each, to emphasise that everything is work-in-progress!
  $\rightarrow$ What you believe is good today may be bad tomorrow.

Use as Non-linear Classifier

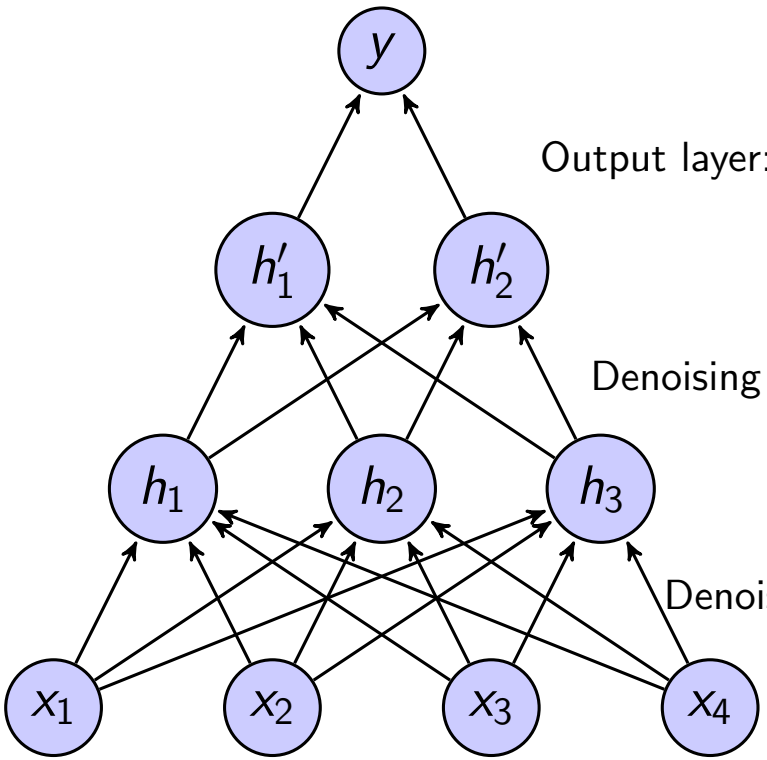
Idea: Directly replace a linear classifer used in NLP with a deep network.

Expected to work if:

1. Difficult to engineer effective features
2. Linear classifier underfits (e.g. high training error)

Case Study 1: Domain Adaptation for Large-Scale Sentiment Classification [Glorot et al., 2011]

- Amazon Review dataset [Blitzer et al., 2007]
  - 4 domains: electronics, books, DVDs, kitchen.
  - Pre-train on unlabeled data to get "good features" across domains
  - Then, train SVM on target labeled data per domain



Output layer: SVM

Denoising Auto-Encoder: noise=gaussian; rectifier non-linearity

Denoising Auto-Encoder: noise=randomly set x_i=0; σ non-linearity

Input: 5000 most frequent 1gram/2gram words

Hyperparameters: Masking noise (0.8), Gaussian noise var, number of hidden layers (1-3), no. hidden layers (1000, 2500, 5000), regularization, learning rate γ

Experiment Results [Glorot et al., 2011]

Evaluation metric:

- $error(A, B)$ = error of classifier trained on DomainA, test on DomainB
- transfer loss = $error(Source, Target) - error(Target, Target)$

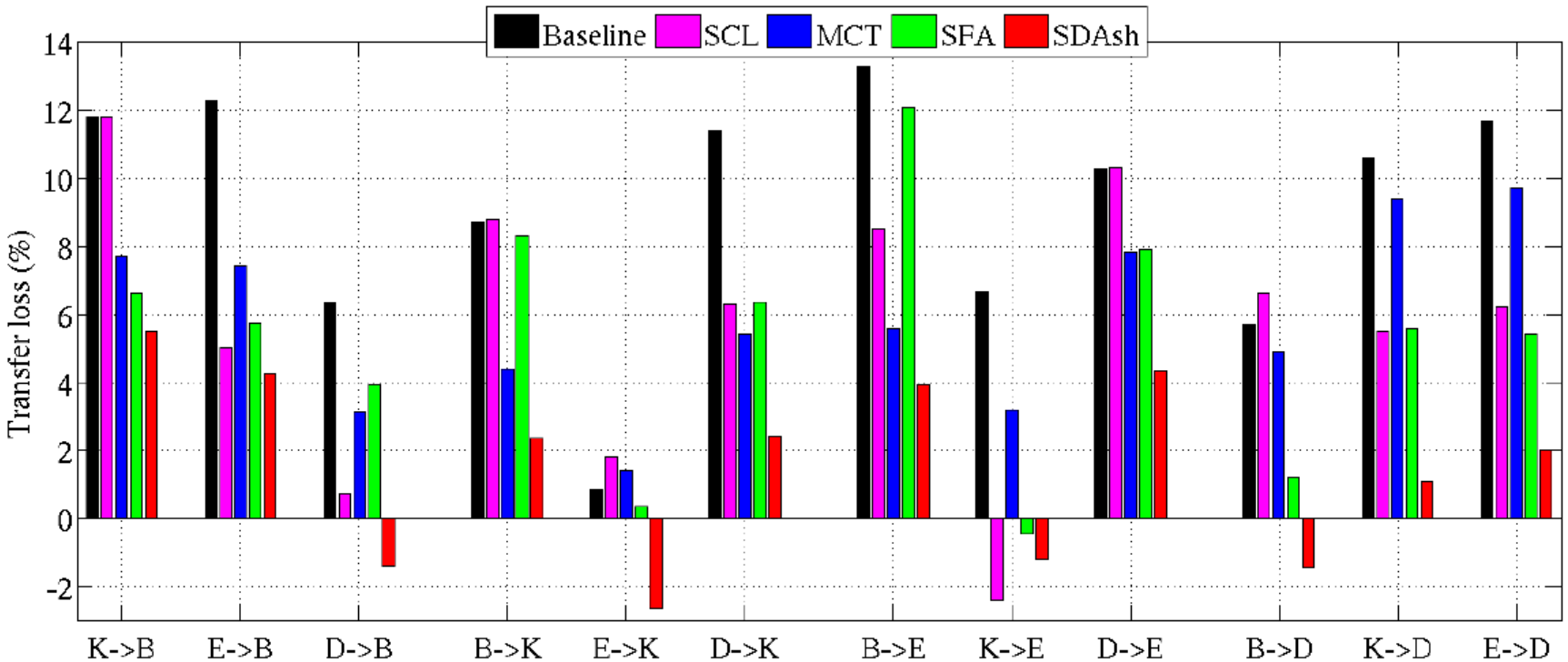
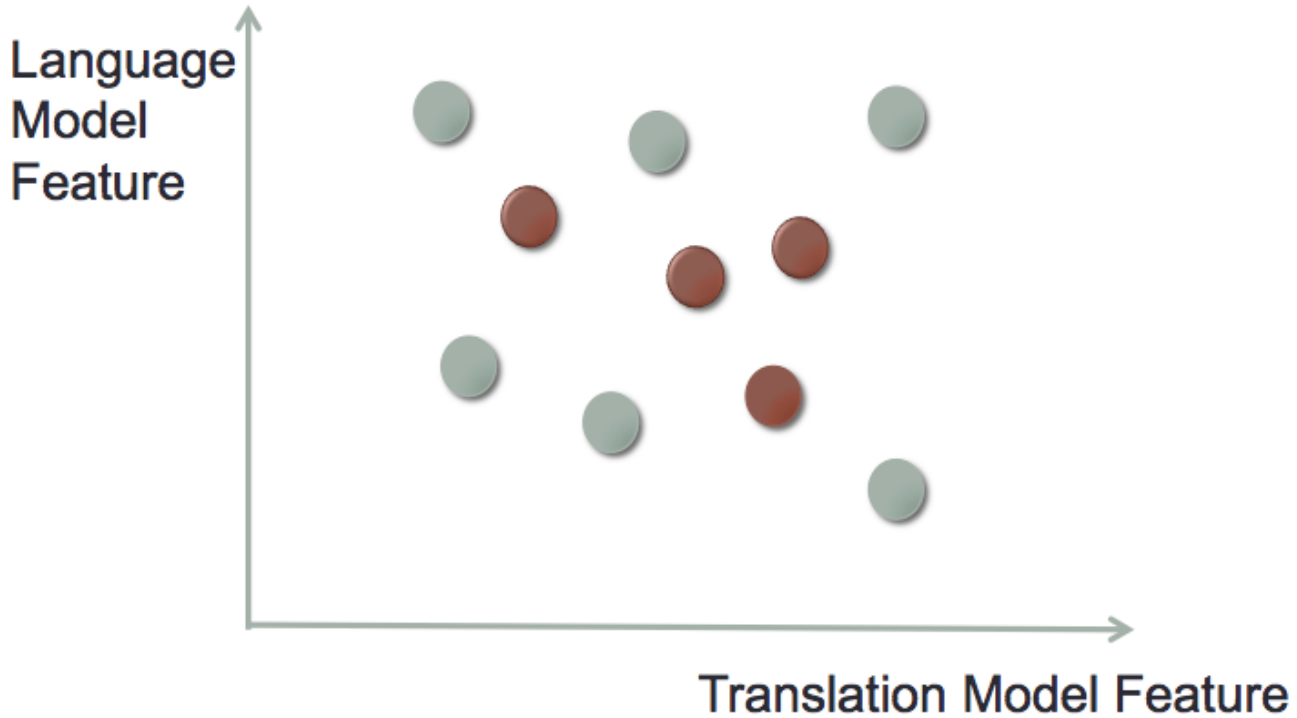


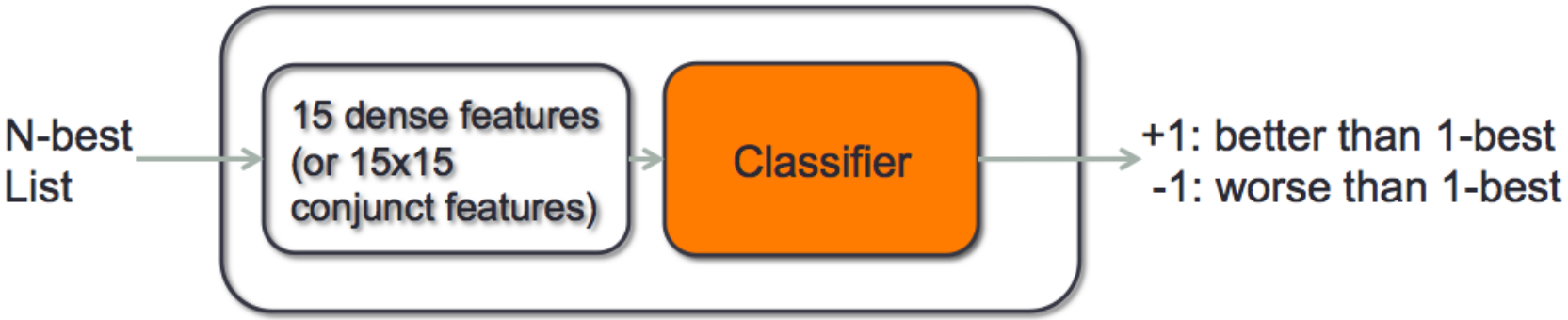
Figure 1. **Transfer losses on the Amazon benchmark** of 4 domains: *Kitchen*(K), *Electronics*(E), *DVDs*(D) and *Books*(B). All methods are trained on the labeled set of one domain and evaluated on the test sets of the others. SDA_{sh} outperforms all others on 11 out of 12 cases.

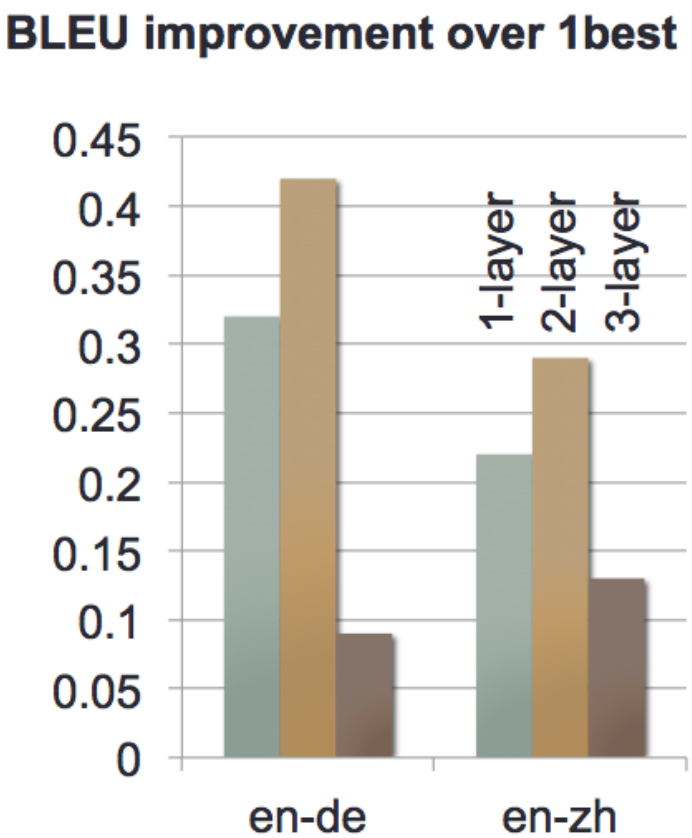
Case Study 2: Machine Translation N-best List Re-ranking

- Hypothesis: Dense features in current systems are not sufficiently expressive [Duh and Kirchhoff, 2008]
  - Translation model, language model scores are too coarse-grained
  - Linear re-ranker attains only convex hull of N-best candidates



Experiment Results





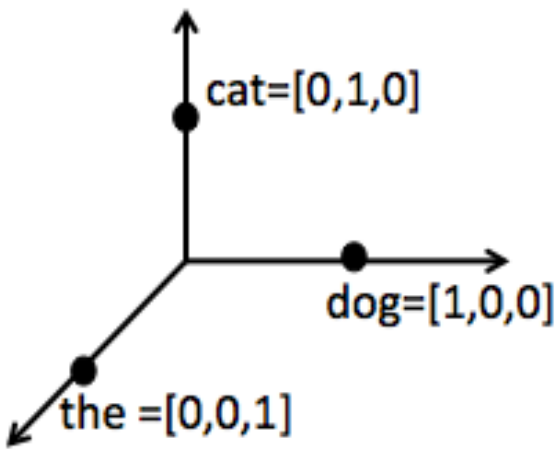
Issue:
3-layer net overfits on wrong objective

Outline

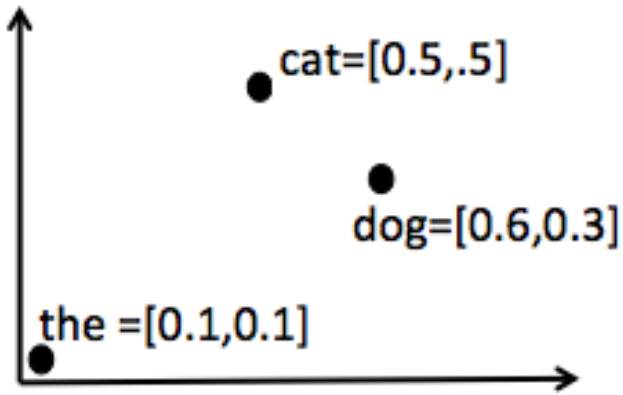
1. Deep Learning Background
   - Neural Networks (1-layer, 2-layer)
   - Potentials and Difficulties of Deep Architecture
   - The Breakthrough in 2006
2. Two Main Types of Deep Architectures
   - Deep Belief Nets (DBN) [Hinton et al., 2006]
   - Stacked Auto-Encoders (SAE) [Bengio et al., 2006]
   - Current Status of Deep Learning
3. Applications in Natural Language Processing and Machine Translation
   - Use as Non-linear classifier
   - Use as Distributed representation
   - Survey of Machine Translation Research

Distributed (Vector) Representation of Words

- Embed word in vector space, such that nearby words are syntactically or semantically similar
- Neural nets can be used to learn these vectors from raw text [Collobert et al., 2011, Chen et al., 2013]



One-Hot Representation:
Word as discrete atomic feature



Distributed Representation:
Word as vectors

Use as Distributed Representation

Idea: Replace/Append original features with distributed representation.

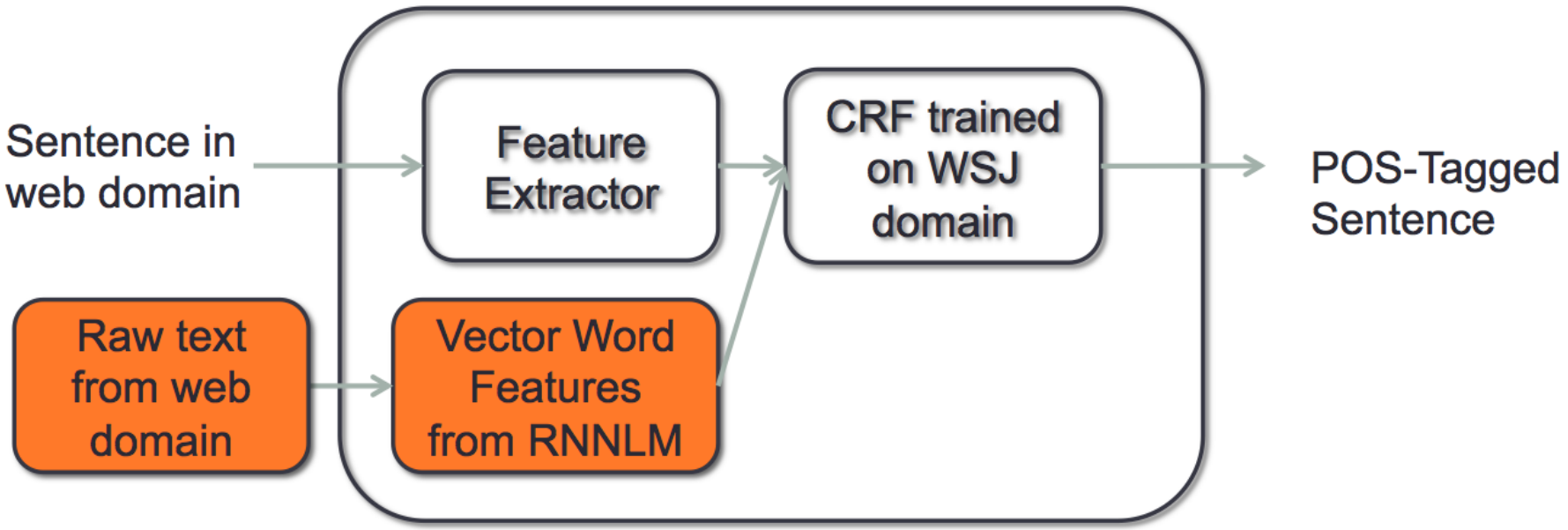
Expected to work if:

1. Original features are too sparse (e.g. small training data)
2. Distributed representation enables more flexible model of language

Case Study 1: POS tagging of Web Text

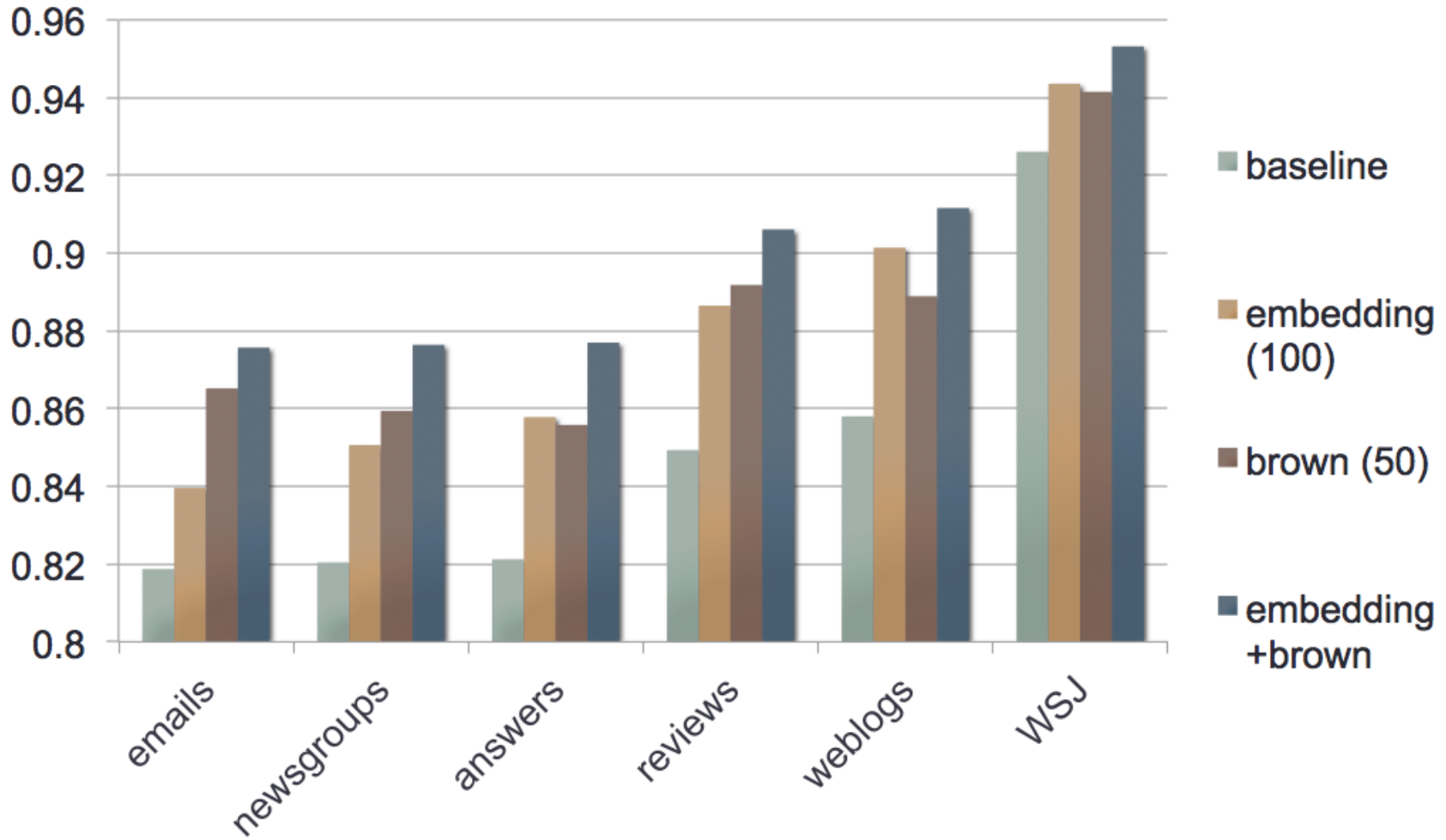
Motivation: POS tagging performs well on WSJ (news) but poorly on web text. One main reason is unknown/rare words.

- Distributed representation ensures unknown words can be tagged because similar words occur in WSJ Treebank



*RNNLM = Recurrent Neural Net Language Model [Mikolov et al., 2011]

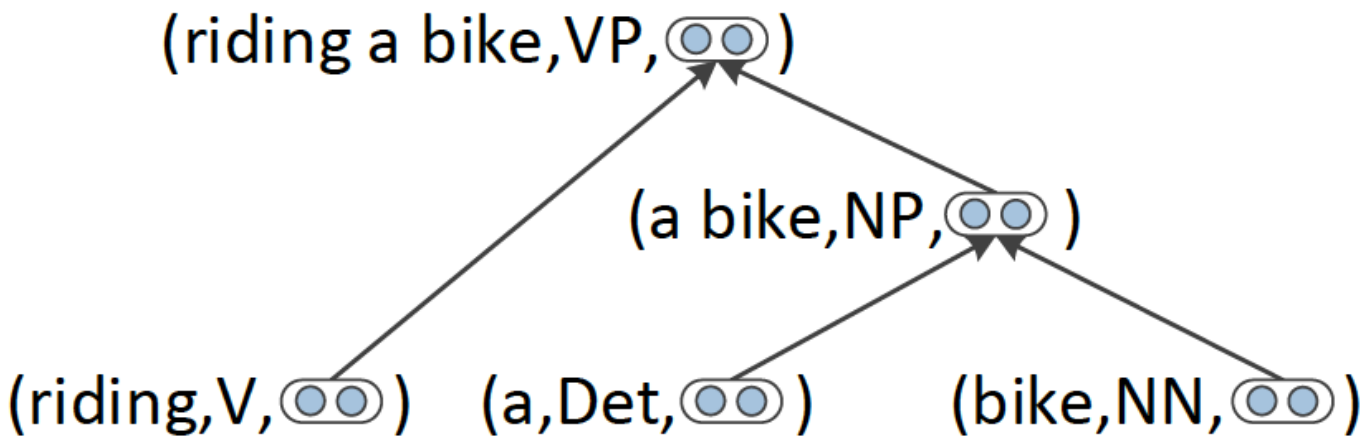
POS Tagging Accuracy (SANCL2012 Shared Task data)



Conclusion:

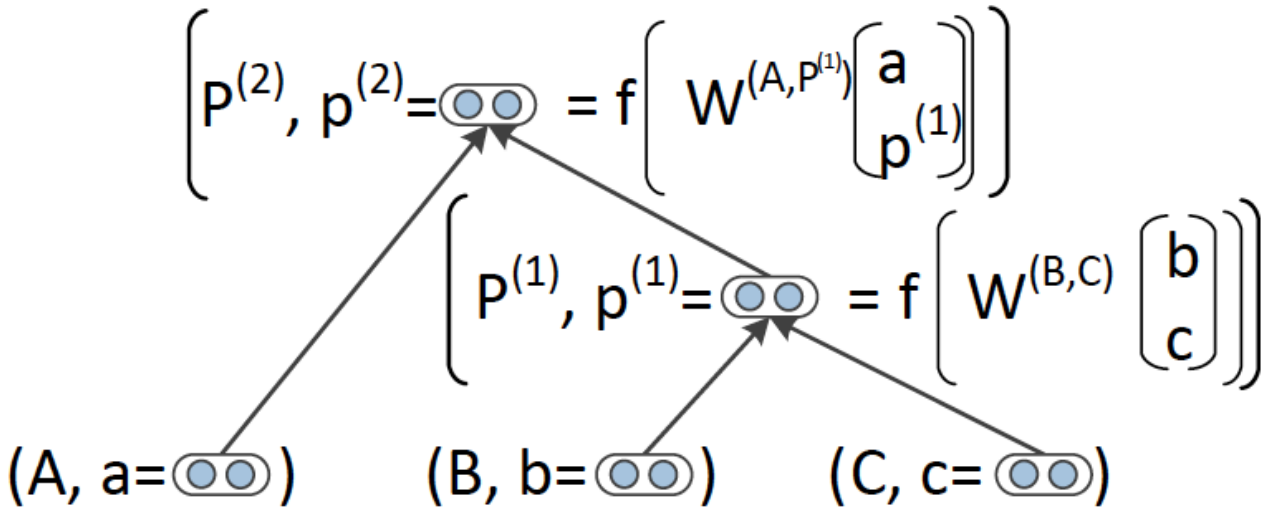
- Distributed representation helps. But Brown clustering is just as good.
- What to improve: Representation? CRF integration? Joint learning?

Case Study 2: Parsing with Compositional Vector Grammar [Socher et al., 2013a]



- Background: Parsing results can be improved by splitting coarse categories (e.g. NP, VP).
- Here, rather than splitting, directly learn distributed representation of phrases

Compositional Vector Grammar [Socher et al., 2013a]



- Begin with word representation. Phrase vector is output of 1 layer net, compute recursively bottom-up.
- The score of e.g. node $p^{(1)}$ is $v^{(B,C)}p^{(1)} + \log P(P_1 \rightarrow BC)$
- Predicted parse tree is the one that achieves max sum of node scores.
- Weight matrix (for each node type) is trained by structured max-margin objective

WSJ Section 23 Experiment Results

System	F1
Stanford Parser (PCFG)	85.5
Stanford Parser (Factored)	86.6
Berkeley Parser	90.1
Compositional Vector Grammar	90.4

End-to-end distributed representation outperforms both manually factored and automatically split state systems.

Case Study Summary

Exploiting Non-linear Classifiers

- It's possible to directly apply Deep Learning to text problems with little modification, as evidenced by [Glorot et al., 2011]
- But sometimes NLP-specific modifications are needed, e.g. training objective mismatch in Machine Translation N-best experiment

Exploiting Distributed Representation

- Distributed Representation is a simple way to improve robustness of NLP, but it's not the only way (POS tagging experiment)
- Promising direction: distributed representations beyond words, considering e.g. compositionality [Socher et al., 2013a]

Note: The above uses are really two sides of the same coin, not mutually-exclusive.

Outline

1. Deep Learning Background
   - Neural Networks (1-layer, 2-layer)
   - Potentials and Difficulties of Deep Architecture
   - The Breakthrough in 2006
2. Two Main Types of Deep Architectures
   - Deep Belief Nets (DBN) [Hinton et al., 2006]
   - Stacked Auto-Encoders (SAE) [Bengio et al., 2006]
   - Current Status of Deep Learning
3. Applications in Natural Language Processing and Machine Translation
   - Use as Non-linear classifier
   - Use as Distributed representation
   - Survey of Machine Translation Research

A Taxonomy of Neural Nets in Machine Translation

Core Engine: What is being modeled?

- Target word probability:
  - Language Model: $p(target\_word_t \mid target\_word_{t-1})$
  - Language Model with Source Information: $p(target\_word_t \mid target\_word_{t-1}, source)$
- Translation/Reordering probabilities under Phrase-based MT:
  - Translation Model: $p(target\_phrase \mid source\_phrase)$
  - Reordering Model: $p(orientation \mid target\_phrase, source\_phrase)$
- Probabilities under Tuple-based MT: $p([target\_phrase, source\_phrase]_t \mid [target\_phrase, source\_phrase]_{t-1})$
- Inversion Transduction Grammar (ITG) Model

Related Components:

- Word Alignment
- Adaptation / Topic Context
- Multilingual Embeddings

A Taxonomy of Neural Nets in Machine Translation

Core Engine: What is being modeled?

- Target word probability:
  - Language Model: [Schwenk et al., 2012, Vaswani et al., 2013, Niehues and Waibel, 2013, Auli and Gao, 2014]
  - LM w/ Source: [Kalchbrenner and Blunsom, 2013, Auli et al., 2013, Devlin et al., 2014, Cho et al., 2014, Bahdanau et al., 2014, Sundermeyer et al., 2014, Sutskever et al., 2014]
- Translation/Reordering probabilities under Phrase-based MT:
  - Translation: [Maskey and Zhou, 2012, Schwenk, 2012, Liu et al., 2013, Gao et al., 2014a, Lu et al., 2014, Tran et al., 2014, Wu et al., 2014a]
  - Reordering: [Li et al., 2014b]
- Tuple-based MT: [Son et al., 2012, Wu et al., 2014b, Hu et al., 2014]
- ITG Model: [Li et al., 2013, Zhang et al., 2014, Liu et al., 2014]

Related Components:

- Word Align: [Yang et al., 2013, Tamura et al., 2014, Songyot and Chiang, 2014]
- Adaptation / Topic Context: [Duh et al., 2013, Cui et al., 2014]
- Multilingual Embeddings: [Klementiev et al., 2012, Lauly et al., 2013, Zou et al., 2013, Kočiský et al., 2014, Faruqui and Dyer, 2014, Hermann and Blunsom, 2014, Chandar et al., 2014]

Next, we'll discuss...

Core Engine: What is being modeled?

- Target word probability:
  - Language Model: [Vaswani et al., 2013, Auli and Gao, 2014]
  - LM w/ Source: [Kalchbrenner and Blunsom, 2013, Devlin et al., 2014, Sutskever et al., 2014]
- Translation/Reordering probabilities under Phrase-based MT:
  - Translation: [Gao et al., 2014a]
  - Reordering
- Tuple-based MT: [Son et al., 2012]
- ITG Model: [Zhang et al., 2014]

Related Components:

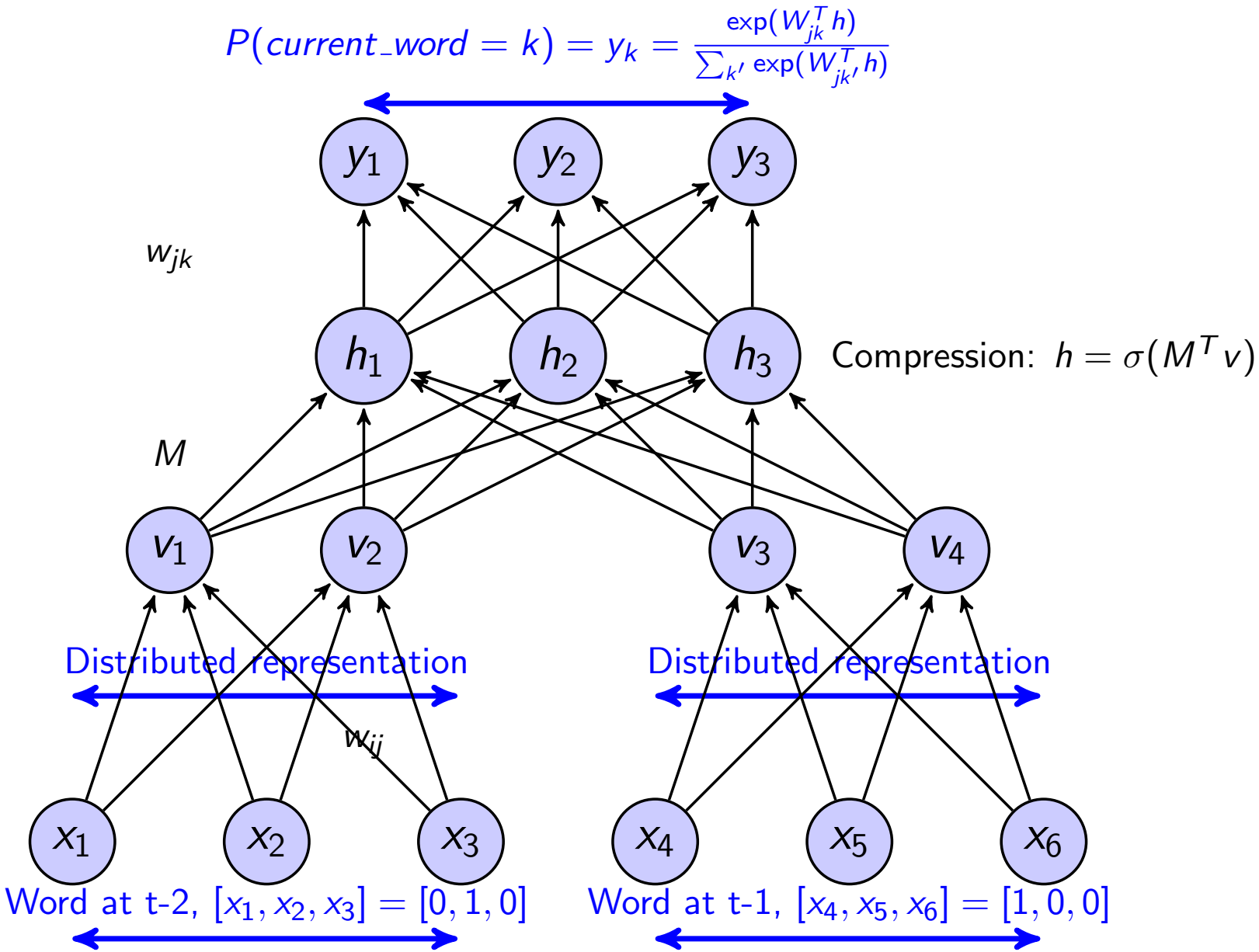
- Word Align
- Adaptation / Topic Context
- Multilingual Embeddings: [Klementiev et al., 2012]

(Obviously there's no time to discuss everything. These papers are chosen to pedagogically demonstrate the diverse ways in which neural nets are used in MT.)

Language Models (LM) using Neural Nets

- Model $P(\textit{current\_word} \mid \textit{previous\_words})$ using neural nets.
  - Motivation: Continuous distributed representations of words learned by neural nets reduce sparsity problems
- Example rare word: "Bar-ba-loots"
  - $P(\mathbf{w_t} = \textit{fruits} \mid \mathbf{w_{t-2}} = \textit{like}, \mathbf{w_{t-2}} = \textit{Bar-ba-loots}) = ?$
  - $P(\mathbf{w_t} = \textit{bars} \mid \mathbf{w_{t-2}} = \textit{like}, \mathbf{w_{t-2}} = \textit{Bar-ba-loots}) = ?$
  - Which has higher probability?
  - What if I tell you **vector**(Bar-ba-loots) is similar to **vector**(bears)?
- Feed-forward Neural Net Language Model:
  - (1) Map $\mathbf{w_{t-1}}, \mathbf{w_{t-2}}, \ldots$ to vectors. (2) Compress. (3) Predict $\mathbf{w_t}$
- Recurrent Neural Net Language Model:
  - (1) Map $\mathbf{w_{t-1}}$ to vector.
    (2) Combine with previous state & compress.
    (3) Predict $\mathbf{w_t}$

(Feed-forward) Neural Language Models [Bengio et al., 2003]

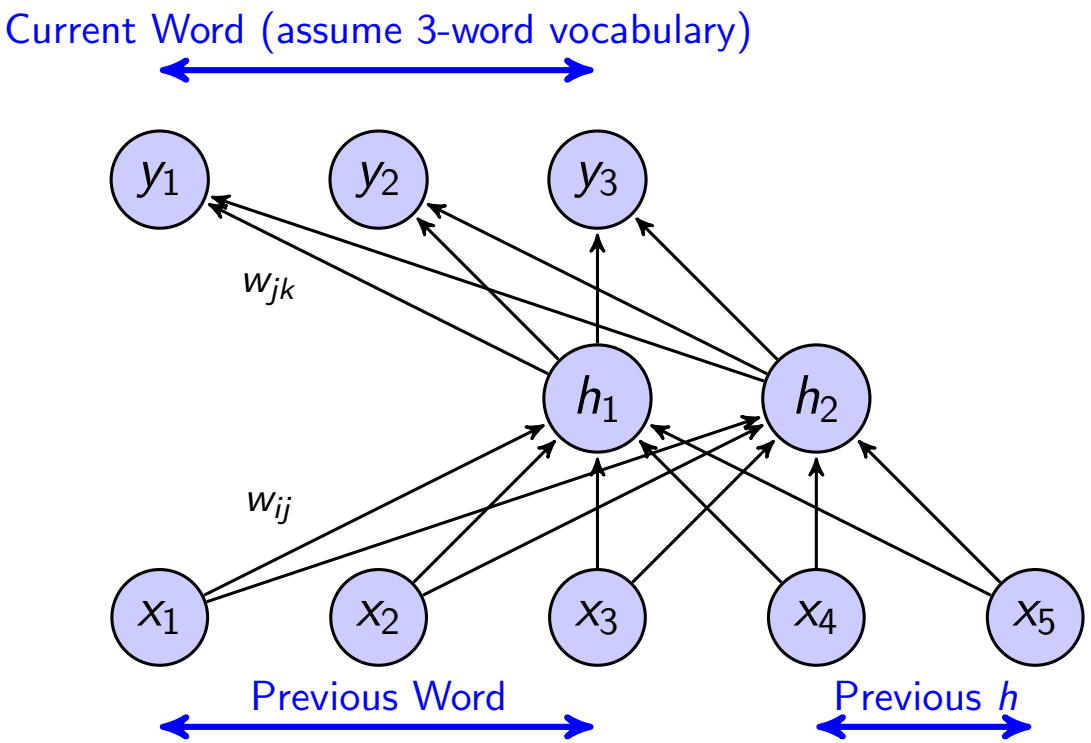


Training Feed-Forward Neural LMs

- Training data = sets of n-gram
  - Supervised task: Given previous n-1 words, predict current word
  - Standard Backpropagation works
  - Deeper nets are possible [Arisoy et al., 2012] (minor gains?)
- By-product: $[w_{ij}]_i$ can be used as "word embeddings". Useful for many applications [Zhila et al., 2013, Turian et al., 2010]
- In practice:
  - $y_k = \frac{\exp(W_{jk}^T h)}{\sum_{k'} \exp(W_{jk'}^T h)}$ requires expensive summation k over vocabulary size
  - Many speed-up techniques proposed, e.g. class-based vocabulary, noise-contrastive estimation, approximate normalization
  - If we only need embeddings, alternative models are recommended, esp. [Collobert et al., 2011], word2vec [Mikolov et al., 2013]

Recurrent Neural Net Language Models [Mikolov et al., 2010]

Model $p(current\_word|previous\_words)$ with a recurrent hidden layer



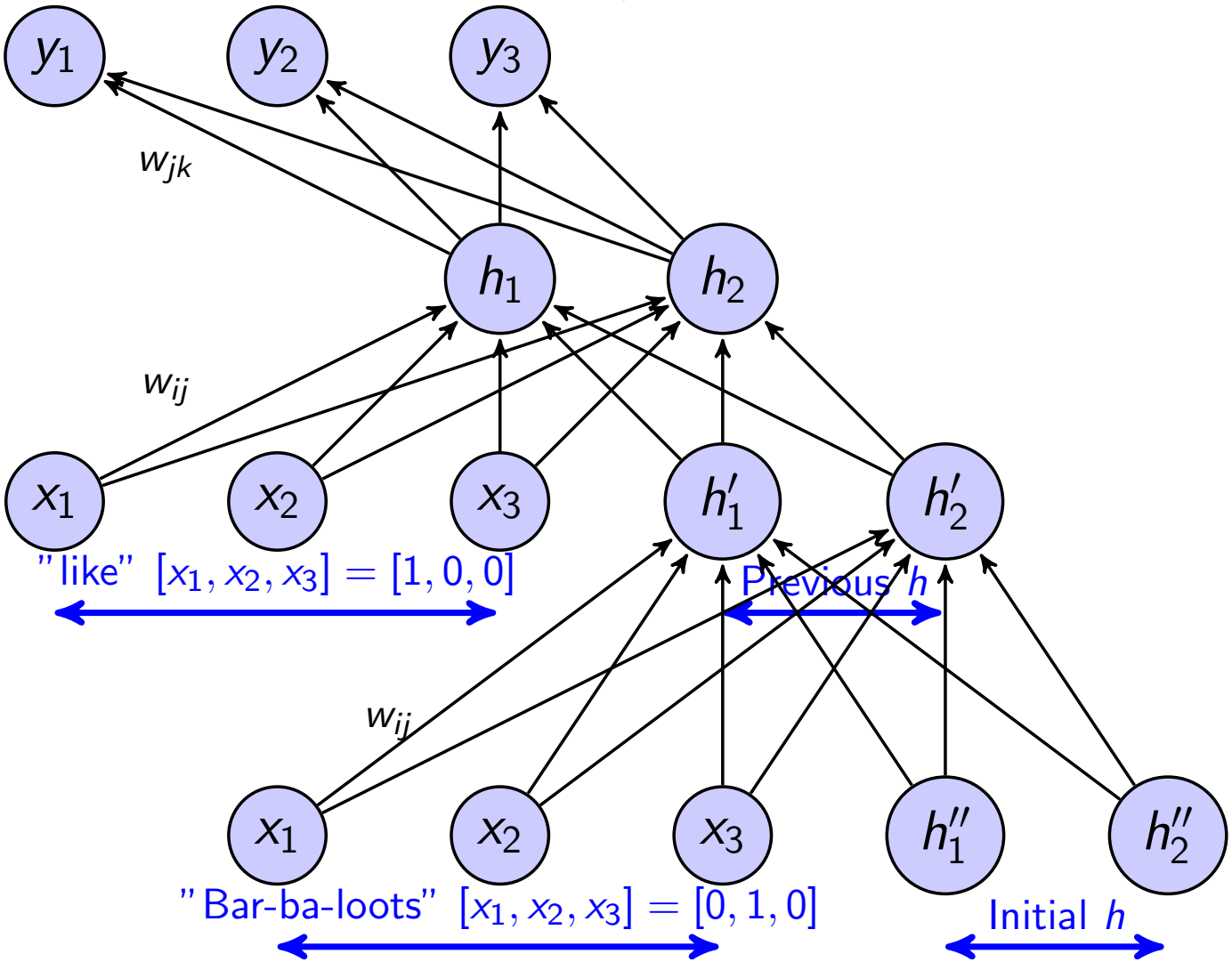
- Probability of word k: $y_k = \frac{\exp(W_{jk}^T h)}{\sum_{k'} \exp(W_{jk'}^T h)}$
- $[x_4, x_5]$ is a copy of $[h_1, h_2]$ from the previous time-step
- $h_j = \sigma(W_{ij}^T x_i)$ is hidden state of partial sentence
- Arbitrarily-long history is (theoretically) kept through recurrence

Training Recurrent Nets: Backpropagation through Time

Unroll the hidden states for certain time-steps.

Given error at y, update weights by backpropagation

Example: Bar-ba-loots like | fruits



Neural Language Model Decoder Integration

- Feed-forward Neural LMs have same form as n-grams, so straightforward to integrate into MT decoder
  - Caveat: For calculation speed-up (esp. normalization constant), resort to approximations and caching.
- Recurrent Neural LMs require history going back to start-of-sentence. Harder to do dynamic programming.
  - [Auli and Gao, 2014]: To score new words, each decoder state needs to maintain h. For recombination, merge hypotheses by traditional n-gram context but keep a beam of h's.

Neural Language Models generally improve MT

Feed-forward Neural LM [Vaswani et al., 2013]

	NIST06	WMT06		
	Zh-En	Fr-En	De-En	Es-En
baseline (n-gram)	34.3	25.5	21.5	32.0
1000-best rescoring	34.7	26.0	21.5	32.2
decoding	34.9	26.1	21.9	32.1

Recurrent Neural LM [Auli and Gao, 2014]

	WMT12 Fr-En	WMT12 De-En
baseline (n-gram)	24.85	19.80
100-best rescoring	25.74	20.54
lattice rescoring	26.43	20.63
decoding	26.86	20.93

Next, we'll discuss...

Core Engine: What is being modeled?

- Target word probability:
  - Language Model: [Vaswani et al., 2013, Auli and Gao, 2014]
  - LM w/ Source: [Kalchbrenner and Blunsom, 2013, Devlin et al., 2014, Sutskever et al., 2014]
- Translation/Reordering probabilities under Phrase-based MT:
  - Translation: [Gao et al., 2014a]
  - Reordering
- Tuple-based MT: [Son et al., 2012]
- ITG Model: [Zhang et al., 2014]

Related Components:

- Word Align
- Adaptation / Topic Context
- Multilingual Embeddings: [Klementiev et al., 2012]

Language Model with Source

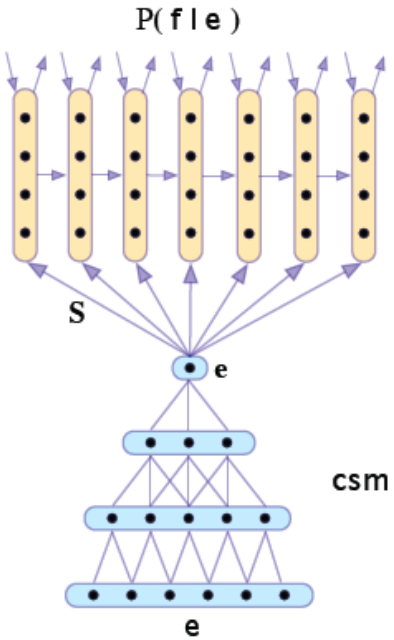
- Model $p(target\_word_t \mid target\_word_{t-1}, source)$
- Main question is how to model source:
  - Entire source sentence or aligned source words only?
  - Vector representation or traditional words?
  - If vector representation, how to compute it?

Model of [Devlin et al., 2014]

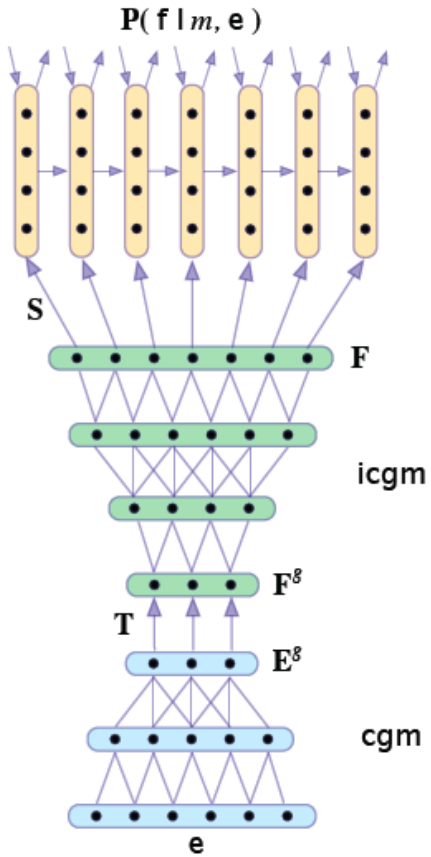
- Extend feed-forward neural LM to include window around aligned source word.
  - Heuristic: If align to multiple source words, choose middle. If unaligned, inherit alignment from closest target word
- Train on bitext with alignment; optimize target likelihood.

Model of [Kalchbrenner and Blunsom, 2013]

(f=target, e=source for our purposes here)



RCTM I

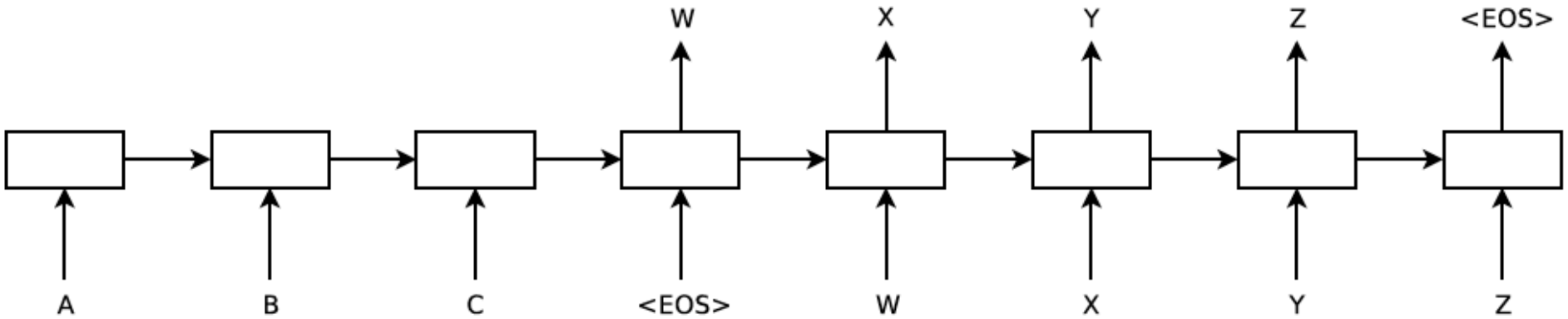


RCTM II

- Extend recurrent LM with vector representation of source sentence
  - A matrix K convolves arbitrary-length source sentence into vector(s) of predetermined dimension
- Train on bitext *without* alignment; optimize target likelihood.

Model of [Sutskever et al., 2014]

("A B C" is source sentence; "W X Y Z" is target sentence)



- Treats MT as general sequence-to-sequence transduction
  - (1) Read source (2) Accumulate hidden state, (3) Generate target.
  - End-of-sentence "⟨EOS⟩" token stops the recurrent process.
  - In practice, read input sentence in reverse gave better MT results.
- Used Long Short-Term Memory (LSTM); better modeling of long-range dependencies than basic recurrent nets.
- Train on bitext; optimize target likelihood. (Common in all LM w/ Source models)

Next, we'll discuss...

Core Engine: What is being modeled?

- Target word probability:
  - Language Model: [Vaswani et al., 2013, Auli and Gao, 2014]
  - LM w/ Source: [Kalchbrenner and Blunsom, 2013, Devlin et al., 2014, Sutskever et al., 2014]
- Translation/Reordering probabilities under Phrase-based MT:
  - Translation: [Gao et al., 2014a]
  - Reordering
- Tuple-based MT: [Son et al., 2012]
- ITG Model: [Zhang et al., 2014]

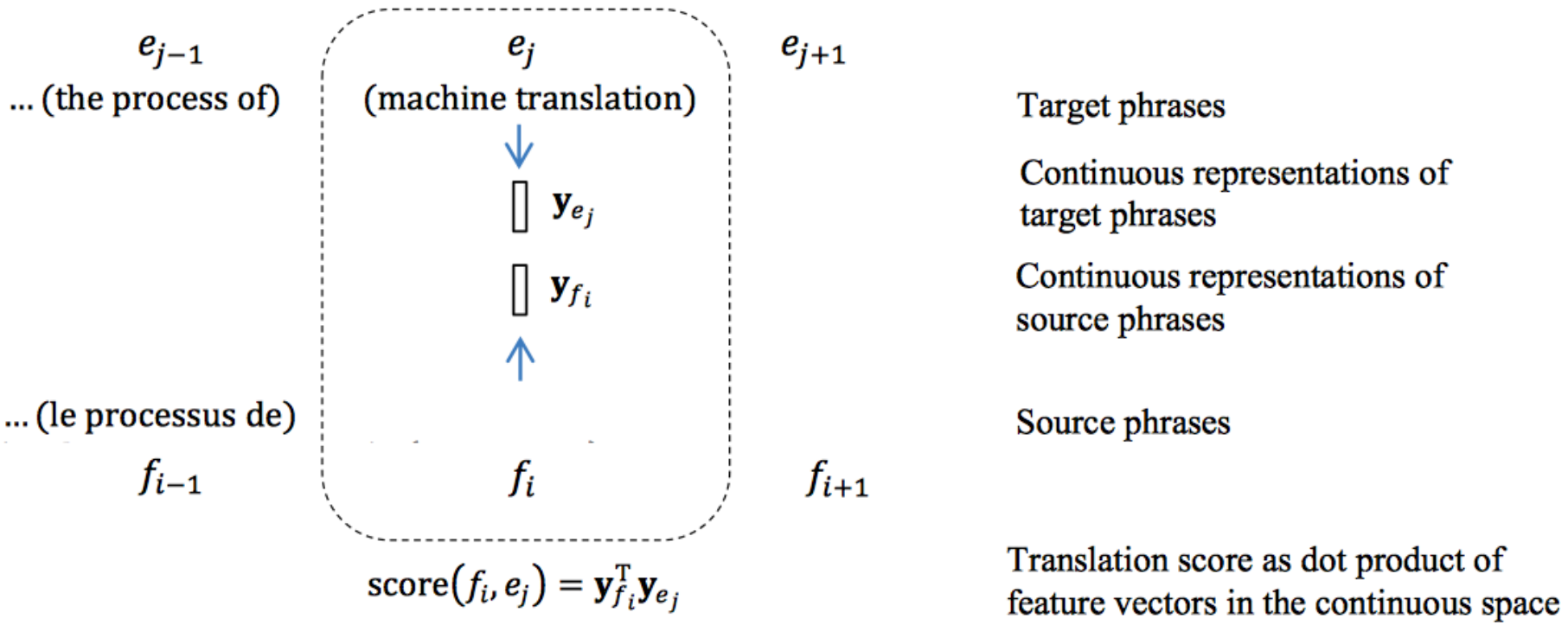
Related Components:

- Word Align
- Adaptation / Topic Context
- Multilingual Embeddings: [Klementiev et al., 2012]

Neural Net Translation Model under Phrase-based MT

- Recall log-linear MT formulation:
  $\arg\max_{target} \sum_k \lambda_k \Phi_k(target, source, align)$ where Φ_k are language model, translation model scores, etc.
- Translation model score $p(target\_phrase \mid source\_phrase)$ is conventionally based on counts (max likelihood estimate)
- Potential advantages of replacing this score with neural net score:
  - Alleviate data sparsity
  - Enable complex scoring functions
  - Incorporate more source side context e.g. [Tran et al., 2014]
- Easy to add as feature, with no decoder modification.

Model of [Gao et al., 2014a]



- Two neural nets (one for source side, one for target side)
  - Input: Bag-of-words representation of source/target phrase
  - Output: Vectors $\mathbf{y}_{\mathbf{f_i}}$ for source phrase, $\mathbf{y}_{\mathbf{e_j}}$ for target phrase
- Score of phrase pair = dot product of these vectors $\mathbf{y}_{\mathbf{f_i}}{}^T \mathbf{y}_{\mathbf{e_j}}$

Training procedure of [Gao et al., 2014a]

1. Baseline MT generates N-best list for training data.
   - Key assumption: oracle in N-best is much better than 1-best, so it's worthwhile to train the neural nets.
2. Optimize neural net parameters:
   - Use "Expected BLEU"[1] objective to enable smooth gradients: $\sum_{e_i,f_j} \frac{\delta ExpBLEU(W)}{\delta score_W(\mathbf{y}_{\mathbf{f_i}},\mathbf{y}_{\mathbf{e_j}})} \frac{\delta score_W(\mathbf{y}_{\mathbf{f_i}},\mathbf{y}_{\mathbf{e_j}})}{\delta W}$
3. Optimize (MERT) feature weights λ in log-linear model $\sum_k \lambda_k \Phi_k(target, source, align)$. [Loop if desired]

Note: Alternative models / training procedures are possible, e.g.

- Pairwise ranking (PRO) objective on dev set [Liu et al., 2013]
- Direct training on extracted phrase table [Schwenk, 2012]
- RBMs/Autoencoders on top of conventional phrase features [Maskey and Zhou, 2012, Lu et al., 2014]

[1] Expected Bleu: $\sum_{E \in nbest} P(E|F) sentBLEU(E, E_{ref})$

Next, we'll discuss...

Core Engine: What is being modeled?

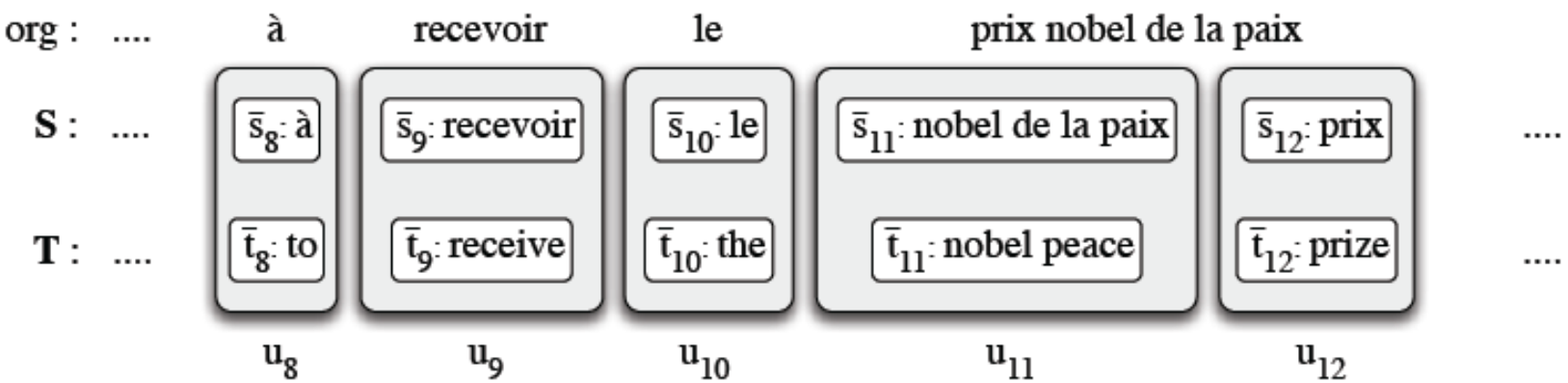
- Target word probability:
  - Language Model: [Vaswani et al., 2013, Auli and Gao, 2014]
  - LM w/ Source: [Kalchbrenner and Blunsom, 2013, Devlin et al., 2014, Sutskever et al., 2014]
- Translation/Reordering probabilities under Phrase-based MT:
  - Translation: [Gao et al., 2014a]
  - Reordering
- Tuple-based MT: [Son et al., 2012]
- ITG Model: [Zhang et al., 2014]

Related Components:

- Word Align
- Adaptation / Topic Context
- Multilingual Embeddings: [Klementiev et al., 2012]

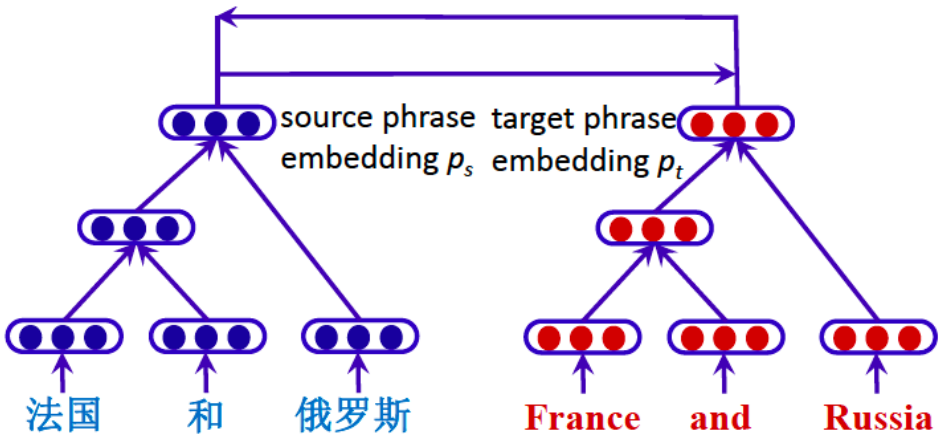
Tuple-based MT [Son et al., 2012]

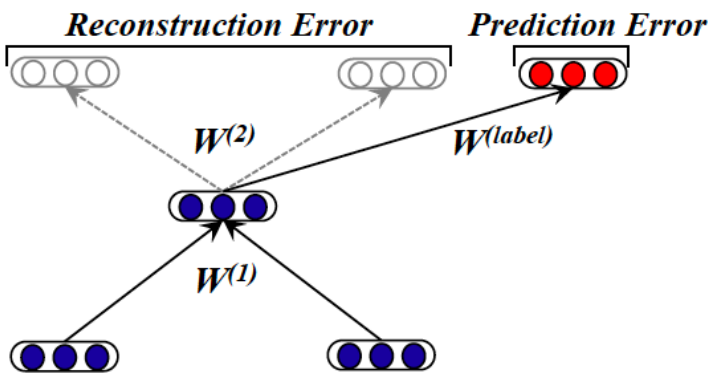
$$p([target\_phrase, source\_phrase]_t \mid [target\_phrase, source\_phrase]_{t-1})$$



- A target and source phrase tuple forms a single unit (**u**)
- Apply standard neural language model to score sequences of **u**
  - Challenge: Large output space using tuple.
  - One Solution: Factorize!

Inversion Transduction Grammar (ITG) Model





ITG views translation as bilingual parsing, allows phrase blocks to combine hierarchically.

Basic component:
$p(straight/inverted \mid block_1, block_2)$
can use neural net [Li et al., 2013]

[Zhang et al., 2014] extension: constrain phrase embeddings of translations to be similar.

Advantage: Elegantly models monolingual composition and bilingual equivalence in unified framework.

Next, we'll discuss...

Core Engine: What is being modeled?

- Target word probability:
  - Language Model: [Vaswani et al., 2013, Auli and Gao, 2014]
  - LM w/ Source: [Kalchbrenner and Blunsom, 2013, Devlin et al., 2014, Sutskever et al., 2014]
- Translation/Reordering probabilities under Phrase-based MT:
  - Translation: [Gao et al., 2014a]
  - Reordering
- Tuple-based MT: [Son et al., 2012]
- ITG Model: [Zhang et al., 2014]

Related Components:

- Word Align
- Adaptation / Topic Context
- Multilingual Embeddings: [Klementiev et al., 2012]

Multilingual Embeddings

- What: Train word representations such that words in different languages map to same space
- Why: Useful for many cross-lingual tasks as well as MT
  - Train classifier on large labeled English data; Test on Xhosa
- Main questions:
  - Amount of multilingual info: parallel bitext? comparable corpora? word alignment table?
  - How to multilingual info incorporated?

Multilingual Embeddings from [Klementiev et al., 2012]

Optimize independent neural language models,
with regularizer $\Omega = vec(word_i)^T \cdot A_{ij} \cdot vec(word_j)$
enforcing word vectors to be similar if alignment score A_{ij} is high:

$$\log p(en\_word_t \mid en\_word_{t-1}) + \log p(zh\_word_t \mid zh\_word_{t-1}) + \Omega$$

Example embeddings & nearby words:

january		*president*		*said*	
en	**de**	**en**	**de**	**en**	**de**
january	januar	president	präsident	said	sagte
february	februar	king	präsidenten	reported	erklärte
november	november	hun	minister	stated	sagten
april	april	areas	staatspräsident	told	meldete
august	august	saddam	hun	declared	berichtete
march	märz	minister	vorsitzenden	stressed	sagt
june	juni	advisers	us-präsident	informed	ergänzte
december	dezember	prince	könig	announced	erklärten
july	juli	representative	berichteten	explained	teilt
september	september	institutional	außenminister	warned	berichteten

[Zou et al., 2013] proposed similar model (different language model/regularizer), show MT gains by adding embedding similarity as translation model score

Discussion: Outlook on neural nets for MT

Active field! Still lots to try. e.g.

- Model tree/forest-based machine translation
- Even better decoder integration
- More synergy with compositional semantics
- Move beyond parallel bitext; exploit comparable corpora
- Improve existing work; experiments on more tasks by more researchers

Three main questions to consider if you want to start:

1. What to model? i.e. What is input/output of neural net?
2. How to setup training data? (Input/ouput is often not explicit in MT)
3. What kind of network and training algorithm? What are reasonable hyper-parameters to try? Details matter.

But also be humble! Lots of ideas hidden in older work, e.g.
[Castano et al., 1997, Jain et al., 1991]

Summary of entire talk

1. Deep Learning Background
   - Neural Networks (1-layer, 2-layer)
   - Potentials and Difficulties of Deep Architecture
   - The Breakthrough in 2006
2. Two Main Types of Deep Architectures
   - Deep Belief Nets (DBN) [Hinton et al., 2006]
   - Stacked Auto-Encoders (SAE) [Bengio et al., 2006]
   - Current Status of Deep Learning
3. Applications in Natural Language Processing and Machine Translation
   - Use as Non-linear classifier
   - Use as Distributed representation
   - Survey of Machine Translation Research

Final Thoughts

Is Deep Learning just a fad?

What ideas will stand the test of time?

Should I jump on the bandwagon?

To Learn More

- Survey paper:
  - Yoshua Bengio's [Bengio, 2009] short book: Learning Deep Architectures for AI[2]
- Courses & In-depth Lecture Notes:
  - My course @ NAIST: `http://cl.naist.jp/~kevinduh/a/deep2014/`
  - Hugo Larochelle's course @ Sherbrooke[3]
  - Geoff Hinton's Coursera course[4]
- Tutorials for NLPers:
  - Richard Socher et. al.'s NAACL2013 tutorial[5]
  - Ed Grefenstette et.al. ACL2014 Tutorial[6]
- To Learn Even More:
  - Theano code samples: `http://deeplearning.net/tutorial/`
  - Blog at `http://deeplearning.net`

[2] `http://www.iro.umontreal.ca/~bengioy/papers/ftml.pdf`
[3] `http://tinyurl.com/qccl66y`
[4] `https://www.coursera.org/course/neuralnets`
[5] `http://www.socher.org/index.php/DeepLearningTutorial/`
[6] `https://www.youtube.com/watch?v=_ASOqXiWBVo`

Thanks for your attention! Questions?

References:

Andreas, J. and Klein, D. (2014).
How much do word embeddings encode about syntax?
In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 822–827, Baltimore, Maryland. Association for Computational Linguistics.

Arisoy, E., Sainath, T. N., Kingsbury, B., and Ramabhadran, B. (2012).
Deep neural network language models.
In *Proceedings of the NAACL-HLT 2012 Workshop: Will We Ever Really Replace the N-gram Model? On the Future of Language Modeling for HLT*, pages 20–28, Montréal, Canada. Association for Computational Linguistics.

Auli, M., Galley, M., Quirk, C., and Zweig, G. (2013).
Joint language and translation modeling with recurrent neural networks.
In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1044–1054, Seattle, Washington, USA. Association for Computational Linguistics.

Auli, M. and Gao, J. (2014).
Decoder integration and expected bleu training for recurrent neural network language models.
In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 136–142, Baltimore, Maryland. Association for Computational Linguistics.

Bahdanau, D., Cho, K., and Bengio, Y. (2014).
Neural machine translation by jointly learning to align and translate.
CoRR, abs/1409.0473.

Bansal, M., Gimpel, K., and Livescu, K. (2014).
Tailoring continuous word representations for dependency parsing.
In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 809–815, Baltimore, Maryland. Association for Computational Linguistics.

Baroni, M., Dinu, G., and Kruszewski, G. (2014).
Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors.
In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 238–247, Baltimore, Maryland. Association for Computational Linguistics.

Bengio, Y. (2009).
Learning Deep Architectures for AI, volume Foundations and Trends in Machine Learning.
NOW Publishers.

Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. (2003).
A neural probabilistic language model.
JMLR.

Bengio, Y., Lamblin, P., Popovici, D., and Larochelle, H. (2006).
Greedy layer-wise training of deep networks.
In *NIPS'06*, pages 153–160.

Bergstra, J., Bardenet, R., Bengio, Y., and Kégel, B. (2011).
Algorithms for hyper-parameter optimization.
In *Proc. Neural Information Processing Systems 24 (NIPS2011)*.

Billingsley, R. and Curran, J. (2012).
Improvements to training an RNN parser.
In *Proceedings of COLING 2012*, pages 279–294, Mumbai, India. The COLING 2012 Organizing Committee.

Bishop, C. (1995).
Neural Networks for Pattern Recognition.
Oxford University Press.

Blitzer, J., Dredze, M., and Pereira, F. (2007).
Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification.
In *ACL*.

Bordes, A., Chopra, S., and Weston, J. (2014).
Question answering with subgraph embeddings.
In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 615–620, Doha, Qatar. Association for Computational Linguistics.

Carreira-Perpinan, M. A. and Hinton, G. E. (2005).
On contrastive divergence learning.
In *AISTATS*.

Castano, M. A., Casacuberta, F., and Vidal, E. (1997).
Machine translation using neural networks and finite-state models.
In *7ths Interantional Conference on Theoretical and Methodological Issues in Machine Translation (TMI)*.

Chandar, A. P. S., Lauly, S., Larochelle, H., Khapra, M. M., Ravindran, B., Raykar, V., and Saha, A. (2014).
An autoencoder approach to learning bilingual word representations.
In *NIPS*.

Chang, K.-W., Yih, W.-t., Yang, B., and Meek, C. (2014).
Typed tensor decomposition of knowledge bases for relation extraction.
In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1568–1579, Doha, Qatar. Association for Computational Linguistics.

Chen, D. and Manning, C. (2014).
A fast and accurate dependency parser using neural networks.
In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 740–750, Doha, Qatar. Association for Computational Linguistics.

Chen, W., Zhang, Y., and Zhang, M. (2014a).
Feature embeddings for dependency parsing.
In *Proceedings of COLING*.

Chen, X., Liu, Z., and Sun, M. (2014b).
A unified model for word sense representation and disambiguation.
In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1025–1035, Doha, Qatar. Association for Computational Linguistics.

Chen, Y., Perozzi, B., Al-Rfou, R., and Skiena, S. (2013).
The expressive power of word embeddings.
In *ICML 2013 Workshop on Deep Learning for Audio, Speech, and Language Processing*.

Cho, K., van Merrienboer, B., Gulcehre, C., Bougares, F., Schwenk, H., and Bengio, Y. (2014).
Learning phrase representations using rnn encoder-decoder for statistical machine translations.
In *Conference on Empirical Methods in Natural Language Processing (EMNLP) 2014*, number 1406.1078 in cs.CL.

Coates, A., Huval, B., Wang, T., Wu, D. J., Catanzaro, B., and Ng, A. Y. (2013).
Deep learning with COTS HPC systems.
In *Proceedings of the International Conference on Machine Learning (ICML)*.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. (2011).
Natural language processing (almost) from scratch.
Journal of Machine Learning Research, 12:2493–2537.

Cui, L., Zhang, D., Liu, S., Chen, Q., Li, M., Zhou, M., and Yang, M. (2014).
Learning topic representation for smt with neural networks.
In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 133–143, Baltimore, Maryland. Association for Computational Linguistics.

Dean, J., Corrado, G. S., Monga, R., Chen, K., Devin, M., Le, Q. V., Mao, M. Z., Ranzato, M., Senior, A., Tucker, P., Yang, K., and Ng, A. Y. (2012).
Large scale distributed deep networks.
In *Neural Information Processing Systems (NIPS)*.

Devlin, J., Zbib, R., Huang, Z., Lamar, T., Schwartz, R., and Makhoul, J. (2014).
Fast and robust neural network joint models for statistical machine translation.
In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1370–1380, Baltimore, Maryland. Association for Computational Linguistics.

Duh, K. and Kirchhoff, K. (2008).
Beyond log-linear models: Boosted minimum error rate training for n-best re-ranking.
In *Proceedings of ACL-08: HLT, Short Papers*, pages 37–40, Columbus, Ohio. Association for Computational Linguistics.

Duh, K., Neubig, G., Sudoh, K., and Tsukada, H. (2013).
Adaptation data selection using neural language models: Experiments in machine translation.
In *Proceedings of the 51th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Sofia, Bulgaria. Association for Computational Linguistics.

Erhan, D., Bengio, Y., Courville, A., Manzagol, P., Vincent, P., and Bengio, S. (2010).
Why does unsupervised pre-training help deep learning?
Journal of Machine Learning Research, 11:625–660.

Erhan, D., Manzagol, P., Bengio, Y., Bengio, S., and Vincent, P. (2009).
The difficulty of training deep architectures and the effect of unsupervised pre-training.
In *AISTATS*.

Faruqui, M. and Dyer, C. (2014).
Improving vector space word representations using multilingual correlation.
In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 462–471, Gothenburg, Sweden. Association for Computational Linguistics.

Fu, R., Guo, J., Qin, B., Che, W., Wang, H., and Liu, T. (2014).
Learning semantic hierarchies via word embeddings.
In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1199–1209, Baltimore, Maryland. Association for Computational Linguistics.

Fyshe, A., Talukdar, P. P., Murphy, B., and Mitchell, T. M. (2014).
Interpretable semantic vectors from a joint model of brain- and text- based meaning.
In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 489–499, Baltimore, Maryland. Association for Computational Linguistics.

Gao, J., He, X., Yih, W.-t., and Deng, L. (2014a).
Learning continuous phrase representations for translation modeling.
In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 699–709, Baltimore, Maryland. Association for Computational Linguistics.

Gao, J., Pantel, P., Gamon, M., He, X., and Deng, L. (2014b).
Modeling interestingness with deep neural networks.
In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2–13, Doha, Qatar. Association for Computational Linguistics.

Gardner, M., Talukdar, P., Krishnamurthy, J., and Mitchell, T. (2014).
Incorporating vector space similarity in random walk inference over knowledge bases.
In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 397–406, Doha, Qatar. Association for Computational Linguistics.

Glorot, X., Bordes, A., and Bengio, Y. (2011).
Domain adaptation for large-scale sentiment classication: A deep learning approach.
In *ICML*.

Guo, J., Che, W., Wang, H., and Liu, T. (2014).
Revisiting embedding features for simple semi-supervised learning.
In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 110–120, Doha, Qatar. Association for Computational Linguistics.

Hashimoto, K., Miwa, M., Tsuruoka, Y., and Chikayama, T. (2013).
Simple customization of recursive neural networks for semantic relation classification.
In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1372–1376, Seattle, Washington, USA. Association for Computational Linguistics.

Hashimoto, K., Stenetorp, P., Miwa, M., and Tsuruoka, Y. (2014).
Jointly learning word representations and composition functions using predicate-argument structures.
In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1544–1555, Doha, Qatar. Association for Computational Linguistics.

Hermann, K. M. and Blunsom, P. (2013).
The role of syntax in vector space models of compositional semantics.
In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 894–904, Sofia, Bulgaria. Association for Computational Linguistics.

Hermann, K. M. and Blunsom, P. (2014).
Multilingual models for compositional distributed semantics.
In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 58–68, Baltimore, Maryland. Association for Computational Linguistics.

Hermann, K. M., Das, D., Weston, J., and Ganchev, K. (2014).
Semantic frame identification with distributed word representations.
In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1448–1458, Baltimore, Maryland. Association for Computational Linguistics.

Hinton, G., Deng, L., Yu, D., Dahl, G., A.Mohamed, Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T., and Kingsbury, B. (2012a).
Deep neural networks for acoustic modeling in speech recognition.
IEEE Signal Processing Magazine, 29.

Hinton, G., Osindero, S., and Teh, Y.-W. (2006).
A fast learning algorithm for deep belief nets.
Neural Computation, 18:1527–1554.

Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2012b).
Improving neural networks by preventing co-adaptation of feature detectors.
CoRR, abs/1207.0580.

Hu, Y., Auli, M., Gao, Q., and Gao, J. (2014).
Minimum translation modeling with recurrent neural networks.
In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 20–29, Gothenburg, Sweden. Association for Computational Linguistics.

Irsoy, O. and Cardie, C. (2014).
Opinion mining with deep recurrent neural networks.
In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 720–728, Doha, Qatar. Association for Computational Linguistics.

Iyyer, M., Boyd-Graber, J., Claudino, L., Socher, R., and Daumé III, H. (2014).
A neural network for factoid question answering over paragraphs.
In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 633–644, Doha, Qatar. Association for Computational Linguistics.

Jain, A. N., Mcnair, A. E., Waibel, A., Saito, H., Hauptmann, A. G., and Tebelskis, J. (1991).
Connectionist and symbolic processing in speech-to-speech translation: the janus systems.
In *MT Summit*, volume 3, pages 113–117.

Ji, Y. and Eisenstein, J. (2014).
Representation learning for text-level discourse parsing.
In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13–24, Baltimore, Maryland. Association for Computational Linguistics.

Kalchbrenner, N. and Blunsom, P. (2013).
Recurrent continuous translation models.
In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1700–1709, Seattle, Washington, USA. Association for Computational Linguistics.

Kiela, D. and Bottou, L. (2014).
Learning image embeddings using convolutional neural networks for improved multi-modal semantics.
In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 36–45, Doha, Qatar. Association for Computational Linguistics.

Klementiev, A., Titov, I., and Bhattarai, B. (2012).
Inducing crosslingual distributed representations of words.
In *Proceedings of COLING 2012*, pages 1459–1474, Mumbai, India. The COLING 2012 Organizing Committee.

Kočiský, T., Hermann, K. M., and Blunsom, P. (2014).
Learning bilingual word representations by marginalizing alignments.
In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 224–229, Baltimore, Maryland. Association for Computational Linguistics.

Lauly, S., Boulanger, A., and Larochelle, H. (2013).
Learning multilingual word representations using a bag-of-words autoencoder.
In *NIPS 2013 Deep Learning Workshop*.

Le, P. and Zuidema, W. (2014).
The inside-outside recursive neural network model for dependency parsing.
In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 729–739, Doha, Qatar. Association for Computational Linguistics.

Le, Q. V., Ranzato, M., Monga, R., Devin, M., Chen, K., Corrado, G. S., Dean, J., and Ng, A. Y. (2012).
Building high-level features using large scale unsupervised learning.
In *ICML*.

Lee, H., Grosse, R., Ranganath, R., and Ng, A. (2009).
Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations.
In *ICML*.

Levy, O. and Goldberg, Y. (2014).
Dependency-based word embeddings.
In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 302–308, Baltimore, Maryland. Association for Computational Linguistics.

Li, J., Li, R., and Hovy, E. (2014a).
Recursive deep models for discourse parsing.
In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2061–2069, Doha, Qatar. Association for Computational Linguistics.

Li, P., Liu, Y., and Sun, M. (2013).
Recursive autoencoders for ITG-based translation.
In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 567–577, Seattle, Washington, USA. Association for Computational Linguistics.

Li, P., Liu, Y., Sun, M., Izuha, T., and Zhang, D. (2014b).
A neural reordering model for phrase-based translation.
In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1897–1907, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Liu, L., Watanabe, T., Sumita, E., and Zhao, T. (2013).
Additive neural networks for statistical machine translation.
In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 791–801, Sofia, Bulgaria. Association for Computational Linguistics.

Liu, S., Yang, N., Li, M., and Zhou, M. (2014).
A recursive recurrent neural network for statistical machine translation.
In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1491–1500, Baltimore, Maryland. Association for Computational Linguistics.

Liu, Y., hua Zhong, S., and Li, W. (2012).
Query-oriented multi-document summarization via unsupervised deep learning.
In *AAAI Conference on Artificial Intelligence*.

Lu, S., Chen, Z., and Xu, B. (2014).
Learning new semi-supervised deep auto-encoder features for statistical machine translation.
In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 122–132, Baltimore, Maryland. Association for Computational Linguistics.

Luong, T., Socher, R., and Manning, C. (2013).
Better word representations with recursive neural networks for morphology.
In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 104–113, Sofia, Bulgaria. Association for Computational Linguistics.

Ma, J., Zhang, Y., and Zhu, J. (2014).
Tagging the web: Building a robust web tagger with neural network.
In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 144–154, Baltimore, Maryland. Association for Computational Linguistics.

Martens, J. (2010).
Deep learning via Hessian-free optimization.
In *Proceedings of the 27th International Conference on Machine Learning (ICML)*.

Maskey, S. and Zhou, B. (2012).
Unsupervised deep belief features for speech translation.
In *ICASSP*.

Mikolov, T., Deoras, A., Povey, D., Burget, L., and Černocký, J. (2011).
Strategies for training large scale neural network language model.
In *ASRU*.

Mikolov, T., Karafiat, S., Burget, L., Černocký, J., and Khudanpur, S. (2010).
Recurrent neural network based language models.
In *Proceedings of the 11th Annual Conference of the International Speech Communication Association (INTERSPEECH 2010)*.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J. (2013).
Distributed representations of words and phrases and their compositionality.
In *NIPS*.

Milajevs, D., Kartsaklis, D., Sadrzadeh, M., and Purver, M. (2014).
Evaluating neural word representations in tensor-based compositional settings.
In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 708–719, Doha, Qatar. Association for Computational Linguistics.

Neelakantan, A., Shankar, J., Passos, A., and McCallum, A. (2014).
Efficient non-parametric estimation of multiple embeddings per word in vector space.
In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1059–1069, Doha, Qatar. Association for Computational Linguistics.

Niehues, J. and Waibel, A. (2013).
Continuous space language models using Restricted Boltzmann Machines.
In *IWLT*.

Pei, W., Ge, T., and Chang, B. (2014).
Max-margin tensor neural network for chinese word segmentation.
In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 293–303, Baltimore, Maryland. Association for Computational Linguistics.

Pennington, J., Socher, R., and Manning, C. (2014).
Glove: Global vectors for word representation.
In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Qi, Y., Das, S., Weston, J., and Collobort, R. (2014).
A deep learning framework for character-based information extraction.
In *Proceedings of the European Conference on Information Retrieval (ECIR)*.

Rocktäschel, T., Bošnjak, M., Singh, S., and Riedel, S. (2014).
Low-dimensional embeddings of logic.
In *Proceedings of the ACL 2014 Workshop on Semantic Parsing*, pages 45–49, Baltimore, MD. Association for Computational Linguistics.

Roth, M. and Woodsend, K. (2014).
Composition of word representations improves semantic role labelling.
In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 407–413, Doha, Qatar. Association for Computational Linguistics.

Salakhutdinov, R. and Hinton, G. (2009).
Deep Boltzmann machines.
In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, volume 5, pages 448–455.

Schwenk, H. (2012).
Continuous space translation models for phrase-based statistical machine translation.
In *COLING (Posters)*.

Schwenk, H., Rousseau, A., and Attik, M. (2012).
Large, pruned or continuous space language models on a gpu for statistical machine translation.
In *Proceedings of the NAACL-HLT 2012 Workshop: Will We Ever Really Replace the N-gram Model? On the Future of Language Modeling for HLT*, pages 11–19, Montréal, Canada. Association for Computational Linguistics.

Socher, R., Bauer, J., Manning, C. D., and Andrew Y., N. (2013a).
Parsing with compositional vector grammars.
In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 455–465, Sofia, Bulgaria. Association for Computational Linguistics.

Socher, R., Huang, E. H., Pennin, J., Ng, A. Y., and Manning, C. D. (2011).
Dynamic pooling and unfolding recursive autoencoders for paraphrase detection.
In *NIPS*.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A., and Potts, C. (2013b).
Recursive deep models for semantic compositionality over a sentiment treebank.
In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Son, L. H., Allauzen, A., and Yvon, F. (2012).
Continuous space translation models with neural networks.
In *NAACL*.

Songyot, T. and Chiang, D. (2014).
Improving word alignment using word similarity.
In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1840–1845, Doha, Qatar. Association for Computational Linguistics.

Srivastava, S., Hovy, D., and Hovy, E. (2013).
A walk-based semantically enriched tree kernel over distributed word representations.
In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1411–1416, Seattle, Washington, USA. Association for Computational Linguistics.

Stenetorp, P. (2013).
Transition-based dependency parsing using recursive neural networks.
In *NIPS 2013 Deep Learning Workhop*.

Sundermeyer, M., Alkhouli, T., Wuebker, J., and Ney, H. (2014).
Translation modeling with bidirectional recurrent neural networks.
In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 14–25, Doha, Qatar. Association for Computational Linguistics.

Sutskever, I., Martens, J., Dahl, G., and Hinton, G. (2013).
On the importance of initialization and momentum in deep learning.
In *Proceedings of the International Conference on Machine Learning (ICML)*.

Sutskever, I., Vinyals, O., and Le, Q. (2014).
Sequence to sequence learning with neural networks.
In *NIPS*.

Szeged, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2014).
Intriguing properties of neural networks.
In *International Conference on Learning Representations (ICLR)*.

Tamura, A., Watanabe, T., and Sumita, E. (2014).
Recurrent neural networks for word alignment model.
In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1470–1480, Baltimore, Maryland. Association for Computational Linguistics.

Tran, K. M., Bisazza, A., and Monz, C. (2014).
Word translation prediction for morphologically rich languages with bilingual neural networks.
In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1676–1688, Doha, Qatar. Association for Computational Linguistics.

Tsubaki, M., Duh, K., Shimbo, M., and Matsumoto, Y. (2013).
Modeling and learning semantic co-compositionality through prototype projections and neural networks.
In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 130–140, Seattle, Washington, USA. Association for Computational Linguistics.

Tsuboi, Y. (2014).
Neural networks leverage corpus-wide information for part-of-speech tagging.
In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 938–950, Doha, Qatar. Association for Computational Linguistics.

Turian, J., Ratinov, L.-A., and Bengio, Y. (2010).
Word representations: A simple and general method for semi-supervised learning.
In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394, Uppsala, Sweden. Association for Computational Linguistics.

Van de Cruys, T. (2014).
A neural network approach to selectional preference acquisition.
In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 26–35, Doha, Qatar. Association for Computational Linguistics.

Vaswani, A., Zhao, Y., Fossum, V., and Chiang, D. (2013).
Decoding with large-scale neural language models improves translation.
In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1387–1392, Seattle, Washington, USA. Association for Computational Linguistics.

Vinyals, O., Jia, Y., Deng, L., and Darrell, T. (2012).
Learning with recursive perceptual representations.
In *NIPS*.

Wang, M. and Manning, C. D. (2013).
Effect of non-linear deep architecture in sequence labeling.
In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 1285–1291, Nagoya, Japan. Asian Federation of Natural Language Processing.

Weston, J., Chopra, S., and Adams, K. (2014).
#tagspace: Semantic embeddings from hashtags.
In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1822–1827, Doha, Qatar. Association for Computational Linguistics.

Wu, H., Dong, D., Hu, X., Yu, D., He, W., Wu, H., Wang, H., and Liu, T. (2014a).
Improve statistical machine translation with context-sensitive bilingual semantic embedding model.
In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 142–146, Doha, Qatar. Association for Computational Linguistics.

Wu, Y., Watanabe, T., and Hori, C. (2014b).
Recurrent neural network-based tuple sequence model for machine translation.
In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1908–1917, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Yang, M.-C., Duan, N., Zhou, M., and Rim, H.-C. (2014).
Joint relational embeddings for knowledge-based question answering.
In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 645–650, Doha, Qatar. Association for Computational Linguistics.

Yang, N., Liu, S., Li, M., Zhou, M., and Yu, N. (2013).
Word alignment modeling with context dependent deep neural network.
In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 166–175, Sofia, Bulgaria. Association for Computational Linguistics.

Yih, W.-t., He, X., and Meek, C. (2014).
Semantic parsing for single-relation question answering.
In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 643–648, Baltimore, Maryland. Association for Computational Linguistics.

Zhang, J., Liu, S., Li, M., Zhou, M., and Zong, C. (2014).
Bilingually-constrained phrase embeddings for machine translation.
In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 111–121, Baltimore, Maryland. Association for Computational Linguistics.

Zhang, X. and Lapata, M. (2014).
Chinese poetry generation with recurrent neural networks.
In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 670–680, Doha, Qatar. Association for Computational Linguistics.

Zheng, X., Chen, H., and Xu, T. (2013).
Deep learning for Chinese word segmentation and POS tagging.
In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 647–657, Seattle, Washington, USA. Association for Computational Linguistics.

Zhila, A., Yih, W.-t., Meek, C., Zweig, G., and Mikolov, T. (2013).
Combining heterogeneous models for measuring relational similarity.
In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1000–1009, Atlanta, Georgia. Association for Computational Linguistics.

Zou, W. Y., Socher, R., Cer, D., and Manning, C. D. (2013).
Bilingual word embeddings for phrase-based machine translation.
In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1393–1398, Seattle, Washington, USA. Association for Computational Linguistics.