Representation Learning for Text

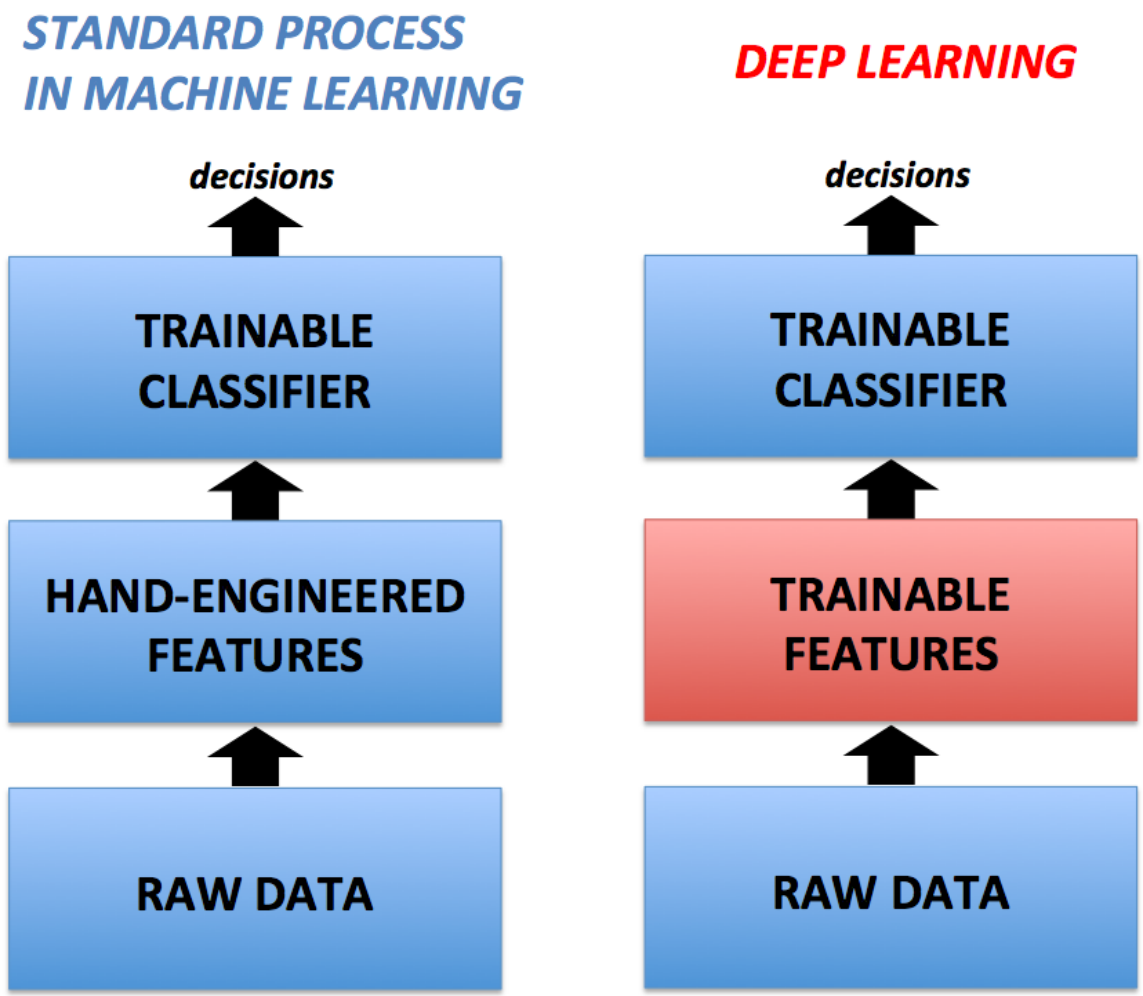
Kevin Duh

Johns Hopkins University

June 2017

Terminology

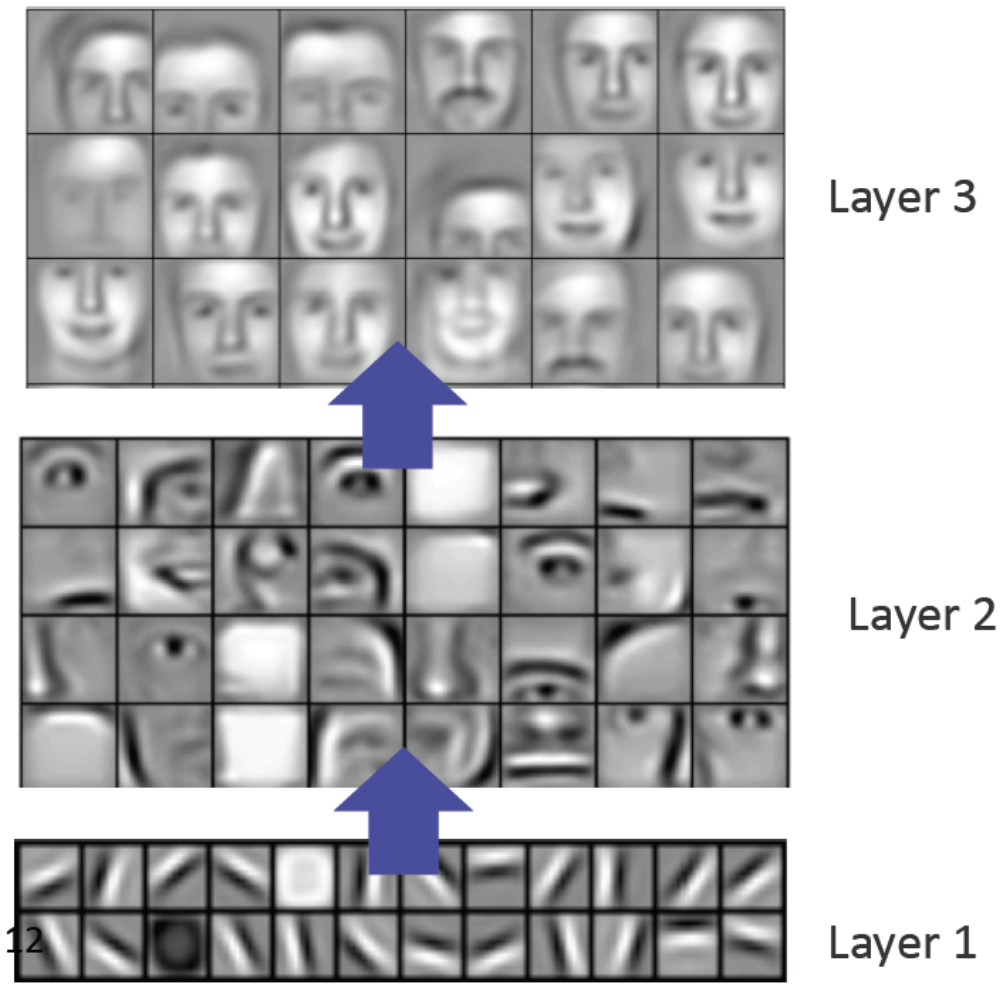
- Representation Learning is a problem statement
- Deep Learning is a solution technique
- Neural network is a way of implementing deep learning



Representation Learning in Images

Automatically trained features make sense! [Lee et al., 2009]

Input: Images (raw pixels)
$\rightarrow$ Output: Features of Edges, Body Parts, Full Faces



Outline

1. Word Embeddings
   - What should be encoded in a word representation?
   - Neural language model and word2vec
   - Multiview Learning

2. Sentence Embeddings
   - What should be encoded?
   - Handling variable length sequences

Outline

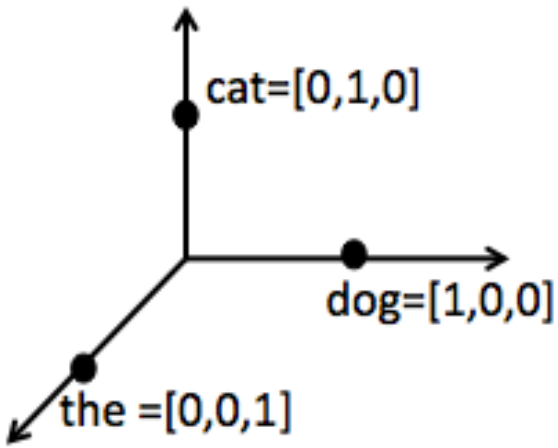
Outline

Representations of Words

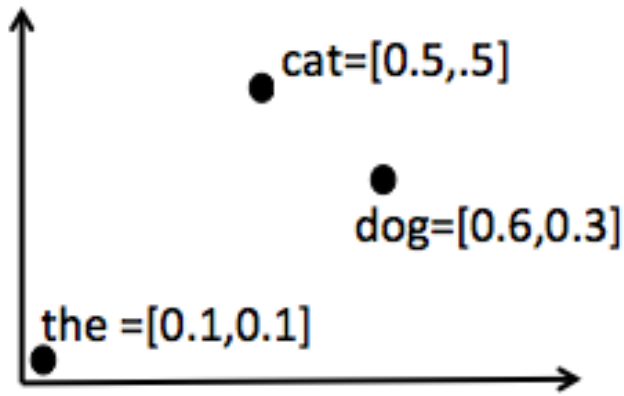
- Central question: How to computationally represent "words" in a natural language processing system?

Vector Representation of Words (word embeddings)

- 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]

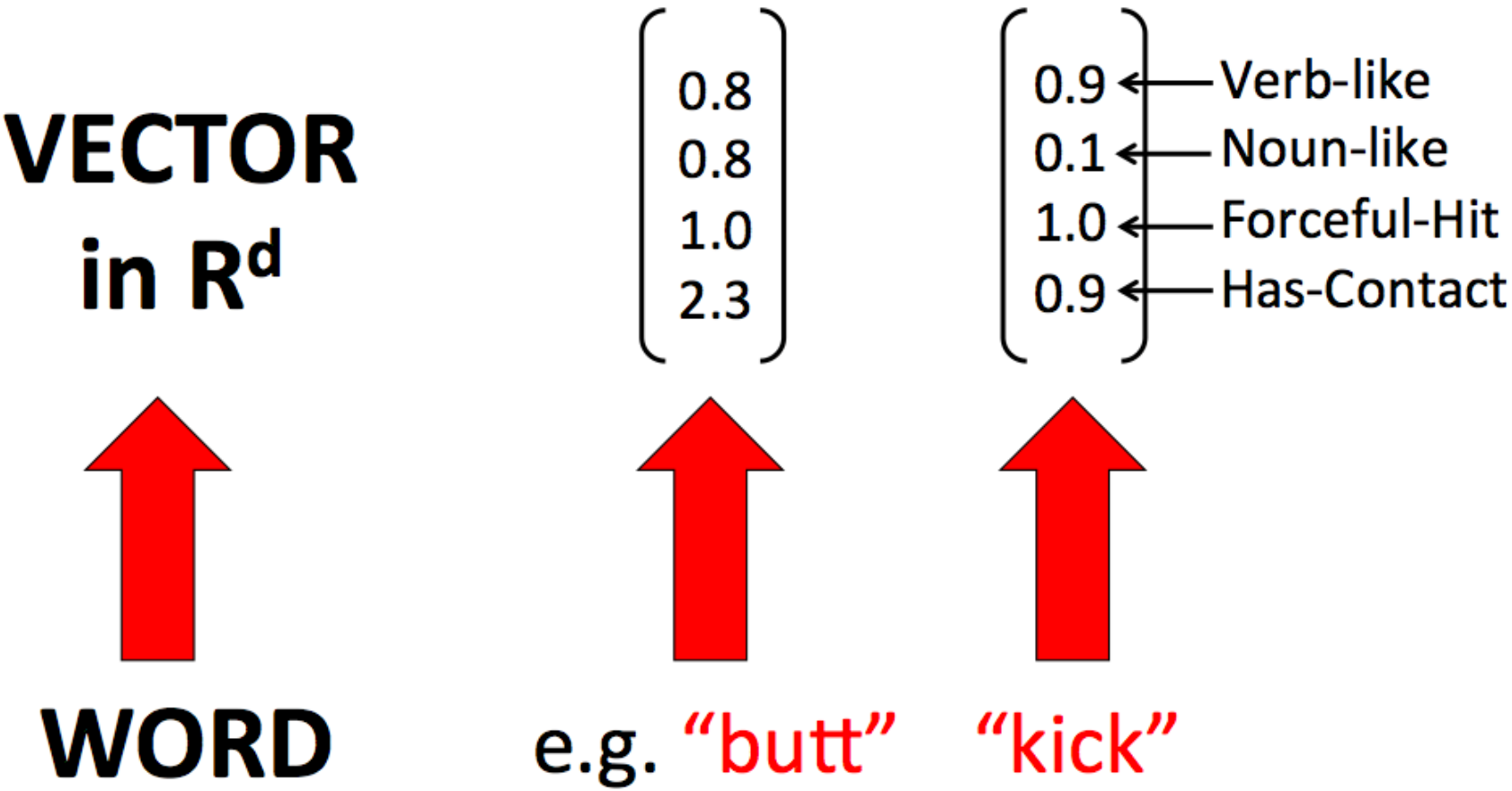


One-Hot Representation:
Word as discrete atomic feature



Distributed Representation:
Word as vectors

Vector Representations of Words (Word Embeddings)



Vector Representations of Words (Word Embeddings)

Vector representation:

$$\text{similarity}(\text{butt}=\begin{pmatrix}0.8\\0.3\\1.0\\2.3\end{pmatrix},\text{kick}=\begin{pmatrix}0.9\\0.1\\1.0\\0.9\end{pmatrix}) > 0$$

Suppose "butt" is a rare word

One-hot representation:

$$\text{similarity}(\text{butt}=\begin{pmatrix}0\\0\\\dots\\1\\0\end{pmatrix},\text{kick}=\begin{pmatrix}0\\0\\\dots\\0\\1\end{pmatrix}) = 0$$

What information is needed for a good representation?

You shall know a word by the company it keeps. – J. R. Firth (linguist)

- Distributional Semantics: a word's meaning is based on its positional distribution in text

Distributional Semantics

Large Text Dataset

Your pet dog is so cute

Your pet cat is so cute

The dog ate my homework

The cat ate my homework

→ dog is-similar-to cats (in usage & meaning)

Example: "Bar-ba-loots"

- Do you know what is a bar-ba-loot?

Example: "Bar-ba-loots"

- Do you know what is a bar-ba-loot?
- Which has higher probability?
  - $P(\mathbf{w_t} = \textit{fruits} \mid \mathbf{w_{t-1}} = \textit{like}, \mathbf{w_{t-2}} = \textit{Bar-ba-loots})$
  - $P(\mathbf{w_t} = \textit{looting} \mid \mathbf{w_{t-1}} = \textit{like}, \mathbf{w_{t-2}} = \textit{Bar-ba-loots})$

Example: "Bar-ba-loots"

- Do you know what is a bar-ba-loot?
- Which has higher probability?
  - $P(\mathbf{w_t} = \textit{fruits} \mid \mathbf{w_{t-1}} = \textit{like}, \mathbf{w_{t-2}} = \textit{Bar-ba-loots})$
  - $P(\mathbf{w_t} = \textit{looting} \mid \mathbf{w_{t-1}} = \textit{like}, \mathbf{w_{t-2}} = \textit{Bar-ba-loots})$
- What if I tell you **vector**(Bar-ba-loots) $\sim$ **vector**(bears)?

Example: "Bar-ba-loots"

- Do you know what is a bar-ba-loot?
- Which has higher probability?
  - $P(\mathbf{w_t} = \textit{fruits} \mid \mathbf{w_{t-1}} = \textit{like}, \mathbf{w_{t-2}} = \textit{Bar-ba-loots})$
  - $P(\mathbf{w_t} = \textit{looting} \mid \mathbf{w_{t-1}} = \textit{like}, \mathbf{w_{t-2}} = \textit{Bar-ba-loots})$
- What if I tell you **vector**(Bar-ba-loots) $\sim$ **vector**(bears)?

- Purpose of word embedding: Improve generalization

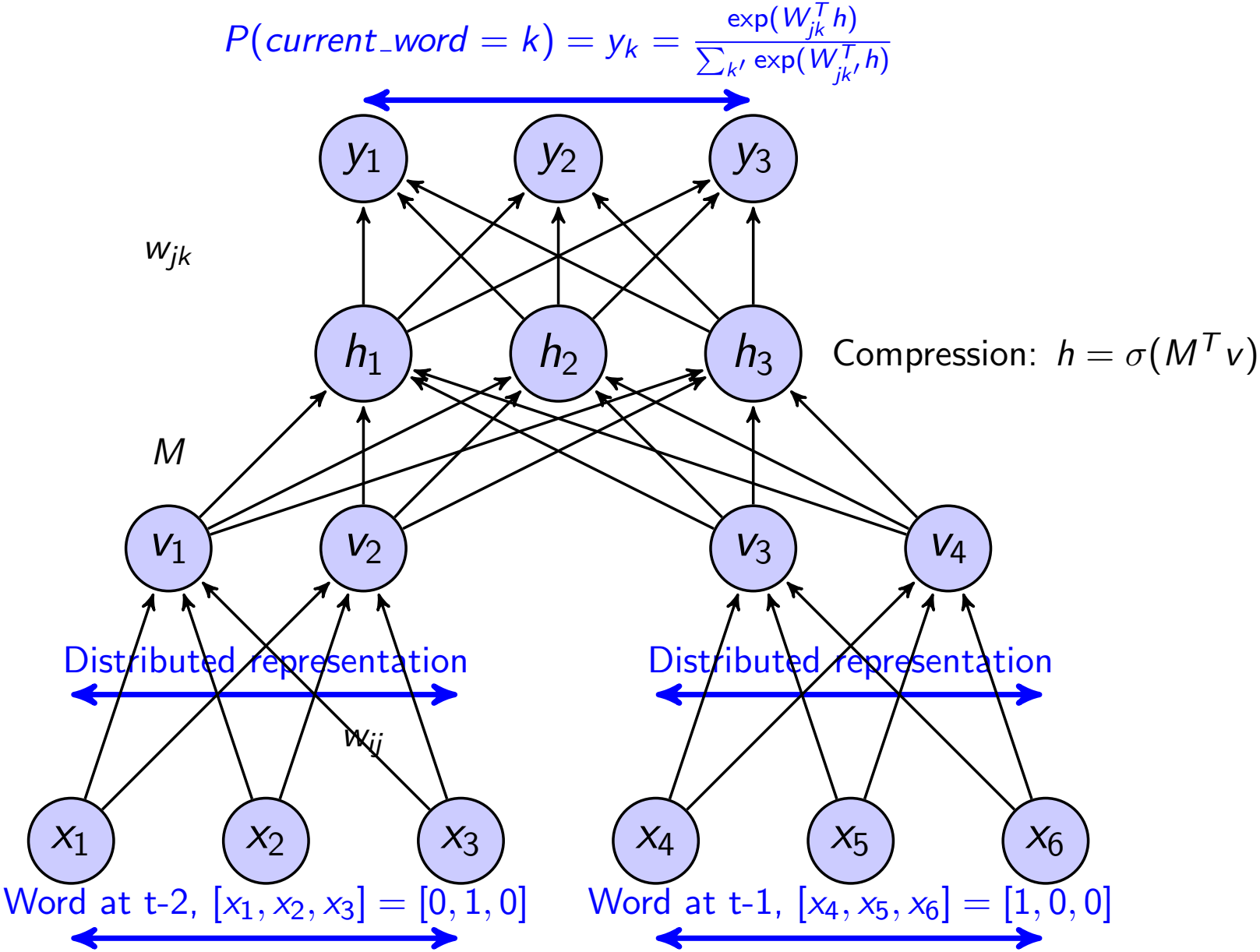
Outline

Neural language model and word2vec

- Feed-forward neural language model
  - Model $P(\mathbf{w_t} = \textit{looting} \mid \mathbf{w_{t-1}} = \textit{like}, \mathbf{w_{t-2}} = \textit{Bar-ba-loots})$ with a feed-forward neural LM
  - Can be used a LM inside machine translation / speech recognition systems
  - Or, can extract first layer as word embedding
- word2vec
  - If we only want word embeddings and don't need a probability distribution over next words, we can save on training time

Word embeddings from Neural LM, one method

[Bengio et al., 2003]



Training a Neural LM

- Find parameters θ that maximizes likelihood $P_\theta(\mathbf{w_t} \mid \mathbf{w_{t-1}}, \mathbf{w_{t-2}})$ on training data
  - For each input $\mathbf{x_t} = (\mathbf{w_{t-1}}, \mathbf{w_{t-2}})$ and output $\mathbf{w_t}$ pair, forward compute $P_\theta(\cdot)$ and then back-prop

Training a Neural LM

- Find parameters θ that maximizes likelihood $P_\theta(\mathbf{w_t} \mid \mathbf{w_{t-1}}, \mathbf{w_{t-2}})$ on training data
  - For each input $\mathbf{x_t} = (\mathbf{w_{t-1}}, \mathbf{w_{t-2}})$ and output $\mathbf{w_t}$ pair, forward compute $P_\theta(\cdot)$ and then back-prop
- $P_\theta(\mathbf{w_t} \mid \mathbf{x_t}) = \mathit{softmax}(\mathit{score}(\mathbf{w_t}, \mathbf{x_t})) = \frac{\exp(\mathit{score}(\mathbf{w_t}, \mathbf{h_t}))}{\sum_{\mathbf{w'} \in \mathit{Vocabulary}} \exp(\mathit{score}(\mathbf{w'}, \mathbf{x_t}))}$
- max-likelihood objective:
  $\log P_\theta(\mathbf{w_t} \mid \mathbf{x_t}) = \mathit{score}(\mathbf{w_t}, \mathbf{x_t}) - \log \sum_{\mathbf{w'} \in \mathit{Vocabulary}} \exp(\mathit{score}(\mathbf{w'}, \mathbf{x_t}))$
  - challenge: sum over all vocabulary for each gradient step may be expensive.

word2vec [Mikolov et al., 2013]

1. Different objective (negative sampling): discriminate target word from a few (k) noise words sampled from distribution q:
   - $\log p(D = 1 \mid \mathbf{w_t}, \mathbf{x_t}) + k\mathbb{E}_{w' \sim q} \log p(D = 0 \mid \mathbf{w}', \mathbf{x_t})$
   - binary classifier discriminates true and noise samples:
     $p(D = 1 \mid \mathbf{w_t}, \mathbf{x_t}) = \frac{score(\mathbf{w_t}, \mathbf{x_t})}{score(\mathbf{w_t}, \mathbf{x_t})+1}$; $p(D = 0 \mid \mathbf{w_t}, \mathbf{x_t}) = \frac{1}{score(\mathbf{w_t}, \mathbf{x_t})+1}$

word2vec [Mikolov et al., 2013]

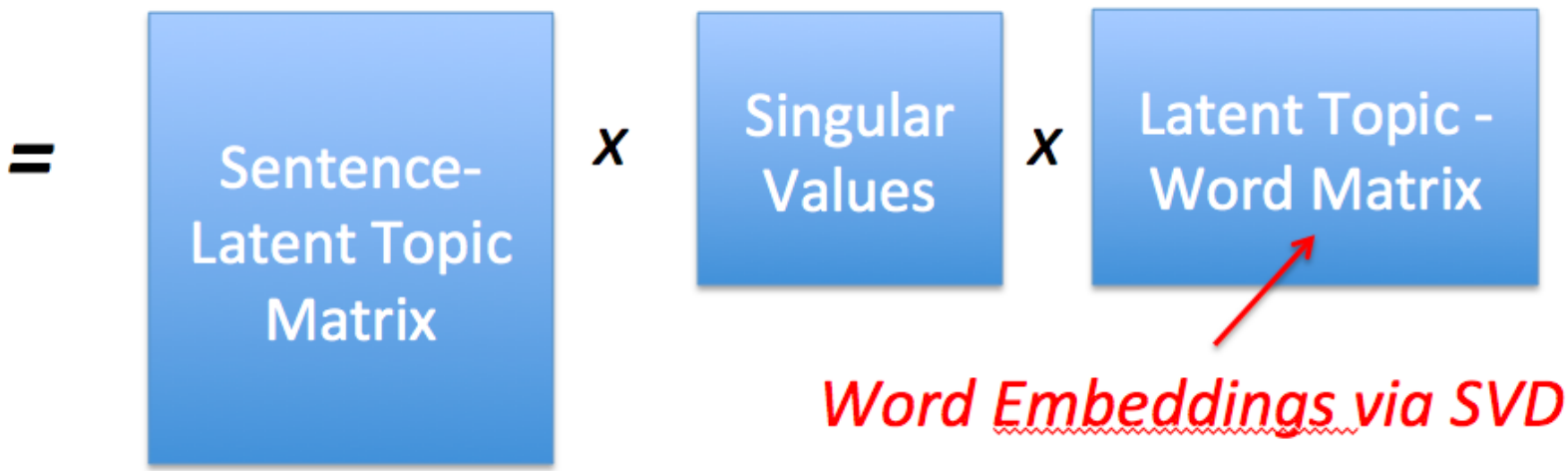
1. Different objective (negative sampling): discriminate target word from a few (k) noise words sampled from distribution q:
   - $\log p(D = 1 \mid \mathbf{w_t}, \mathbf{x_t}) + k\mathbb{E}_{w' \sim q} \log p(D = 0 \mid \mathbf{w}', \mathbf{x_t})$
   - binary classifier discriminates true and noise samples: $p(D = 1 \mid \mathbf{w_t}, \mathbf{x_t}) = \frac{score(\mathbf{w_t}, \mathbf{x_t})}{score(\mathbf{w_t}, \mathbf{x_t})+1}$; $p(D = 0 \mid \mathbf{w_t}, \mathbf{x_t}) = \frac{1}{score(\mathbf{w_t}, \mathbf{x_t})+1}$
2. Simplified architecture:
   - $score(\mathbf{w_t}, \mathbf{x_t}) = \exp(\mathbf{vec}(\mathbf{w_t})^T \cdot \mathbf{vec}(\mathbf{x_t}))$
   - skip-gram variant: $\mathbf{x_t}$ is input, predict left/right neighbors
   - note: input/output embeddings may differ

Learning Distributional Semantics via SVD

Sentence-Term Matrix

	pet	dog	is	cat	the	ate
S1	1	1	1	0	0	0
S2	1	0	1	1	0	0
S3	0	1	0	0	1	1
S4	0	0	0	1	1	1

S1: ... *pet dog is* ...
S2: ... *pet cat is* ...
S3: *The dog ate* ...
S4: *The cat ate* ...



Outline

From single-view to multi-view

- What are other "definitions" of word meaning, besides distributional semantics?
- Can we include them in the encoding of word representations?

View 2: Relational Semantics

dog is-a-kind-of canine

poodle is-a-kind-of dog

dog has-a tail

dog barking sounds-like woof-woof!

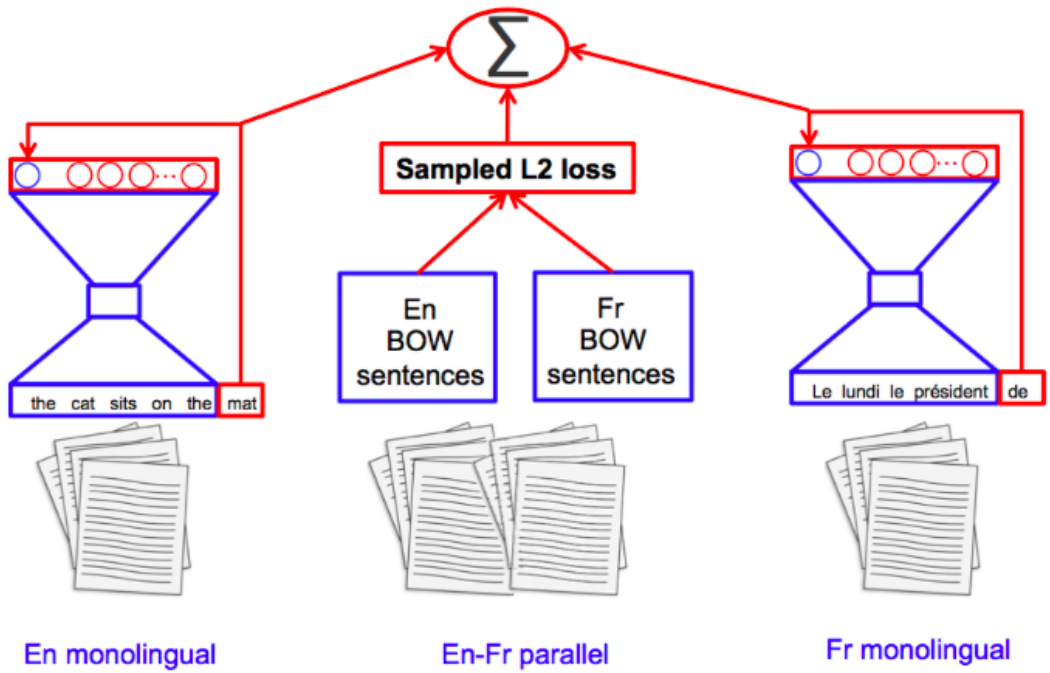
- Many such relational knowledge have been created in e.g. WordNet [Miller, 1995], Freebase [Bollacker et al., 2008]
- [Faruqui et al., 2014, Fried and Duh, 2014] incorporate both relational and distributional objectives in embeddings

View 3: Bilingual information

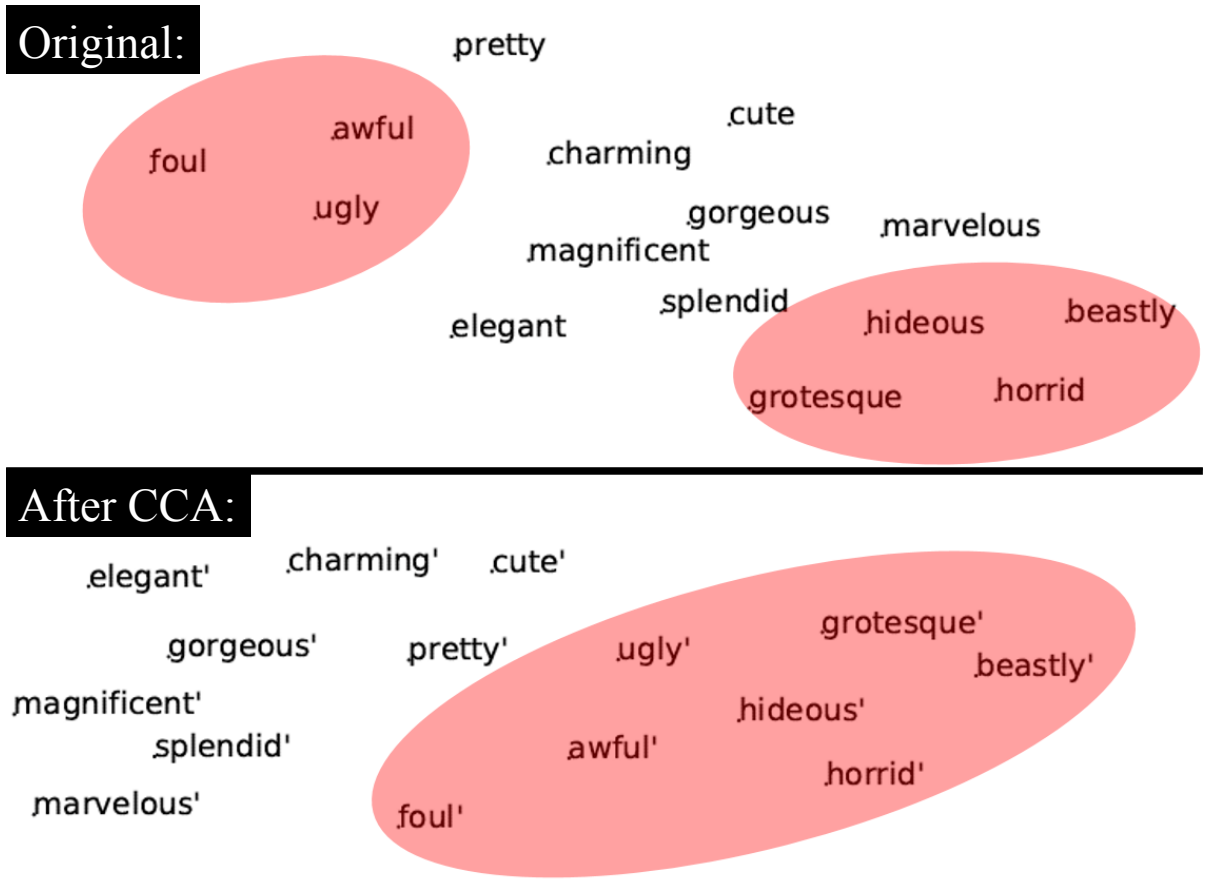
- The translation y of a word x provides an additional view (different form, same meaning)
- Method 1: Assume word translation pairs [Faruqui and Dyer, 2014]:
  - $v, w = CCA(x, y) = \arg\max_{v,w} \rho(xv, yw)$

View 3: Bilingual information

- The translation y of a word x provides an additional view (different form, same meaning)
- Method 1: Assume word translation pairs [Faruqui and Dyer, 2014]:
  - $v, w = CCA(x, y) = \arg\max_{v,w} \rho(xv, yw)$
- Method 2: Assume sentence translation pairs [Gouws et al., 2015]

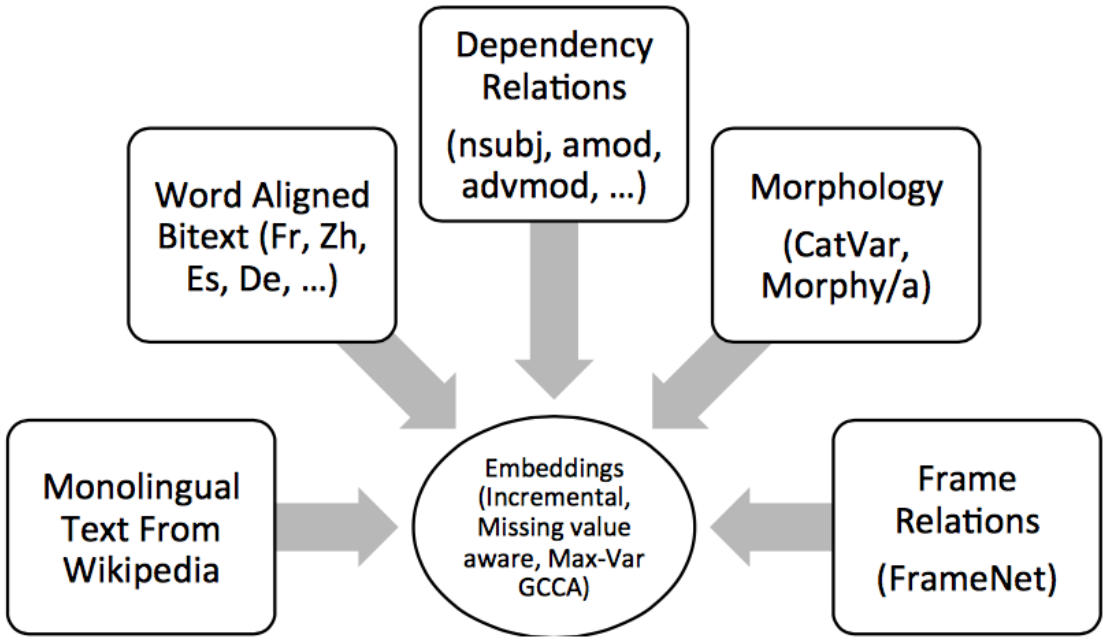


Example [Faruqui and Dyer, 2014]



Antonyms are mixed up when relying only on one language's distributional semantics (view 1), but is nicely teased apart with multilingual views.

2+ views [Rastogi et al., 2015]



$$\arg\min_{G,U_j} \sum_{j=1}^{J} \left\| G - X_j U_j \right\|_F^2$$

$$\text{subject to } G^\top G = I.$$

Outline

1. Word Embeddings
   - What should be encoded in a word representation?
   - Neural language model and word2vec
   - Multiview Learning

2. Sentence Embeddings
   - What should be encoded?
   - Handling variable length sequences

Outline

1. Word Embeddings
   - What should be encoded in a word representation?
   - Neural language model and word2vec
   - Multiview Learning

2. Sentence Embeddings
   - What should be encoded?
   - Handling variable length sequences

Challenges

1. What to encode in the representation?
2. How to handle variable length sequences?

What should be encoded? (1) Semantic equivalence

- "Bob gave Jane a book on her birthday"
  $\rightarrow$ vector $\mathbf{s_1}$
- "Jane's birthday present from Bob was a book"
  $\rightarrow$ vector $\mathbf{s_2}$
- Hopefully $\mathbf{s_1} = \mathbf{s_2}$

What should be encoded? (1) Semantic equivalence

- "Bob gave Jane a book on her birthday"
  $\rightarrow$ vector $\mathbf{s_1}$
- "Jane's birthday present from Bob was a book"
  $\rightarrow$ vector $\mathbf{s_2}$
- Hopefully $\mathbf{s_1} = \mathbf{s_2}$

- "Bob gave Jane a book on her birthday"
  $\rightarrow$ vector $\mathbf{s_1}$
- "Bob gab Jane ein Buch an ihren Geburstag"
  $\rightarrow$ vector $\mathbf{s_3}$
- Hopefully $\mathbf{s_1} = \mathbf{s_3}$

What should be encoded? (2) Entailment, Relatedness, and Inference

- s1: A sea turtle is hunting for fish
- s2: A sea turtle is hunting for food
- s3: The turtle is following the fish
- s4: The turtle is following the red fish

Does s1 entail s2? Are they semantically equivalent?

Does s1 entail s3? Or can you infer s3 is likely true given s1?

What should be encoded? (2) Entailment, Relatedness, and Inference

- s5: The first settlements on the site of Jakarta were established at the mouth of the Ciliwung, perhaps as early as the 5th century AD.
- s6: The first settlements on the site of Jakarta were established as early as the 5th century AD.
- s7: The first settlements on the site of Jakarta were established as late as the 4th century AD.

s6 summarizes s5. How should the vectors relate mathematically?

s6 and s7 are syntactically similar, but contradictory. What should be their vectors?

What should be encoded? (2) Entailment, Relatedness, and Inference

- s7: Tom was accidentally shot by his teammate in the Army
- s8: Tom dies
- s9: Tom is bleeding

s8 is a possible consequence of s7 by inference. How should the vectors relate?

s9 is an even more likely consequence of s7 by inference. How do the three vectors relate?

The cynic

Does it even make sense to represent one sentence as one vector?

One possible work-around: attention

- Representation is $\sum_i \alpha_i v_i$ where $0 \leq \alpha_i \leq 1, v_i \in \mathbb{R}^d$
- Integrate representation in an end-to-end system, and α_i is adaptively re-computed based on some state

Attention in machine translation [Bahdanau et al., 2014]

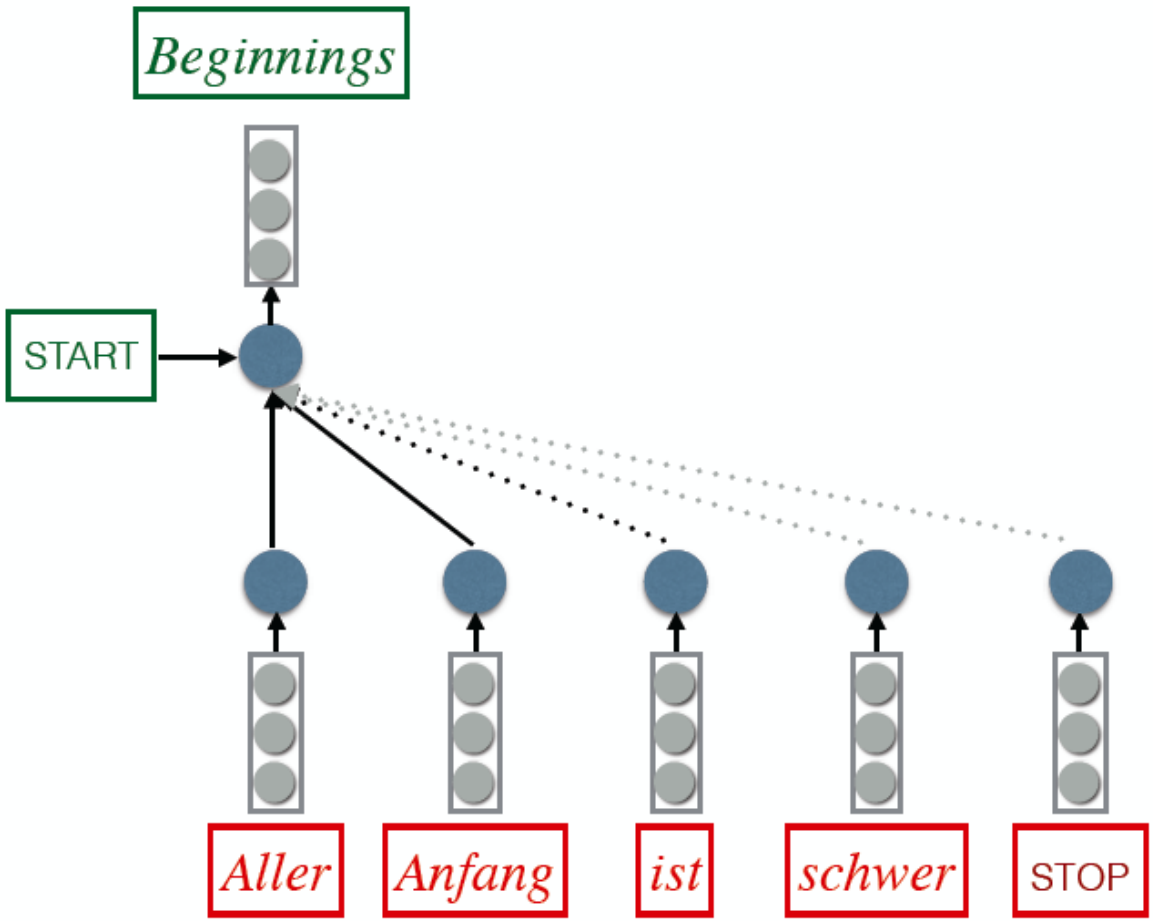
- It seems too much to expect one vector encoding of input $\mathbf{c}$ can do everything

Attention in machine translation [Bahdanau et al., 2014]

- It seems too much to expect one vector encoding of input $\mathbf{c}$ can do everything
- Idea: During decoding, dynamically pay attention to different parts of the input.

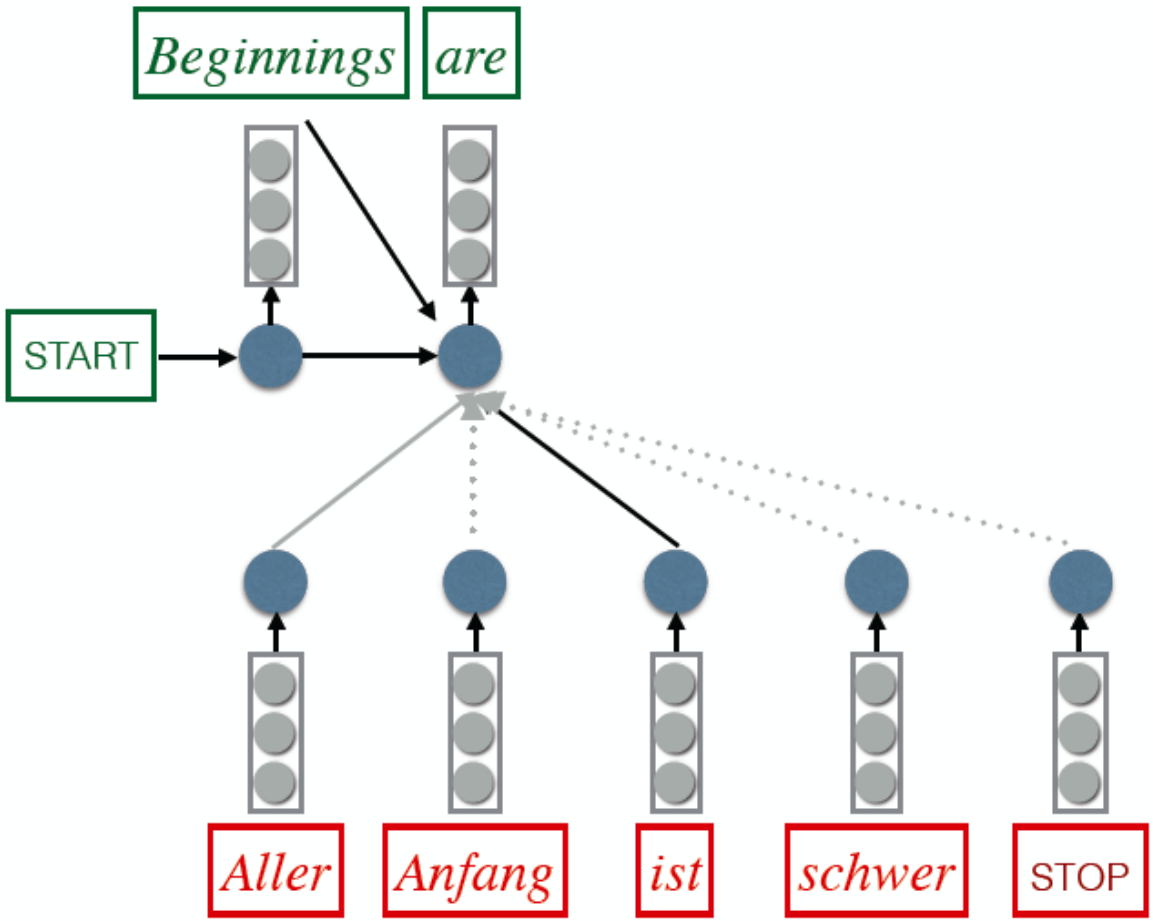
Attention in machine translation [Bahdanau et al., 2014]

- It seems too much to expect one vector encoding of input $\mathbf{c}$ can do everything
- Idea: During decoding, dynamically pay attention to different parts of the input.



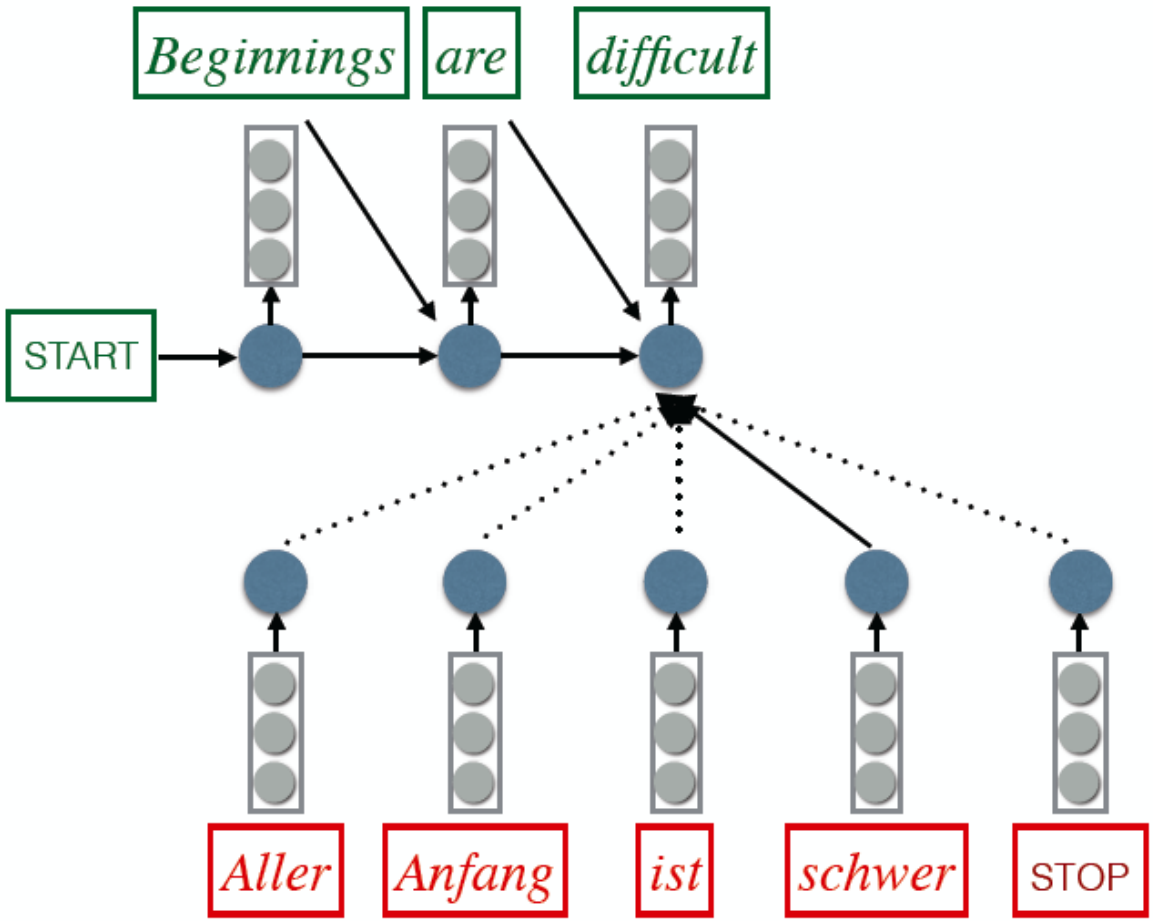
Attention in machine translation [Bahdanau et al., 2014]

- It seems too much to expect one vector encoding of input $\mathbf{c}$ can do everything
- Idea: During decoding, dynamically pay attention to different parts of the input.



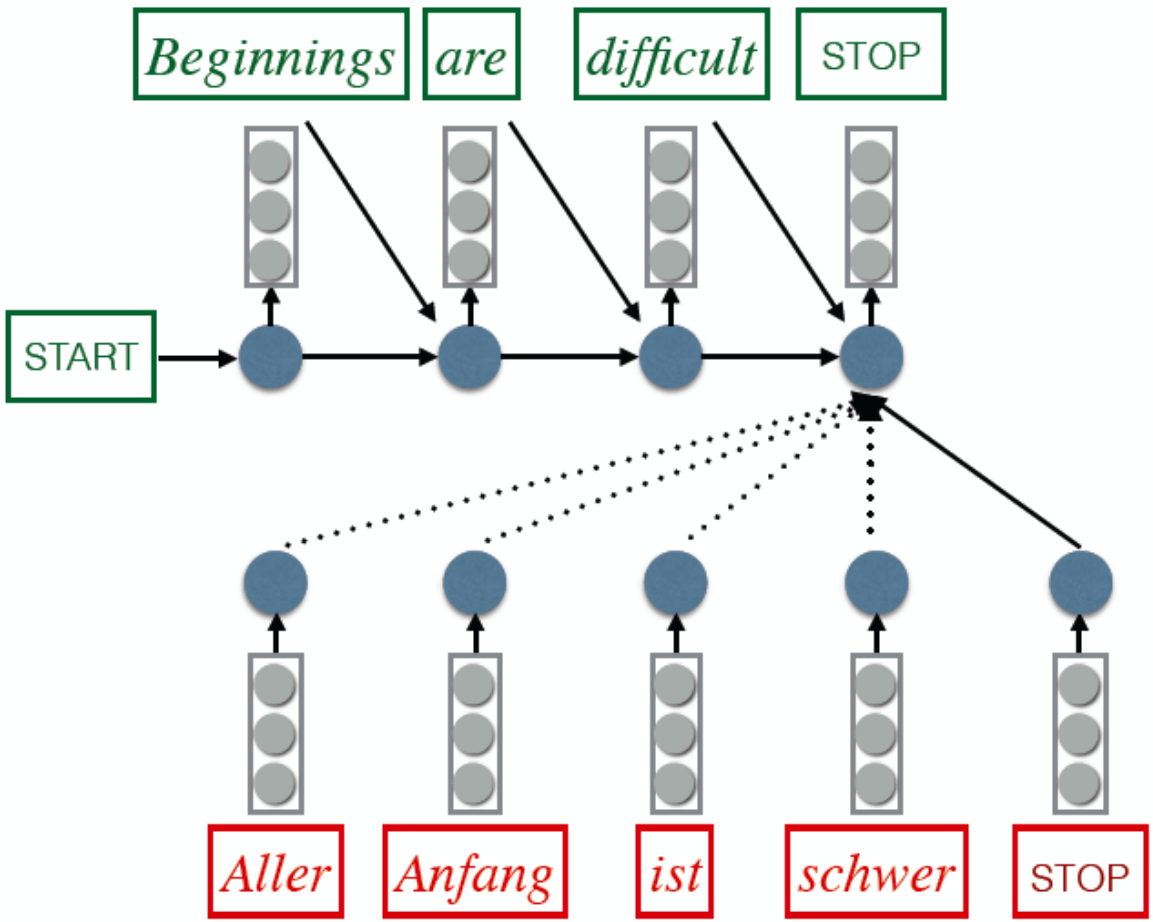
Attention in machine translation [Bahdanau et al., 2014]

- It seems too much to expect one vector encoding of input $\mathbf{c}$ can do everything
- Idea: During decoding, dynamically pay attention to different parts of the input.



Attention in machine translation [Bahdanau et al., 2014]

- It seems too much to expect one vector encoding of input $\mathbf{c}$ can do everything
- Idea: During decoding, dynamically pay attention to different parts of the input.



Outline

1. Word Embeddings
   - What should be encoded in a word representation?
   - Neural language model and word2vec
   - Multiview Learning

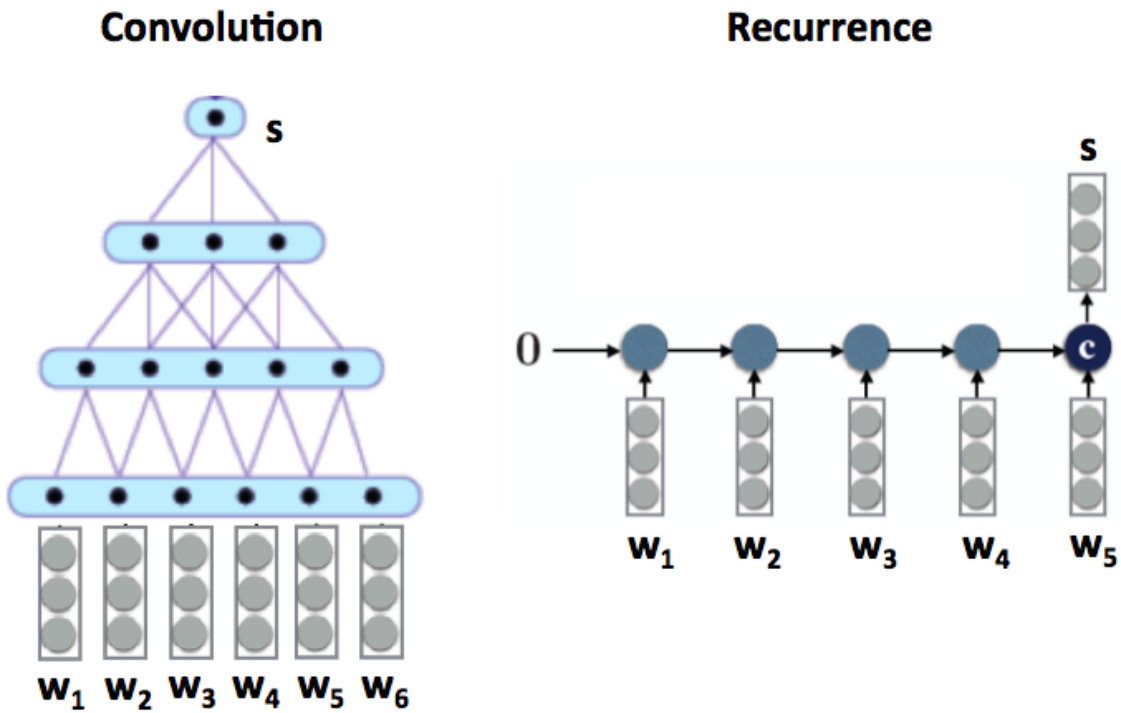
2. Sentence Embeddings
   - What should be encoded?
   - Handling variable length sequences

Approach 1: Bag of words

- Sentence vector = sum of word vectors
- $\mathbf{s} = w(Bob) + w(gave) + w(Jane) + w(a) + w(book) + w(on) + w(her) + w(birthday)$
- Variants: product, weighted sum, bag of bigrams, ...

Approach 2: Convolution and Recurrent units

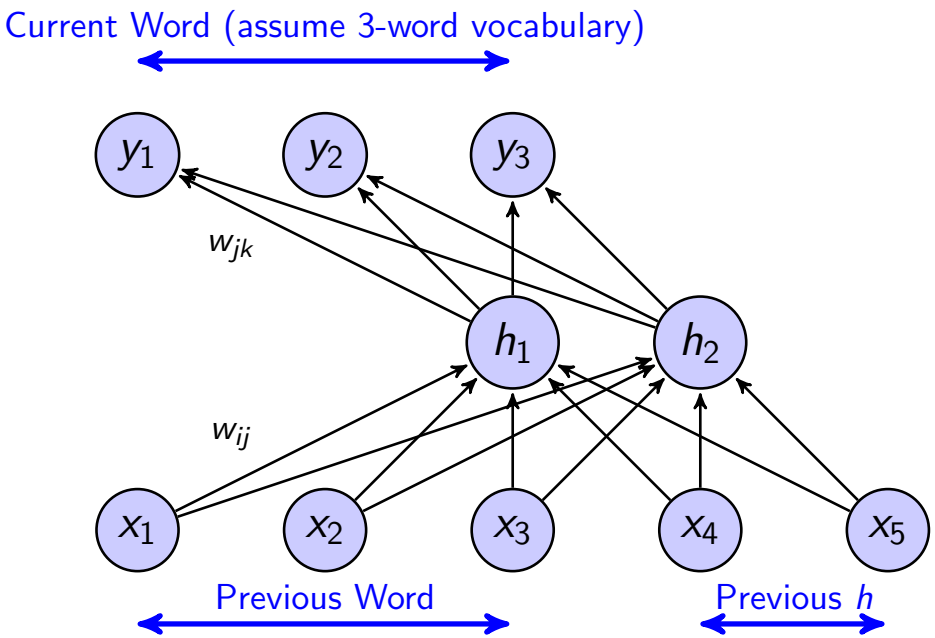
- Encode variable-length sequence into fix-length vector



- Examples in sentence translation
  [Kalchbrenner and Blunsom, 2013, Sutskever et al., 2014]

Recurrent Neural Net Language Models

Model $p(\textit{current\_word}|\textit{previous\_words})$ with a recurrent hidden layer [Mikolov et al., 2010]

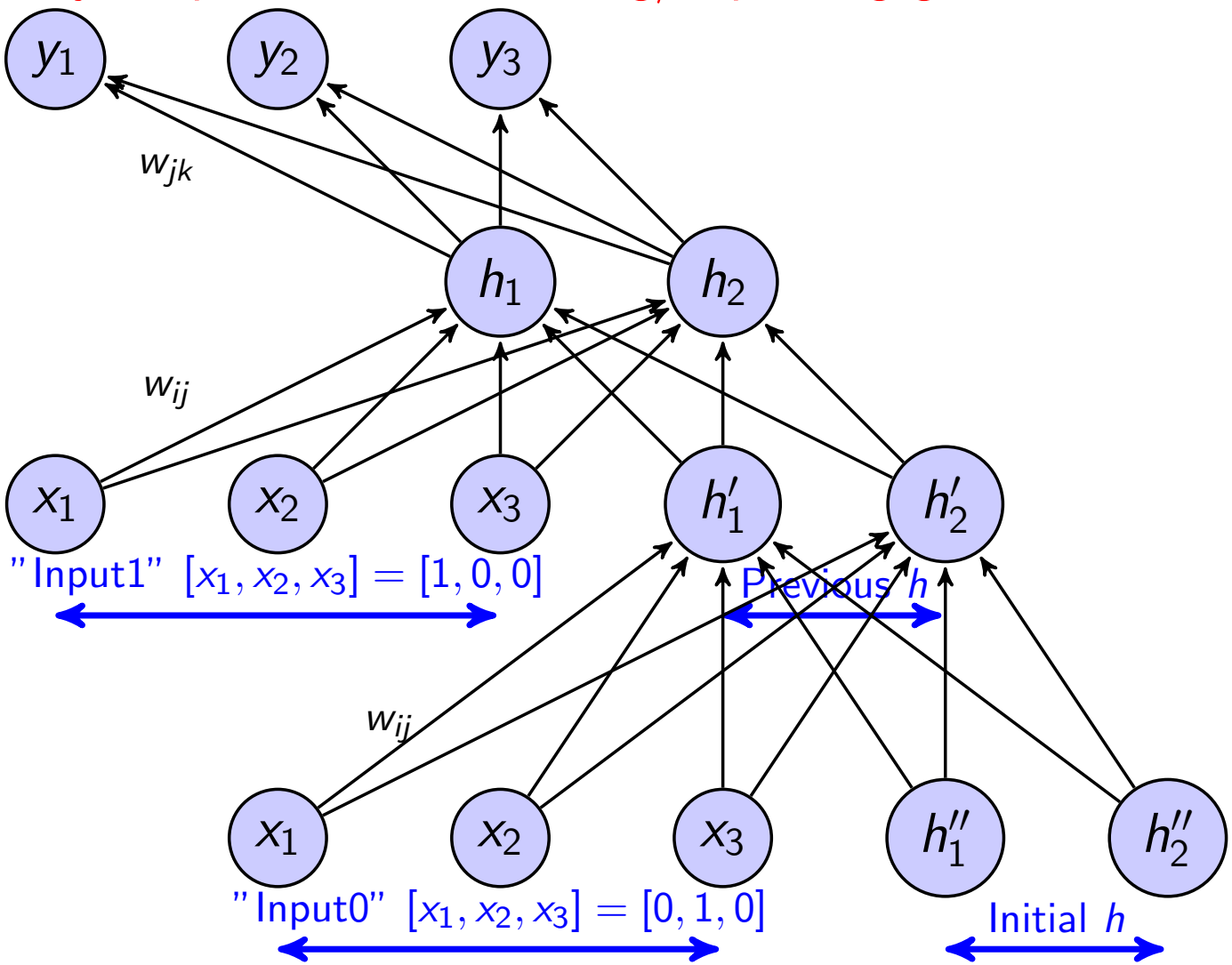


- Probability of word: $y_k = \frac{\exp(W_{jk}^T h)}{\sum_{k'} \exp(W_{jk'}^T h)}$
- $[x_4, x_5]$ is a copy of $[h_1, h_2]$ from the previous time-step
- $h_j = \sigma(W_{ij}^T x_i)$ is hidden state of partial sentence
- Arbitrarily-long history is (theoretically) kept through recurrence

Training: Backpropagation through Time

Unroll the hidden states for a few time-steps, then backprop.

Very deep network: vanishing/exploding gradients!



A popular recurrent unit: LSTM

Long Short-term Memory (LSTM):

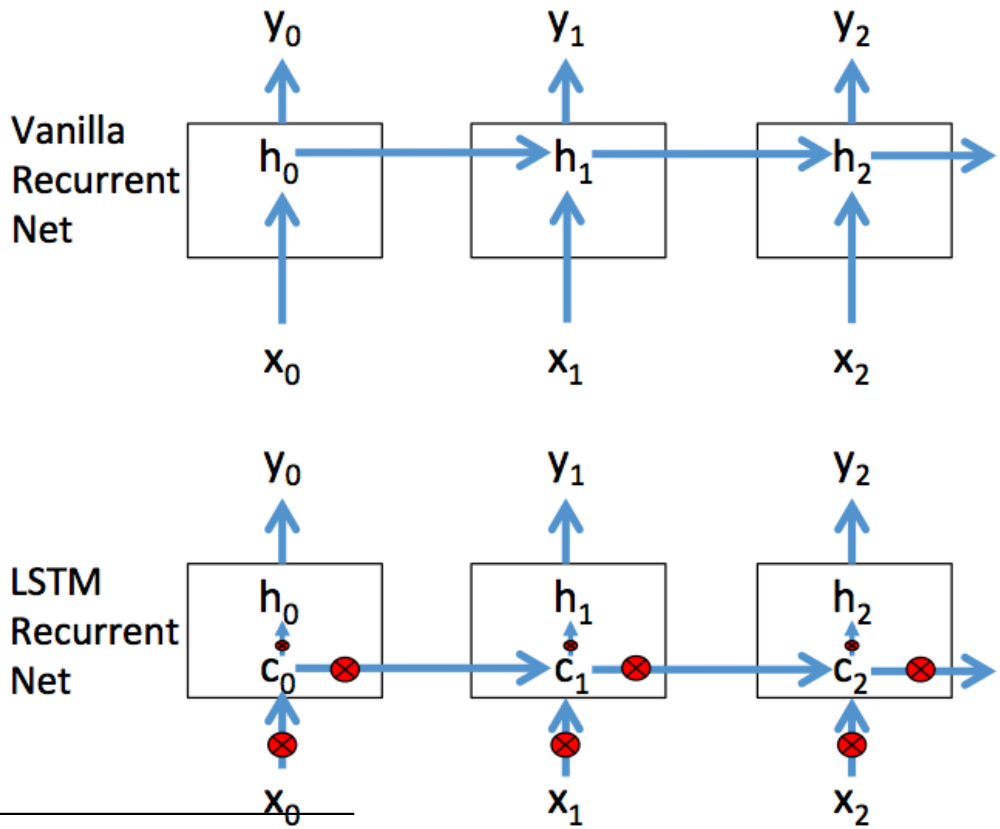
- Idea: Shouldn't need to back-prop to beginning of history. Just keep a memory of important bits.

*Nice explanation of LSTM and variants here:
`http://colah.github.io/posts/2015-08-Understanding-LSTMs/`

A popular recurrent unit: LSTM

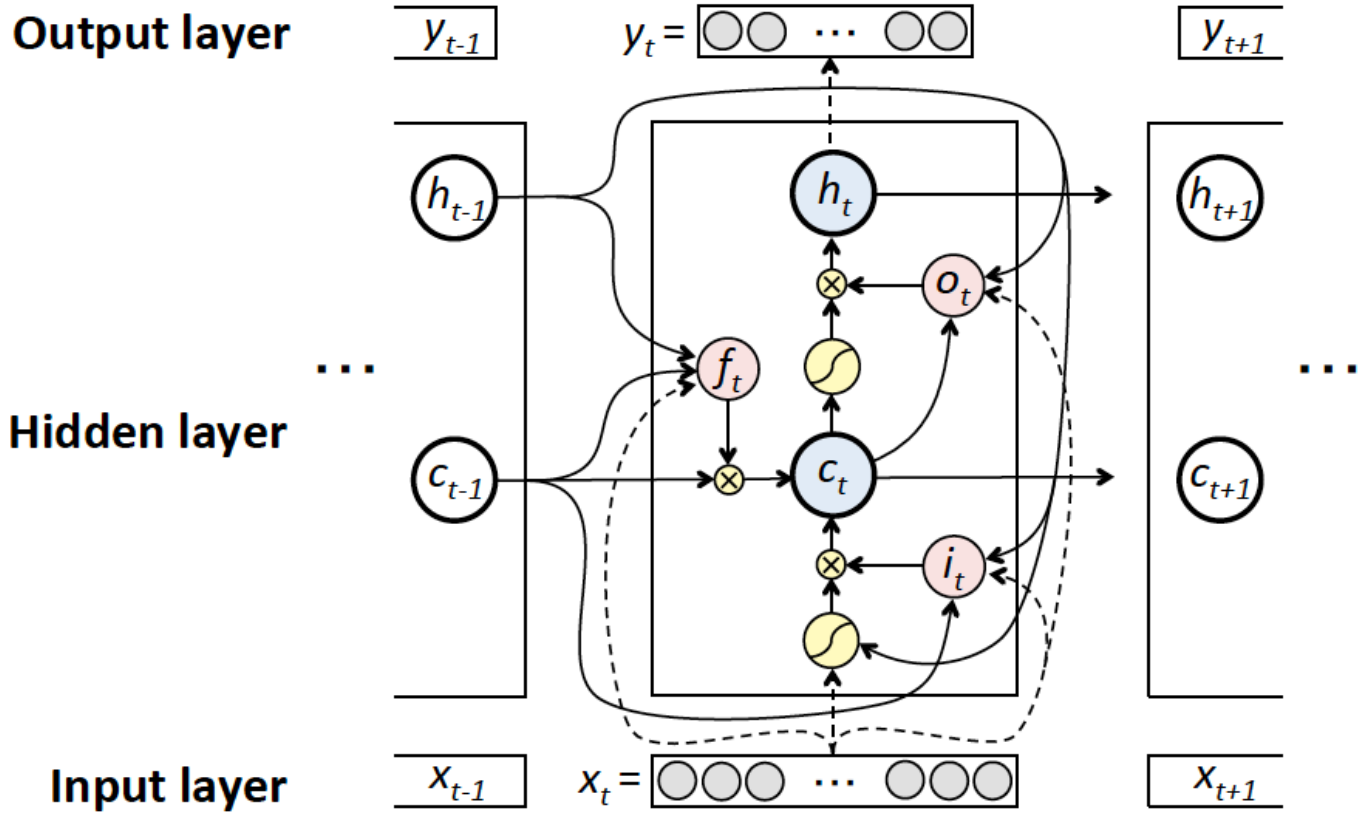
Long Short-term Memory (LSTM):

- Idea: Shouldn't need to back-prop to beginning of history. Just keep a memory of important bits.
- Introduces memory cell (c_t), with input/output/forget gates to keep track of what to remember and when



*Nice explanation of LSTM and variants here:
http://colah.github.io/posts/2015-08-Understanding-LSTMs/

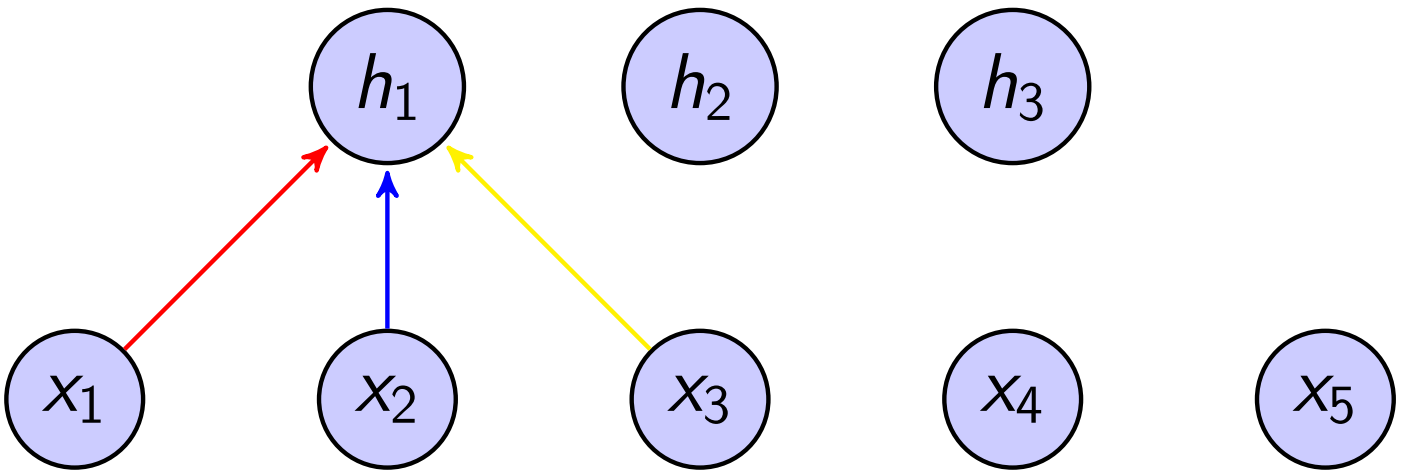
LSTM in detail [Gers et al., 2002]



$$
\begin{aligned}
i_t &= \sigma(W_{xi}x_t + W_{ci}c_{t-1} + W_{hi}h_{t-1}) \\
f_t &= \sigma(W_{xf}x_t + W_{cf}c_{t-1} + W_{hf}h_{t-1}) \\
c_t &= f_t c_{t-1} + i_t \ tanh(W_{xc}x_t + W_{hc}h_{t-1}) \\
o_t &= \sigma(W_{xo}x_t + W_{co}c_t + W_{ho}h_{t-1}) \\
h_t &= o_t \tanh(c_t), \quad y_t = softmax(W_{hy}h_t)
\end{aligned}
$$

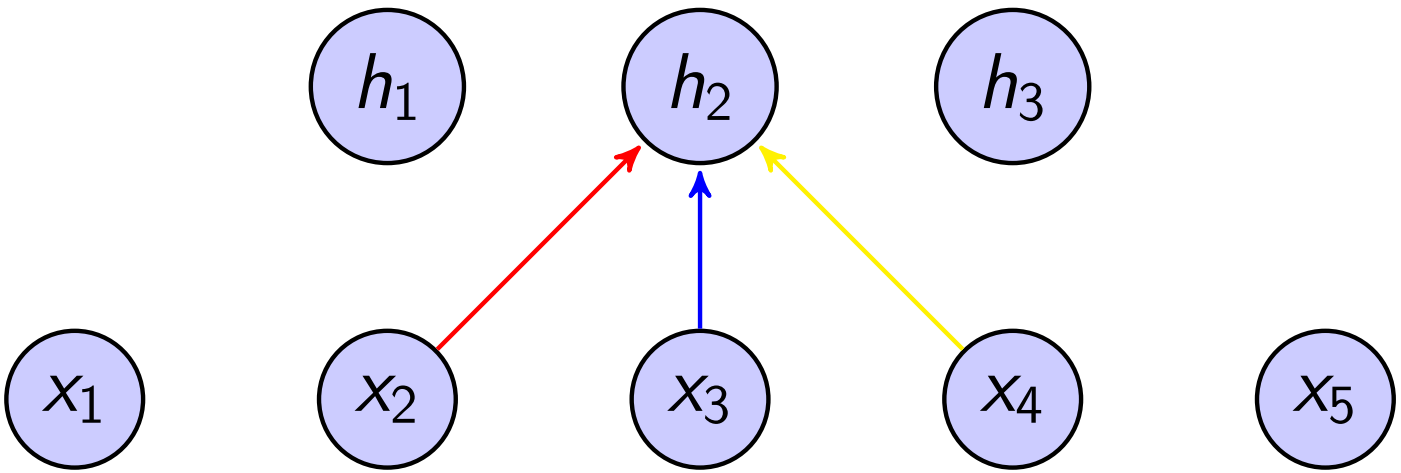
Figure courtesy of Adhi Kuncoro & Yuichiro Sawai

Convolution



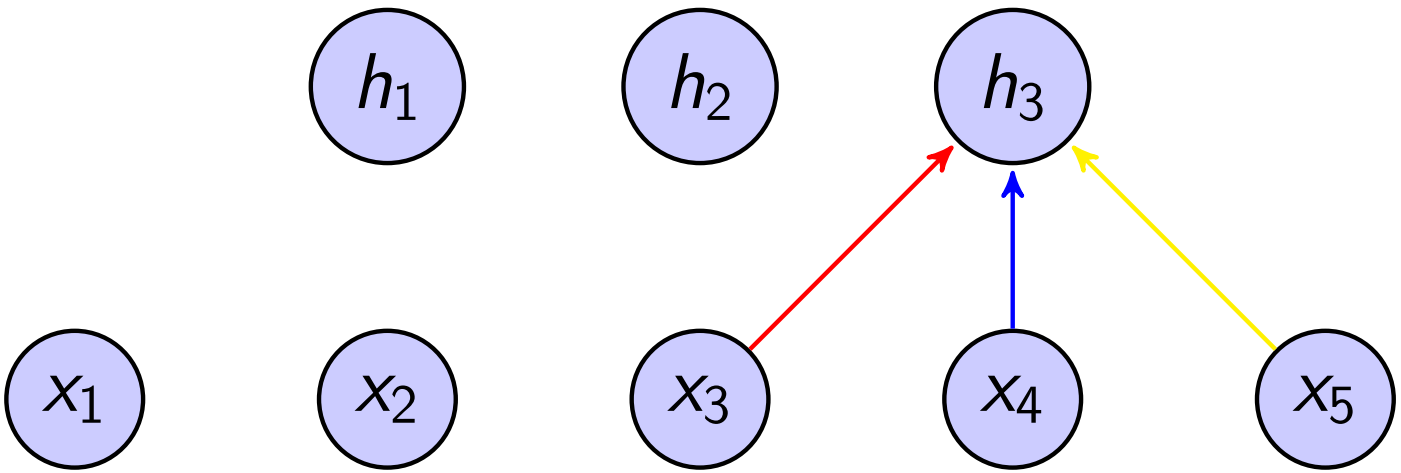
Sliding window of shared weights: $h_j = \sum_{\tau=0}^{2} w_\tau x_{j+\tau}$

Convolution



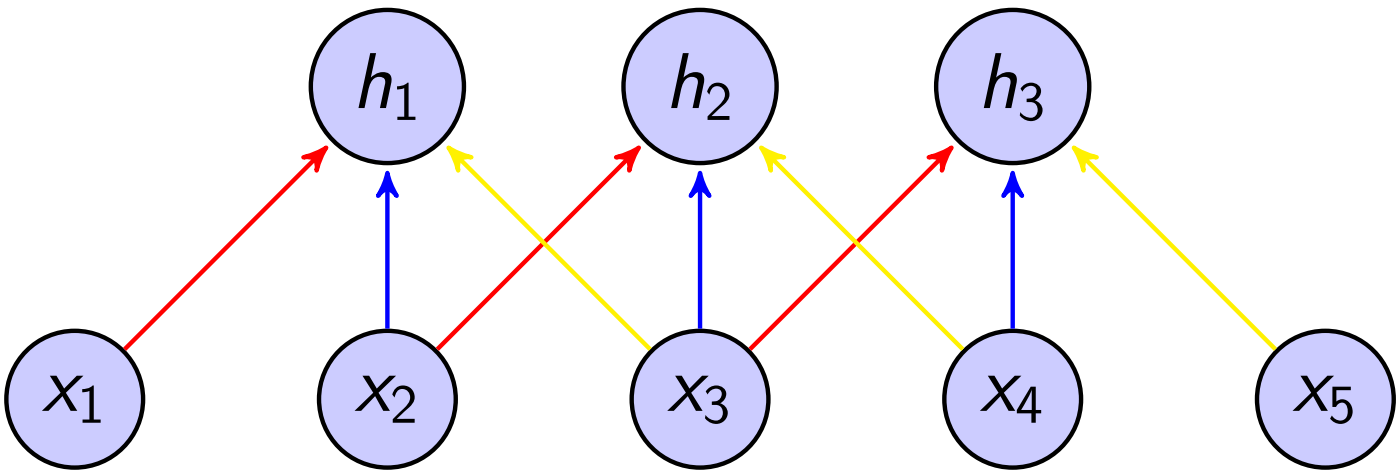
Sliding window of shared weights: $h_j = \sum_{\tau=0}^{2} w_\tau x_{j+\tau}$

Convolution



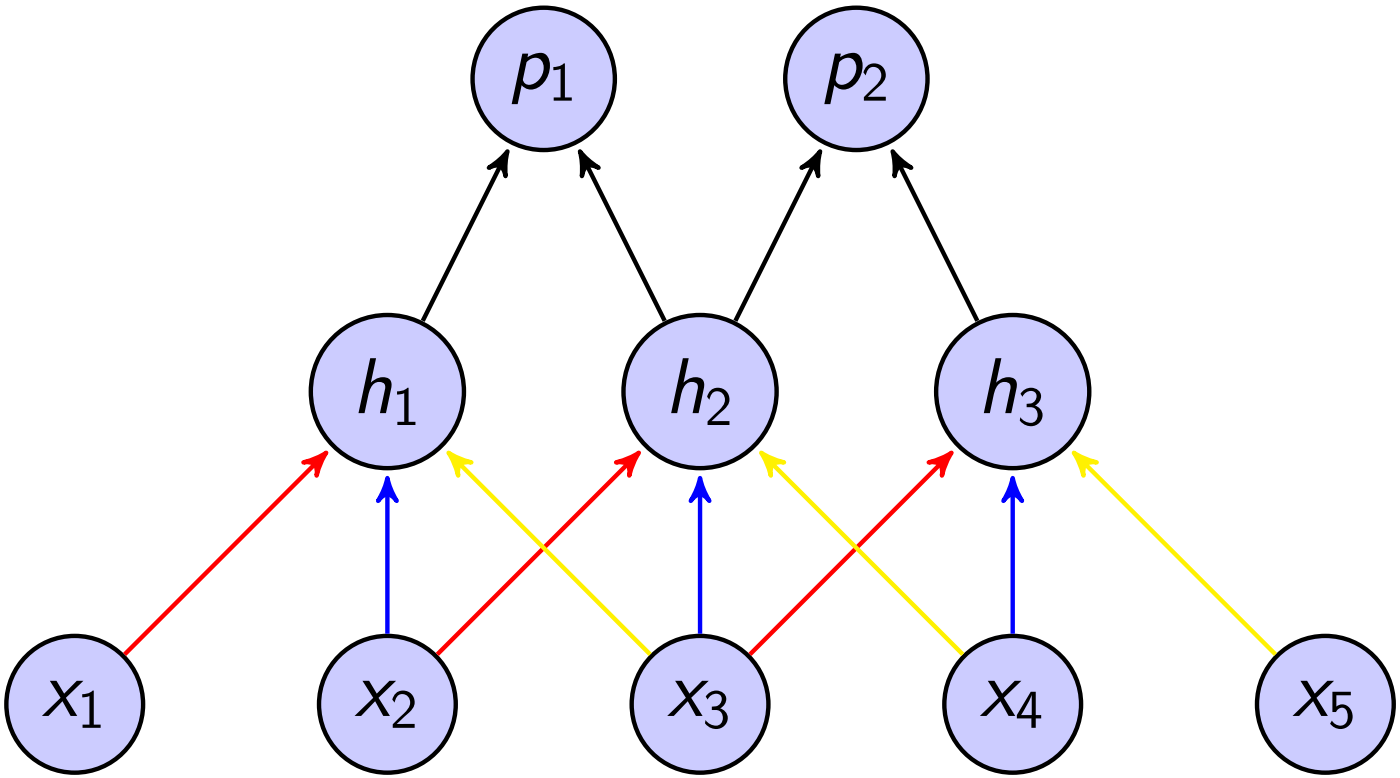
Sliding window of shared weights: $h_j = \sum_{\tau=0}^{2} w_\tau x_{j+\tau}$

Convolution



Sliding window of shared weights: $h_j = \sum_{\tau=0}^{2} w_\tau x_{j+\tau}$

Pooling



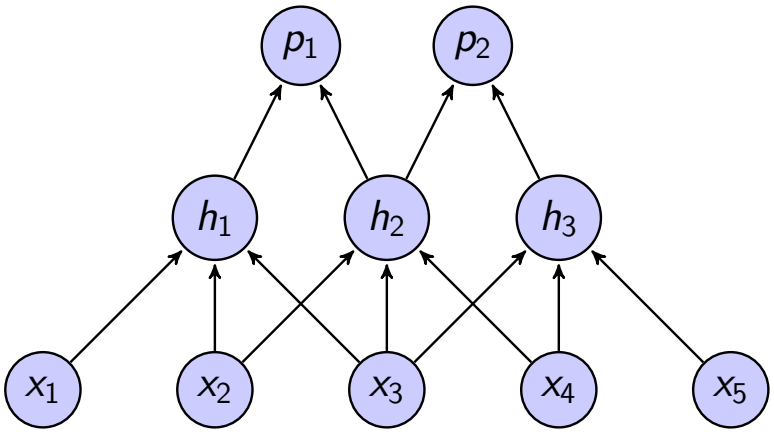
Max-Pooling

nodes: $p_1 = max(h_1, h_2)$, $p_2 = max(h_2, h_3)$

Advantages of Convolution + Pooling:

1. Fewer weights
2. Shift invariance

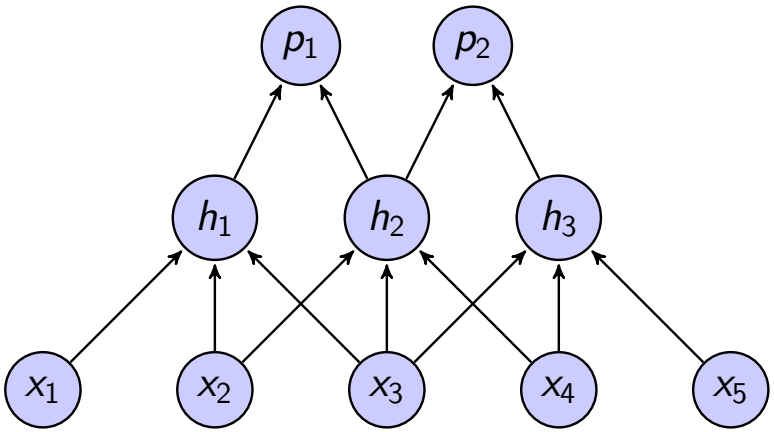
Convolutional Nets in vision [LeCun et al., 1998]



Receptive Field (RF): each h_j only connects to small input region via convolution.

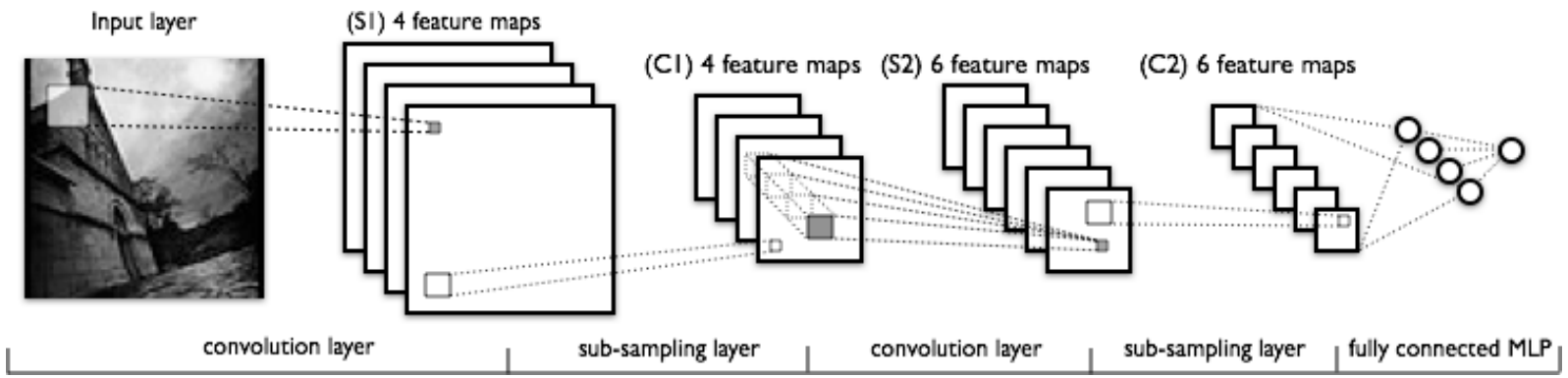
Pooling: e.g. $p_1 = max(h_1, h_2)$ or $p_1 = \sqrt{h_1^2 + h_2^2}$

Convolutional Nets in vision [LeCun et al., 1998]



Receptive Field (RF): each h_j only connects to small input region via convolution.

Pooling: e.g. $p_1 = max(h_1, h_2)$ or $p_1 = \sqrt{h_1^2 + h_2^2}$



(Figure from `http://deeplearning.net/tutorial/lenet.html`)

Summary and Thoughts

1. Word representations:
   - Distributional semantics, relational semantics, etc. $\rightarrow$ multiple views we want to encode
   - Feed-forward neural language models and word2vec

Summary and Thoughts

1. Word representations:
   - Distributional semantics, relational semantics, etc. $\rightarrow$ multiple views we want to encode
   - Feed-forward neural language models and word2vec
2. Sentence representations:
   - Big challenge in what to encode
   - Technical challenge: Handling variable-length sequences. Current solutions: convolution, recurrent nets

Summary and Thoughts

1. Word representations:
   - Distributional semantics, relational semantics, etc. $\rightarrow$ multiple views we want to encode
   - Feed-forward neural language models and word2vec
2. Sentence representations:
   - Big challenge in what to encode
   - Technical challenge: Handling variable-length sequences. Current solutions: convolution, recurrent nets
3. Questions for you:
   - Should there be multiple representations per word?
     - ★ "I went to see the *play* at the theater"; "I went to *play* in the park"; "That was a great *play* by the goal keeper."

Summary and Thoughts

1. Word representations:
   - Distributional semantics, relational semantics, etc. $\rightarrow$ multiple views we want to encode
   - Feed-forward neural language models and word2vec
2. Sentence representations:
   - Big challenge in what to encode
   - Technical challenge: Handling variable-length sequences. Current solutions: convolution, recurrent nets
3. Questions for you:
   - Should there be multiple representations per word?
     - ★ "I went to see the *play* at the theater"; "I went to *play* in the park"; "That was a great *play* by the goal keeper."
   - Do you think sentence representations should be learned bottom-up (e.g. convolution) or left-to-right (e.g. LSTM)?

Summary and Thoughts

1. Word representations:
   - Distributional semantics, relational semantics, etc. $\rightarrow$ multiple views we want to encode
   - Feed-forward neural language models and word2vec
2. Sentence representations:
   - Big challenge in what to encode
   - Technical challenge: Handling variable-length sequences. Current solutions: convolution, recurrent nets
3. Questions for you:
   - Should there be multiple representations per word?
     - ★ "I went to see the *play* at the theater"; "I went to *play* in the park"; "That was a great *play* by the goal keeper."
   - Do you think sentence representations should be learned bottom-up (e.g. convolution) or left-to-right (e.g. LSTM)?
   - What is the verdict on end-to-end learning vs. pre-trained word/sentence embeddings?

References:

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

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

Bollacker, K., Evans, C., Paritosh, P., Sturge, T., and Taylor, J. (2008).
Freebase: A collaboratively created graph database for structuring human knowledge.
In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, pages 1247–1250, New York, NY, USA. ACM.

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*.

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.

Faruqui, M., Dodge, J., Jauhar, S. K., Dyer, C., Hovy, E. H., and Smith, N. A. (2014).
Retrofitting word vectors to semantic lexicons.
CoRR, abs/1411.4166.

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.

Fried, D. and Duh, K. (2014).
Incorporating both distributional and relational semantics in word representations.
CoRR, abs/1412.5836.

Gers, F. A., Schraudolph, N. N., and Schmidhuber, J. (2002).
Learning precise timing with LSTM recurrent networks.
JMLR.

Gouws, S., Bengio, Y., and Corrado, G. (2015).
Bilbowa: Fast bilingual distributed representations without word alignments.
In *Proceedings of the 32nd International Conference on Machine Learning*.

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.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998).
Gradient-based learning applied to document recognition.
Proc, 86(11):2278–2324.

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

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*.

Miller, G. A. (1995).
WordNet: A lexical database for English.
Communications of the ACM, 38(11):39–41.

Rastogi, P., Van Durme, B., and Arora, R. (2015).
Multiview lsa: Representation learning via generalized cca.
In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 556–566, Denver, Colorado. Association for Computational Linguistics.

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