Language Representation and Recurrent Nets

Kapil Thadani
kapil@cs.columbia.edu

YAHOO!
RESEARCH

Outline

- Language representation
  - Bag of words
  - Distributional hypothesis
  - Word embeddings: word2vec, GloVe
  - Beyond words: paragraph vector

- Recurrent neural networks
  - Backpropagation through time
  - Long short-term memory (LSTM)
  - Gated recurrent units (GRU)

- NLP scenarios
  - Classification
  - Tagging
  - Generation
  - Text-to-text

Formal language

(i) Set of sequences over symbols from an alphabet

sentences	words	vocabulary
utterances	morphemes	
documents	MWEs	
⋮	⋮	

(ii) Rules for valid sequences

spelling	orthography, morphology
grammar	syntax
meaning	semantics, discourse, pragmatics, ⋯

Natural language

(i) Set of sequences over symbols from an alphabet

sentences	words	vocabulary
utterances	morphemes	
documents	MWEs	
⋮	⋮	

(ii) Rules for valid sequences

spelling	orthography, morphology
grammar	syntax
meaning	semantics, discourse, pragmatics, $\cdots$

Sparse representations

Word: one-hot (1-of-V) vectors

Document: "bag of words"

Emphasize rare words with inverse document frequency (IDF)

$$|V| \quad \begin{array}{cl} \boxed{\begin{matrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{matrix}} & \begin{matrix} \text{a} \\ \\ \text{dog} \\ \\ \text{zygote} \end{matrix} \end{array}$$

Compare documents with cosine similarity

+ Simple and interpretable
− No notion of word order
− No implicit semantics
− Curse of dimensionality with large $|V|$

Sparse representations

Consider a neural network to read documents with:

- at most T words
- drawn from vocabulary V
- into a hidden layer with H units

How many parameters in the input layer for:

- Tweets?

- News stories?

Sparse representations

Consider a neural network to read documents with:

- at most T words
- drawn from vocabulary V
- into a hidden layer with H units

How many parameters in the input layer for:

- Tweets?

  $T = 50,\ |V| = 100\text{K},\ H = 100\ \rightarrow 0.5\text{B}$

- News stories?

Sparse representations

Consider a neural network to read documents with:

- at most T words
- drawn from vocabulary V
- into a hidden layer with H units

How many parameters in the input layer for:

- Tweets?

  $T = 50,\ |V| = 100\mathrm{K},\ H = 100\ \rightarrow 0.5\mathrm{B}$

- News stories?

Sparse representations

Consider a neural network to read documents with:

- at most T words
- drawn from vocabulary V
- into a hidden layer with H units

How many parameters in the input layer for:

- Tweets?

  $T = 50,\ |V| = 100\text{K},\ H = 100\ \rightarrow 0.5\text{B}$

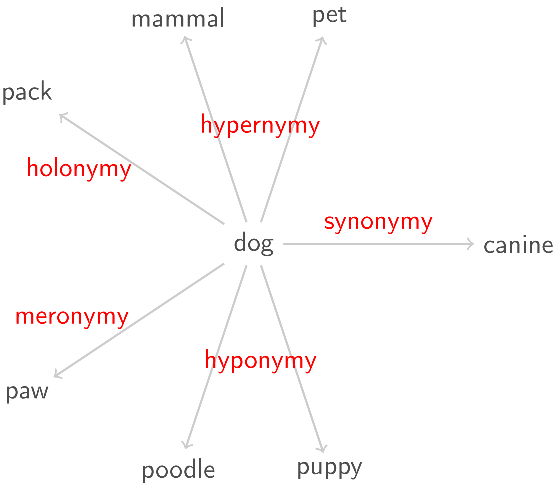
- News stories?

  $T = 2000,\ |V| = 200\text{K},\ H = 100 \rightarrow 40\text{B}$

Lexical semantics

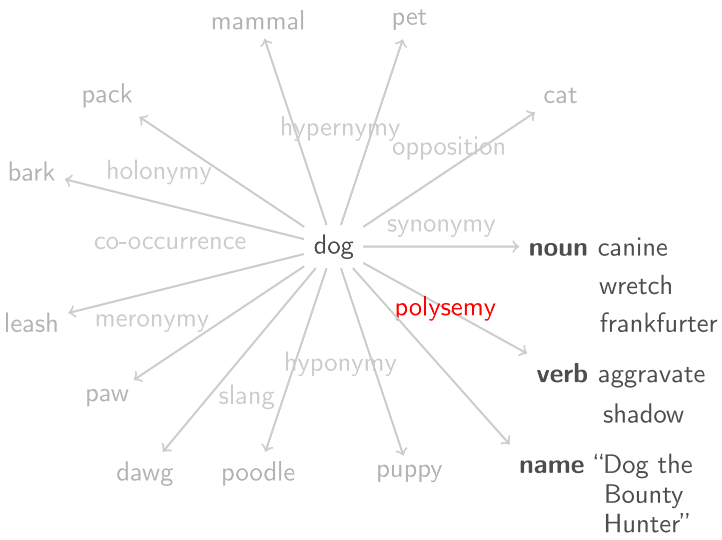
dog

Lexical semantics



Lexical semantics

Lexical semantics



Distributional approaches

Words that occur in similar contexts have similar meanings

e.g., record word co-occurrence within a context window over a large corpus

$$
|V| \quad \begin{array}{c} \text{dog} \\ \left[\begin{array}{c} \vdots \\ 0.3 \\ \vdots \\ 0.1 \\ \vdots \\ 0.2 \\ \vdots \end{array}\right] \end{array} \begin{array}{l} \\ \\ \text{bark} \\ \\ \text{leash} \\ \\ \text{wag} \\ \\ \end{array}
$$

Weight association with pointwise mutual information (PMI), etc

$$PMI(w_1, w_2) = \log_2 \frac{p(w_1, w_2)}{p(w_1)p(w_2)}$$

+ Implicit semantics, i.e., related words have similar representations
– Domain dependence on training corpus
– Curse of dimensionality with large $|V|$

Latent Semantic Analysis

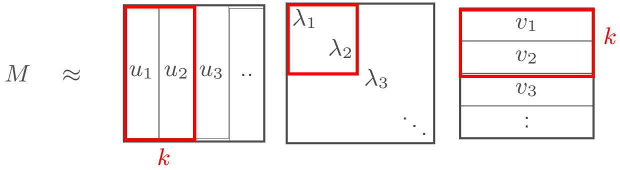
Deerwester et al. (1990)

Indexing by Latent Semantic Analysis

Construct term-document matrix

$$M = \underbrace{\begin{bmatrix} w_1^{(1)} & w_1^{(2)} & \cdots \\ w_2^{(1)} & \ddots & \\ \vdots & & \end{bmatrix}}_{|D|} \Bigg\} |V|$$

Singular value decomposition



Select top k singular vectors for k-dim embeddings of words/docs

word2vec

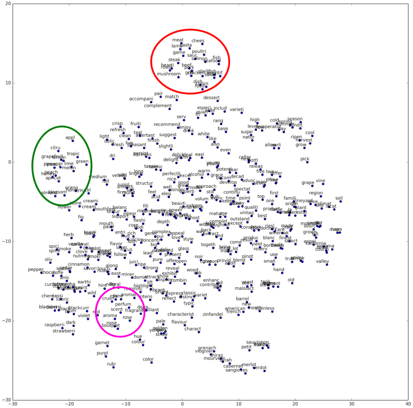
Mikolov et al. (2013)

Efficient Estimation of Word Representations in Vector Space

word2vec

Mikolov et al. (2013)

Efficient Estimation of Word Representations in Vector Space



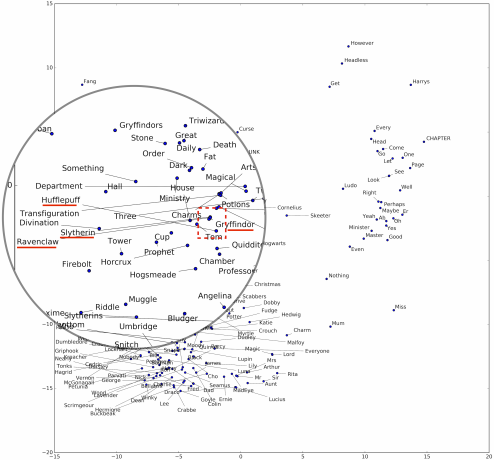
t-SNE projection of word embeddings from 58k Winemaker's Notes

http://methodmatters.blogspot.com/2017/11/using-word2vec-to-analyze-word.html

word2vec

Mikolov et al. (2013)

Efficient Estimation of Word Representations in Vector Space



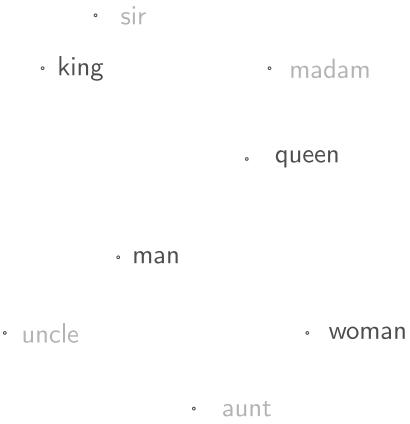
t-SNE projection of name embeddings from all 7 Harry Potter books

`https://github.com/nchah/word2vec4everything`

word2vec

Mikolov et al. (2013)

Efficient Estimation of Word Representations in Vector Space

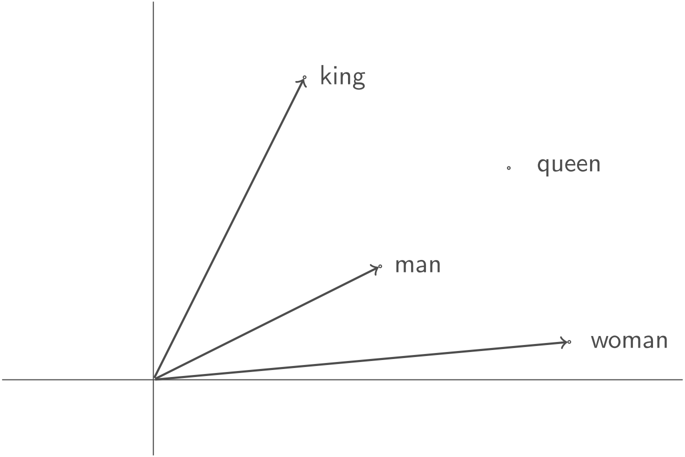


Linear substructure for related words

word2vec

Mikolov et al. (2013)

Efficient Estimation of Word Representations in Vector Space

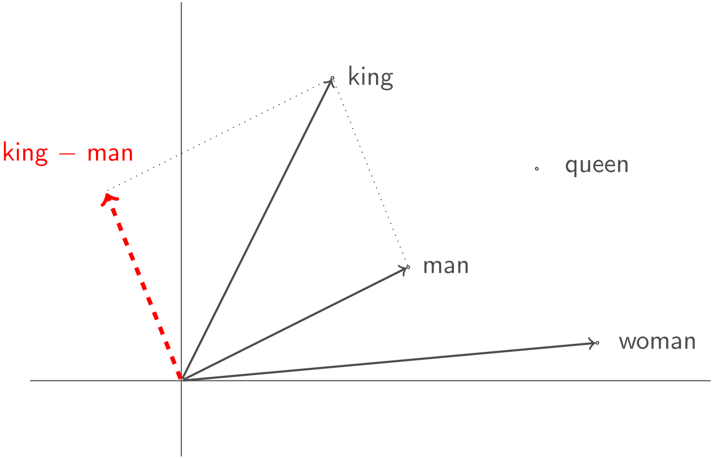


Linear substructure for related words

word2vec

Mikolov et al. (2013)

Efficient Estimation of Word Representations in Vector Space

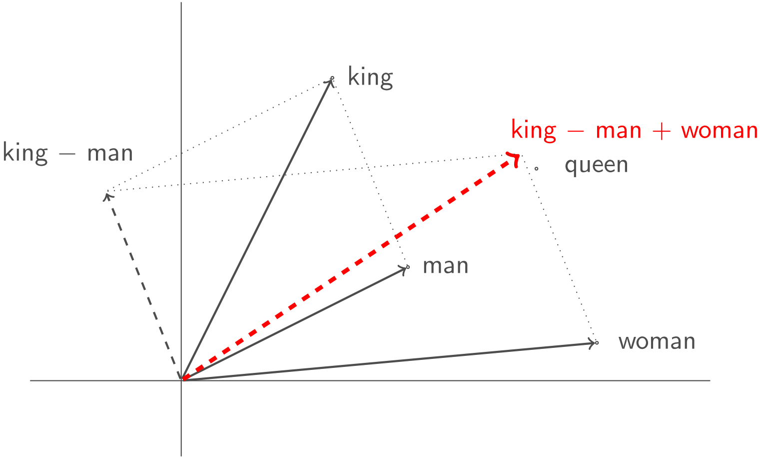


Linear substructure for related words

word2vec

Mikolov et al. (2013)

Efficient Estimation of Word Representations in Vector Space



Linear substructure for related words

word2vec

Mikolov et al. (2013)

Efficient Estimation of Word Representations in Vector Space

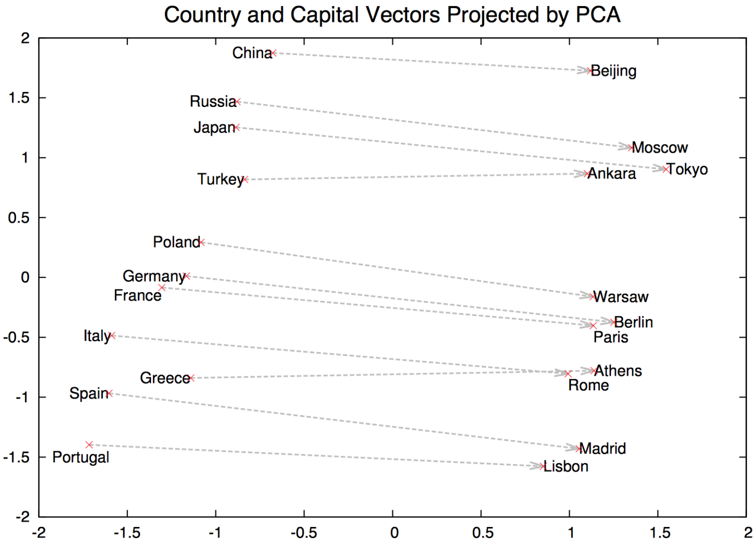
Relationship	Example 1	Example 2	Example 3
France - Paris	Italy: Rome	Japan: Tokyo	Florida: Tallahassee
big - bigger	small: larger	cold: colder	quick: quicker
Miami - Florida	Baltimore: Maryland	Dallas: Texas	Kona: Hawaii
Einstein - scientist	Messi: midfielder	Mozart: violinist	Picasso: painter
Sarkozy - France	Berlusconi: Italy	Merkel: Germany	Koizumi: Japan
copper - Cu	zinc: Zn	gold: Au	uranium: plutonium
Berlusconi - Silvio	Sarkozy: Nicolas	Putin: Medvedev	Obama: Barack
Microsoft - Windows	Google: Android	IBM: Linux	Apple: iPhone
Microsoft - Ballmer	Google: Yahoo	IBM: McNealy	Apple: Jobs
Japan - sushi	Germany: bratwurst	France: tapas	USA: pizza

Analogical reasoning

word2vec

Mikolov et al. (2013)

Distributed Representations of Words and Phrases and their Compositionality



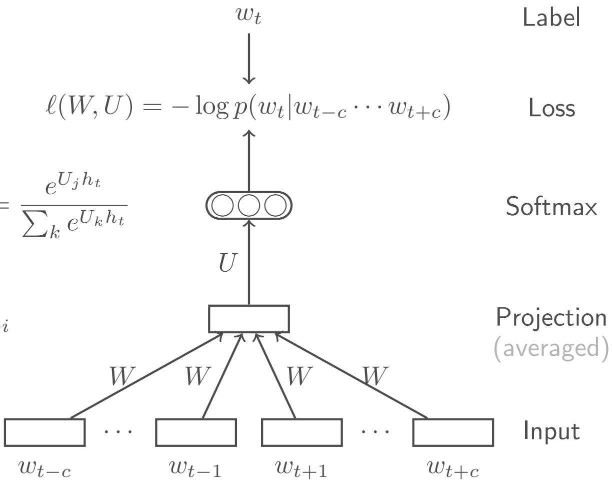
Visualizing lexical relationships

word2vec

Mikolov et al. (2013)

Continuous Bag-of-Words (CBOW)

- Predict target w_t given context $w_{t-c}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+c}$



$$p(w_j | w_{t-c} \cdots w_{t+c}) = \frac{e^{U_j h_t}}{\sum_k e^{U_k h_t}}$$

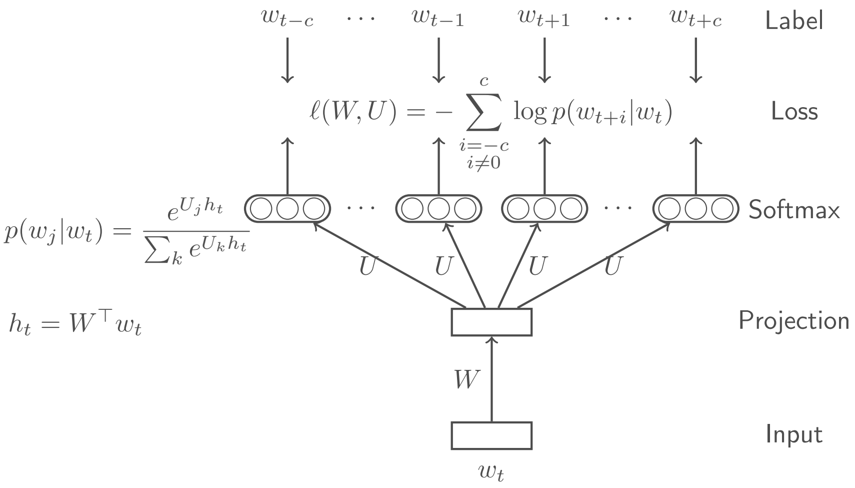
$$h_t = \frac{1}{2c} \sum_{\substack{i=-c \\ i \neq 0}}^{c} W^\top w_{t+i}$$

word2vec

Mikolov et al. (2013)

Skip-gram

- Predict context $w_{t-c}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+c}$ given target w_t



word2vec

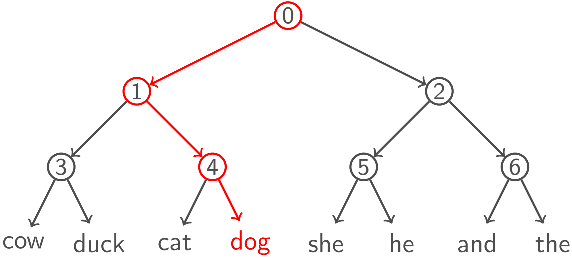
Mikolov et al. (2013)

Cost of computing $\nabla p(w_j | \cdots)$ is proportional to V!

Alternative 1: Hierarchical softmax

- Predict path in binary tree representation of output layer
- Reduces to $\log_2(V)$ binary decisions

$$p(w_t = \text{“dog”} | \cdots) = (1 - \sigma(U_0 h_t)) \times \sigma(U_1 h_t) \times \sigma(U_4 h_t)$$



word2vec

Mikolov et al. (2013)

Cost of computing $\nabla p(w_j | \cdots)$ is proportional to V!

Alternative 2: Negative sampling

- Change objective to differentiate target vector from noisy samples with logistic regression

$$\max \ \log \sigma(u_j^\top h_t) + \sum_{k=1}^{K} \mathbb{E}_{w_m \sim \Psi} \log \sigma(-{u_m}^\top h_t)$$

where $u_j = U_j = j$'th column of U

and $w_j \in \text{context}(w_t)$

- Noise distribution Ψ typically unigram, uniform or in between
- Number of samples K typically 5–20

word2vec

Levy & Goldberg (2014)

Neural Word Embedding as Implicit Matrix Factorization

Skip-gram with negative sampling increases $u_j^\top h_t$ for real word-context pairs $\langle w_t, w_j \rangle$ and decreases it for noise pairs

Given:

- a matrix of d-dim word vectors W ($|V_w| \times d$)
- a matrix of d-dim context vectors U ($|V_u| \times d$)

Skip-gram is implicitly factorizing the matrix $M = WU^\top$

What is M?

- Word-context matrix where each cell (i, j) contains $PMI(w_i, w_j)$
- If number of negative samples $K > 1$, this is shifted by a constant $-\log K$
- (Assuming large enough d and iterations)

GloVe

Pennington et al. (2014)

Global Vectors for Word Representation

Similar words have similar ratios of co-occurrence probabilities for context words

$$f\left((u_i - u_j)^\top \tilde{u}_k\right) = \frac{p(w_k|w_i)}{p(w_k|w_j)}$$

$$\Rightarrow u_i^\top \tilde{u}_k + b_i + \tilde{b}_k = \log \frac{count(w_i, w_k)}{count(w_i)}$$

+ Explicitly encodes linear substructure between similar words
+ Scales to huge corpora

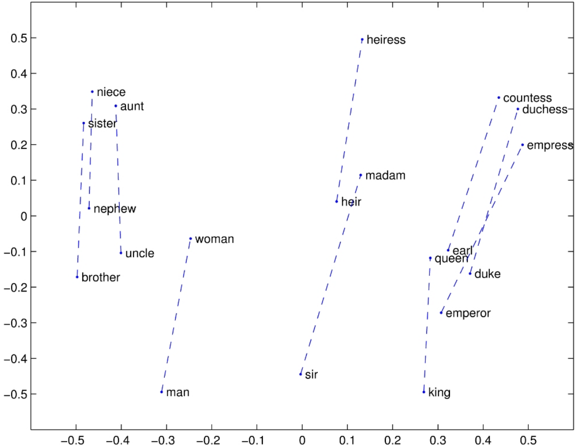
Pre-trained embeddings at `https://nlp.stanford.edu/projects/glove/`

- Common Crawl: 840B tokens, $|V| = 2.2\text{M}$, $d = 300$
- Twitter: 27B tokens, $|V| = 1.2\text{M}$, $d = 25 - 200$

GloVe

Pennington et al. (2014)

Global Vectors for Word Representation

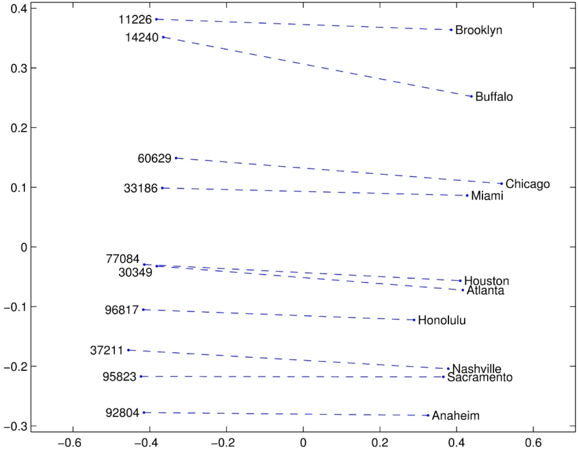


Linear substructure (male vs female)

GloVe

Pennington et al. (2014)

Global Vectors for Word Representation

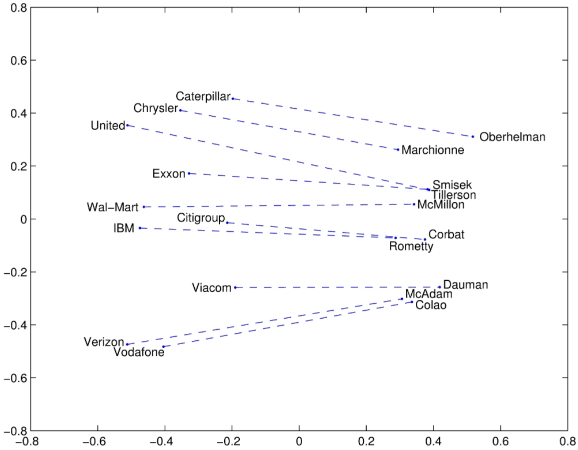


Linear substructure (city vs zip code)

GloVe

Pennington et al. (2014)

Global Vectors for Word Representation

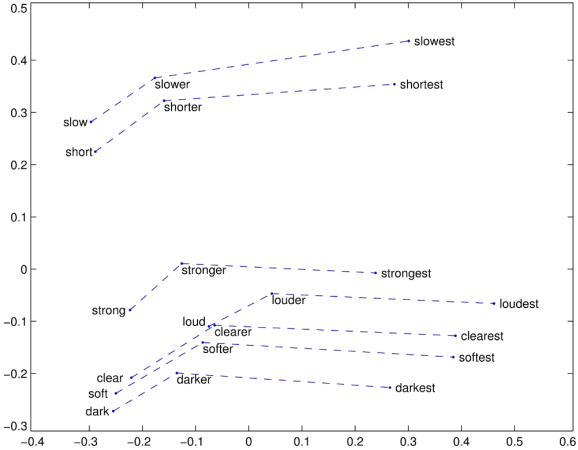


Linear substructure (company vs CEO)

GloVe

Pennington et al. (2014)

Global Vectors for Word Representation



Linear substructure (comparative vs superlative adjectives)

word2vec with phrases

Mikolov et al. (2013)

Distributed Representations of Words and Phrases and their Compositionality

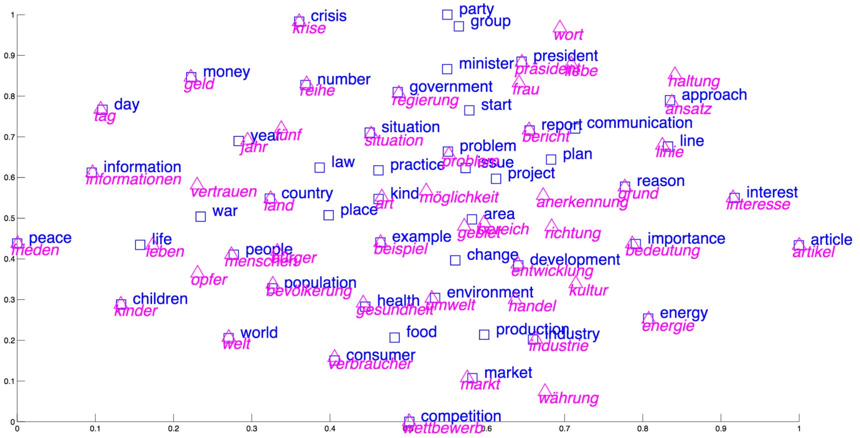
Newspapers			
New York	New York Times	Baltimore	Baltimore Sun
San Jose	San Jose Mercury News	Cincinnati	Cincinnati Enquirer
NHL Teams			
Boston	Boston Bruins	Montreal	Montreal Canadiens
Phoenix	Phoenix Coyotes	Nashville	Nashville Predators
NBA Teams			
Detroit	Detroit Pistons	Toronto	Toronto Raptors
Oakland	Golden State Warriors	Memphis	Memphis Grizzlies
Airlines			
Austria	Austrian Airlines	Spain	Spainair
Belgium	Brussels Airlines	Greece	Aegean Airlines
Company executives			
Steve Ballmer	Microsoft	Larry Page	Google
Samuel J. Palmisano	IBM	Werner Vogels	Amazon

Phrase analogies

Czech + currency	Vietnam + capital	German + airlines	Russian + river	French + actress
koruna	Hanoi	airline Lufthansa	Moscow	Juliette Binoche
Check crown	Ho Chi Minh City	carrier Lufthansa	Volga River	Vanessa Paradis
Polish zolty	Viet Nam	flag carrier Lufthansa	upriver	Charlotte Gainsbourg
CTK	Vietnamese	Lufthansa	Russia	Cecile De

Additive compositionality

Bilingual word embeddings



Aligned embeddings for English and German (Luong et al., 2015)

Resources for word embeddings

Original code and pre-trained embeddings:
https://code.google.com/archive/p/word2vec/

Python library for word and document embeddings:
https://radimrehurek.com/gensim/models/word2vec.html

Tensorflow tutorial and implementation:
https://www.tensorflow.org/tutorials/representation/word2vec

FastText library and pre-trained embeddings for 157 languages:
https://github.com/facebookresearch/fastText

Beyond words

Can we add word vectors to make sentence/paragraph/doc vectors?

doc $A = a_1 + a_2 + a_3$

doc $B = b_1 + b_2 + b_3$

$$\begin{aligned}
cos(A, B) &= \frac{A \cdot B}{||A|| \cdot ||B||} \\
&= \frac{1}{||A|| \cdot ||B||}(a_1 \cdot b_1 + a_1 \cdot b_2 + a_1 \cdot b_3 + \\
&\qquad a_2 \cdot b_1 + a_2 \cdot b_2 + a_2 \cdot b_3 + \\
&\qquad a_3 \cdot b_1 + a_3 \cdot b_2 + a_3 \cdot b_3) \\
&= \text{weighted all-pairs similarity over } A \text{ and } B
\end{aligned}$$

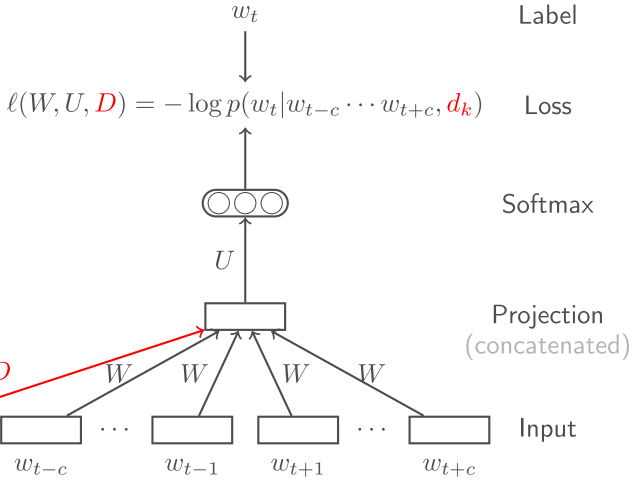
Paragraph vector (a.k.a doc2vec)

Le & Mikolov (2014)

Distributed memory

Distributed Representations of Sentences and Documents

- Predict target w_t given context $w_{t-c}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+c}$ and doc label d_k
- At test time, hold U, W fixed and back-prop into expanded D



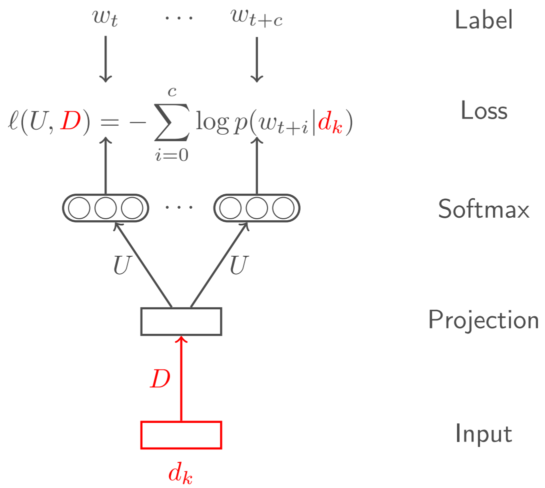
Paragraph vector (a.k.a doc2vec)

Le & Mikolov (2014)

Distributed Representations of Sentences and Documents

Distributed Bag-of-Words (DBOW)

- Predict target n-grams $w_t, \dots, w_{t+c}$ given doc label d_k
- At test time, hold U fixed and back-prop into expanded D



Semantics are elusive

Visitors saw her duck with binoculars .

Did she duck or does she *have* a duck?

Who has the binoculars?

How many pairs of binoculars are there?

Semantics are elusive

Visitors saw her duck with binoculars .

Did she duck or does she *have* a duck?

Who has the binoculars?

How many pairs of binoculars are there?

Semantics are elusive

Visitors	saw	her	duck	with	binoculars	.
NNS	VBD	PRP$	NN	IN	NNS	.

Did she duck or does she *have* a duck?

Who has the binoculars?

How many pairs of binoculars are there?

Semantics are elusive

Visitors	saw	her	duck	with	binoculars	.
NNS	VBD	PRP	VB	IN	NNS	.

Did she duck or does she *have* a duck?

Who has the binoculars?

How many pairs of binoculars are there?

Semantics are elusive

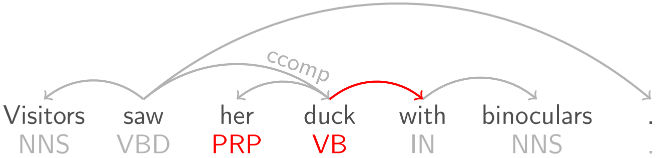
Visitors	saw	her	duck	with	binoculars	.
NNS	VBD	PRP	VB	IN	NNS	.

Did she duck or does she *have* a duck?

Who has the binoculars?

How many pairs of binoculars are there?

Semantics are elusive

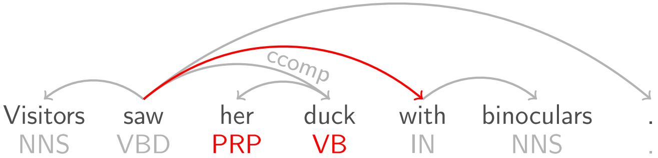


Did she duck or does she *have* a duck?

Who has the binoculars?

How many pairs of binoculars are there?

Semantics are elusive

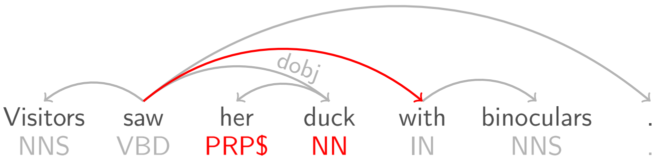


Did she duck or does she *have* a duck?

Who has the binoculars?

How many pairs of binoculars are there?

Semantics are elusive

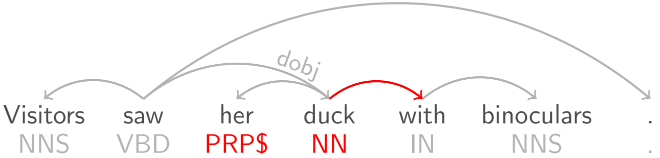


Did she duck or does she *have* a duck?

Who has the binoculars?

How many pairs of binoculars are there?

Semantics are elusive

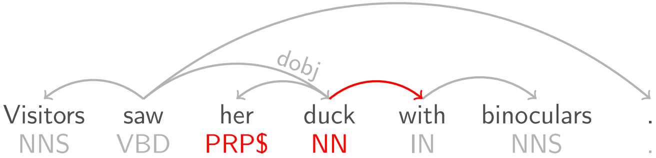


Did she duck or does she *have* a duck?

Who has the binoculars?

How many pairs of binoculars are there?

Semantics are elusive

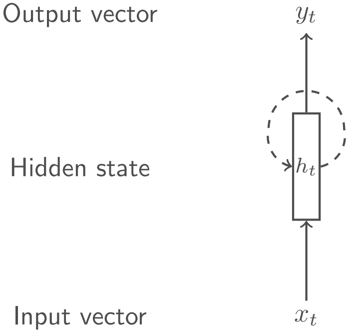


Did she duck or does she *have* a duck?

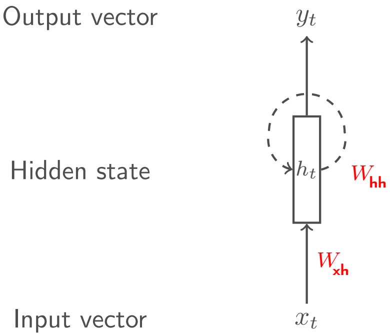
Who has the binoculars?

How many pairs of binoculars are there?

Recurrent connections



Recurrent connections



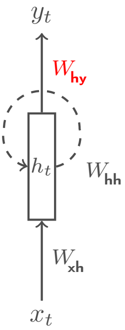
$$h_t = \phi_h(W_{xh}\, x_t + W_{hh}\, h_{t-1})$$

Recurrent connections

Output vector

Hidden state

Input vector



$$y_t = \phi_{\mathbf{y}}(W_{\mathbf{hy}}\, h_t)$$

$$h_t = \phi_{\mathbf{h}}(W_{\mathbf{xh}}\, x_t + W_{\mathbf{hh}}\, h_{t-1})$$

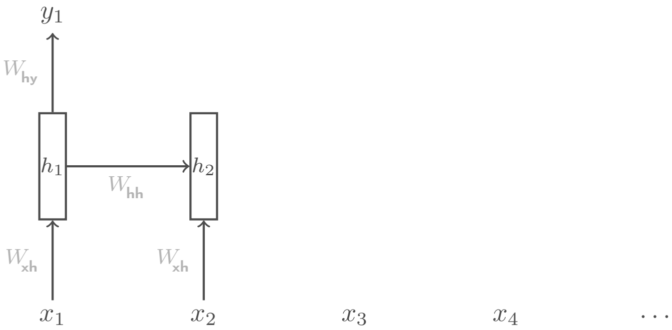
Recurrent connections: Unfolding

x_1 x_2 x_3 x_4 $\cdots$

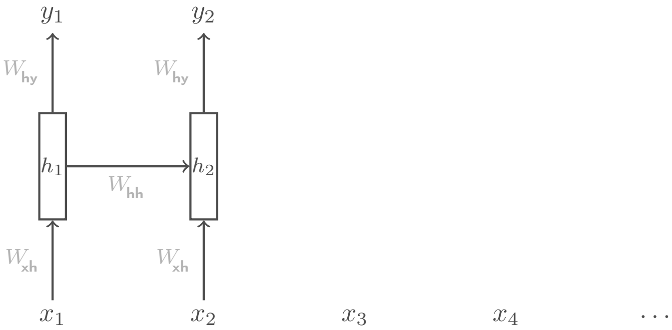
Recurrent connections: Unfolding

Recurrent connections: Unfolding

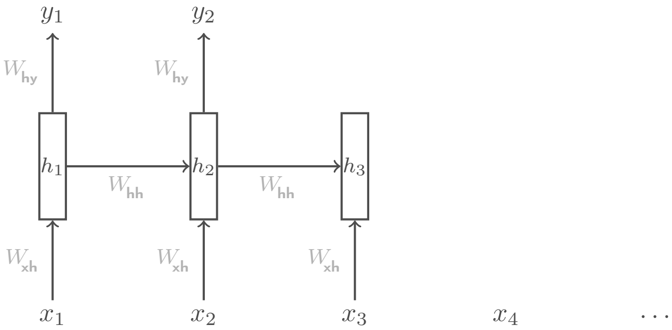
Recurrent connections: Unfolding



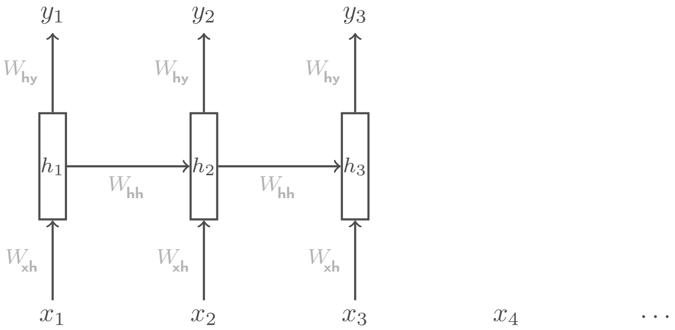
Recurrent connections: Unfolding



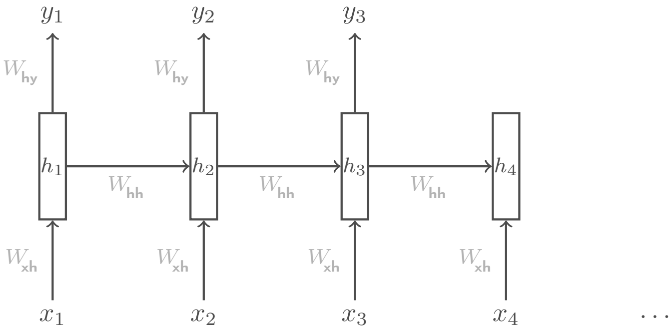
Recurrent connections: Unfolding



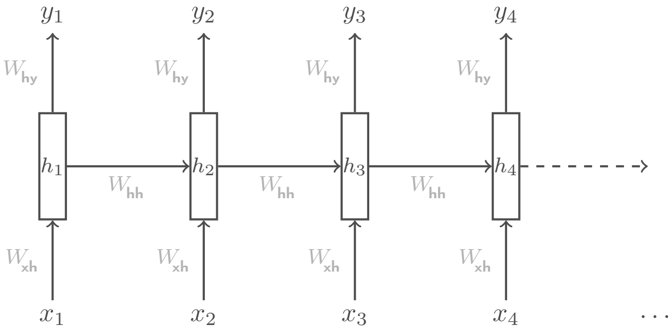
Recurrent connections: Unfolding



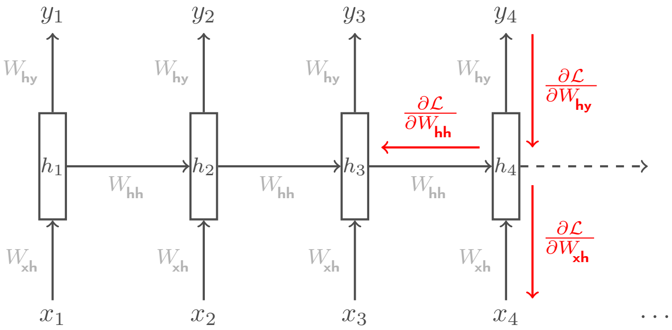
Recurrent connections: Unfolding



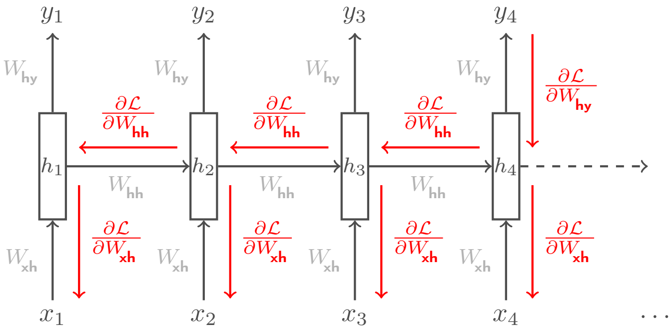
Recurrent connections: Unfolding



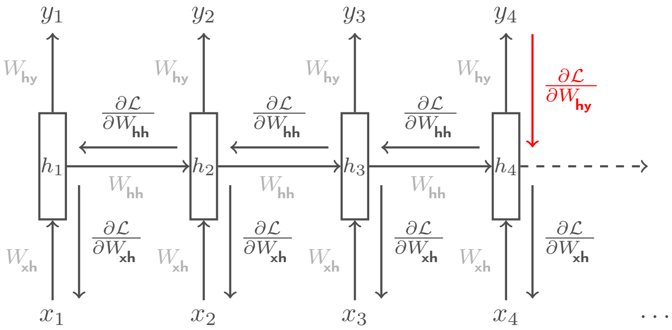
Recurrent connections: Backprop through time



Recurrent connections: Backprop through time

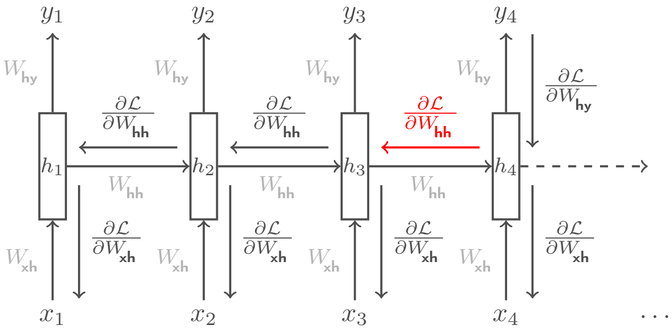


Recurrent connections: Backprop through time



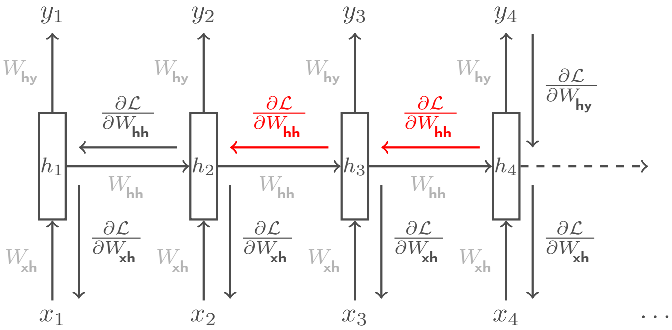
$$\frac{\partial \mathcal{L}}{\partial W_{\mathsf{hy}}} = \frac{\partial \mathcal{L}}{\partial y_4} \frac{\partial y_4}{\partial W_{\mathsf{hy}}}$$

Recurrent connections: Backprop through time



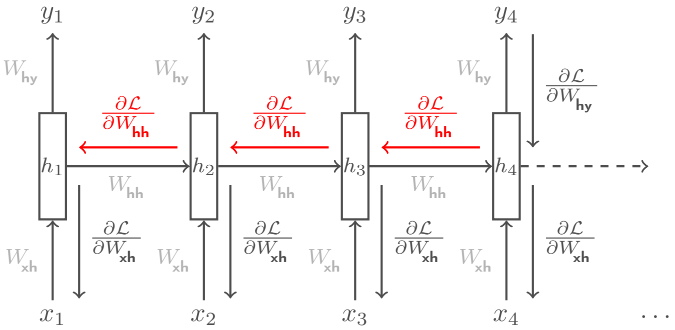
$$\frac{\partial \mathcal{L}}{\partial W_{\text{hh}}} = \frac{\partial \mathcal{L}}{\partial y_4} \frac{\partial y_4}{\partial h_4} \frac{\partial h_4}{\partial W_{\text{hh}}}$$

Recurrent connections: Backprop through time



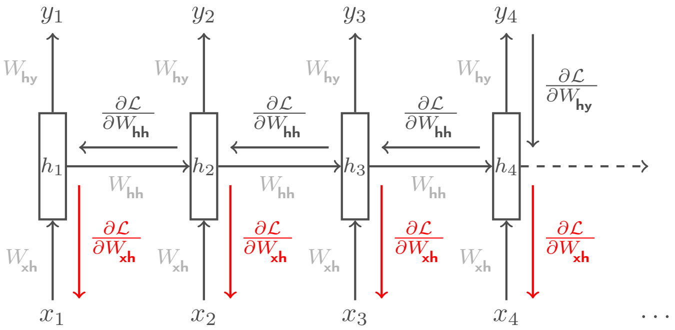
$$\frac{\partial \mathcal{L}}{\partial W_{\mathsf{hh}}} = \frac{\partial \mathcal{L}}{\partial y_4}\frac{\partial y_4}{\partial h_4}\frac{\partial h_4}{\partial W_{\mathsf{hh}}} + \frac{\partial \mathcal{L}}{\partial y_4}\frac{\partial y_4}{\partial h_4}\frac{\partial h_4}{\partial h_3}\frac{\partial h_3}{\partial W_{\mathsf{hh}}}$$

Recurrent connections: Backprop through time



$$\frac{\partial \mathcal{L}}{\partial W_{\mathbf{hh}}} = \frac{\partial \mathcal{L}}{\partial y_4}\frac{\partial y_4}{\partial h_4}\frac{\partial h_4}{\partial W_{\mathbf{hh}}} + \frac{\partial \mathcal{L}}{\partial y_4}\frac{\partial y_4}{\partial h_4}\frac{\partial h_4}{\partial h_3}\frac{\partial h_3}{\partial W_{\mathbf{hh}}} + \frac{\partial \mathcal{L}}{\partial y_4}\frac{\partial y_4}{\partial h_4}\frac{\partial h_4}{\partial h_3}\frac{\partial h_3}{\partial h_2}\frac{\partial h_2}{\partial W_{\mathbf{hh}}}$$

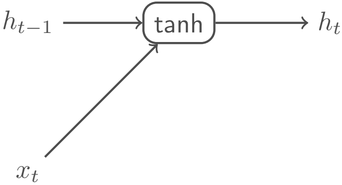
Recurrent connections: Backprop through time



$$\frac{\partial \mathcal{L}}{\partial W_{xh}} = \frac{\partial \mathcal{L}}{\partial y_4}\frac{\partial y_4}{\partial h_4}\frac{\partial h_4}{\partial W_{xh}} + \frac{\partial \mathcal{L}}{\partial y_4}\frac{\partial y_4}{\partial h_4}\frac{\partial h_4}{\partial h_3}\frac{\partial h_3}{\partial W_{xh}} + \frac{\partial \mathcal{L}}{\partial y_4}\frac{\partial y_4}{\partial h_4}\frac{\partial h_4}{\partial h_3}\frac{\partial h_3}{\partial h_2}\frac{\partial h_2}{\partial W_{xh}} + \frac{\partial \mathcal{L}}{\partial y_4}\frac{\partial y_4}{\partial h_4}\frac{\partial h_4}{\partial h_3}\frac{\partial h_3}{\partial h_2}\frac{\partial h_2}{\partial h_1}\frac{\partial h_1}{\partial W_{xh}}$$

Activation functions

$\phi_{\mathbf{h}}$ is typically a smooth, bounded function, e.g., σ, tanh



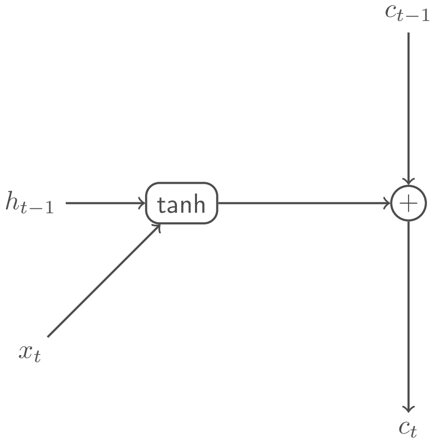
$$h_t = \mathsf{tanh}(W_{\mathbf{xh}}\, x_t + W_{\mathbf{hh}}\, h_{t-1})$$

- Susceptible to vanishing gradients
- Can fail to capture long-term dependencies

Long short-term memory (LSTM)

Gers et al (1999)

Learning to Forget: Continual Prediction with LSTM

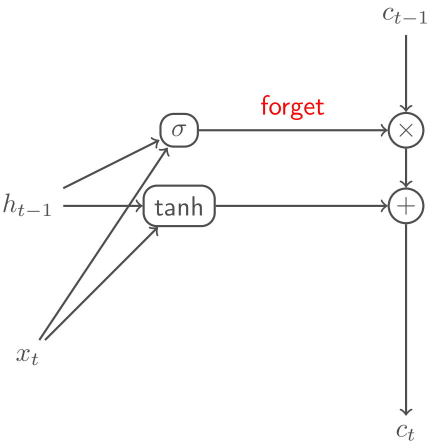


$$\tilde{c}_t = \mathsf{tanh}(W_{\mathsf{xh}}\, x_t + W_{\mathsf{hh}}\, h_{t-1})$$
$$c_t = c_{t-1} + \tilde{c}_t$$

Long short-term memory (LSTM)

Gers et al (1999)

Learning to Forget: Continual Prediction with LSTM



$$f_t = \sigma(W_{\mathsf{fx}}\, x_t + W_{\mathsf{fh}}\, h_{t-1})$$

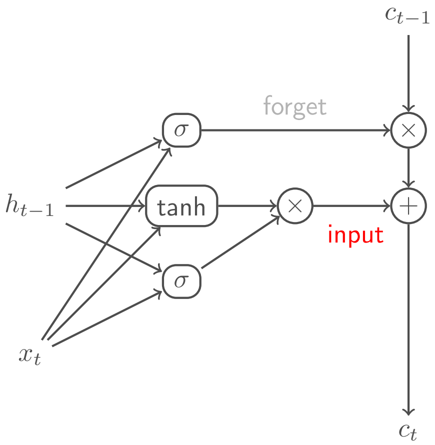
$$\tilde{c}_t = \mathsf{tanh}(W_{\mathsf{xh}}\, x_t + W_{\mathsf{hh}}\, h_{t-1})$$

$$c_t = f_t \odot c_{t-1} + \tilde{c}_t$$

Long short-term memory (LSTM)

Gers et al (1999)

Learning to Forget: Continual Prediction with LSTM



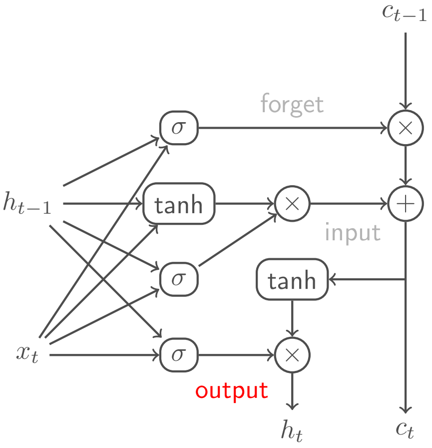
$$f_t = \sigma(W_{\mathsf{fx}}\, x_t + W_{\mathsf{fh}}\, h_{t-1})$$
$$i_t = \sigma(W_{\mathsf{ix}}\, x_t + W_{\mathsf{ih}}\, h_{t-1})$$

$$\tilde{c}_t = \mathrm{tanh}(W_{\mathsf{xh}}\, x_t + W_{\mathsf{hh}}\, h_{t-1})$$
$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t$$

Long short-term memory (LSTM)

Gers et al (1999)

Learning to Forget: Continual Prediction with LSTM



$$f_t = \sigma(W_{\mathsf{fx}}\, x_t + W_{\mathsf{fh}}\, h_{t-1})$$
$$i_t = \sigma(W_{\mathsf{ix}}\, x_t + W_{\mathsf{ih}}\, h_{t-1})$$
$$o_t = \sigma(W_{\mathsf{ox}}\, x_t + W_{\mathsf{oh}}\, h_{t-1})$$

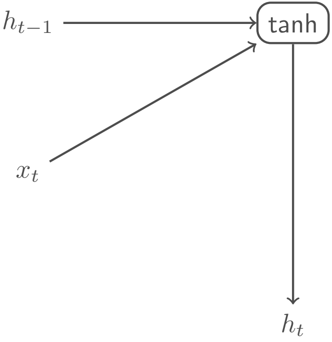
$$\tilde{c}_t = \tanh(W_{\mathsf{xh}}\, x_t + W_{\mathsf{hh}}\, h_{t-1})$$
$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t$$

$$h_t = o_t \odot \tanh(c_t)$$

Gated Recurrent Unit (GRU)

Cho et al. (2014)

Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation

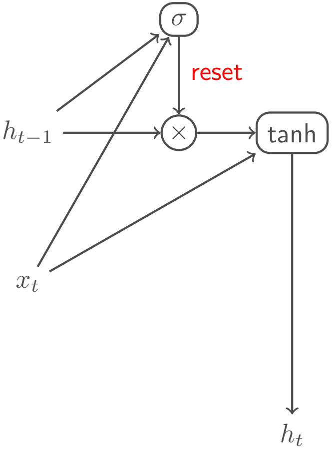


$$\tilde{h}_t = \mathsf{tanh}(W_{\mathsf{xh}}\, x_t + W_{\mathsf{hh}}\, h_{t-1})$$
$$h_t = \tilde{h}_t$$

Gated Recurrent Unit (GRU)

Cho et al. (2014)

Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation



$$r_t = \sigma(W_{\mathsf{rx}}\, x_t + W_{\mathsf{rh}}\, h_{t-1})$$

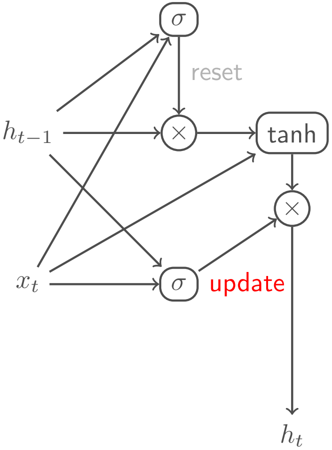
$$\tilde{h}_t = \tanh(W_{\mathsf{xh}}\, x_t + W_{\mathsf{hh}}\, (r_t \odot h_{t-1}))$$

$$h_t = \tilde{h}_t$$

Gated Recurrent Unit (GRU)

Cho et al. (2014)

Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation



$$r_t = \sigma(W_{\mathbf{rx}}\, x_t + W_{\mathbf{rh}}\, h_{t-1})$$

$$z_t = \sigma(W_{\mathbf{zx}}\, x_t + W_{\mathbf{zh}}\, h_{t-1})$$

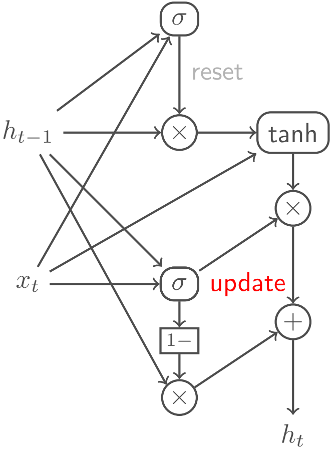
$$\tilde{h}_t = \tanh(W_{\mathbf{xh}}\, x_t + W_{\mathbf{hh}}\, (r_t \odot h_{t-1}))$$

$$h_t = z_t \odot \tilde{h}_t$$

Gated Recurrent Unit (GRU)

Cho et al. (2014)

Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation



$$r_t = \sigma(W_{\mathsf{rx}}\, x_t + W_{\mathsf{rh}}\, h_{t-1})$$

$$\color{red}{z_t} = \sigma(W_{\mathsf{zx}}\, x_t + W_{\mathsf{zh}}\, h_{t-1})$$

$$\tilde{h}_t = \mathrm{tanh}(W_{\mathsf{xh}}\, x_t + W_{\mathsf{hh}}\, (r_t \odot h_{t-1}))$$

$$h_t = \color{red}{(1 - z_t)} \odot h_{t-1} + \color{red}{z_t} \odot \tilde{h}_t$$

NLP scenarios: Classification

Given variable-length text $w_1 \cdots w_n$ (sentence, document, etc), find label y

Normal discriminative approach:

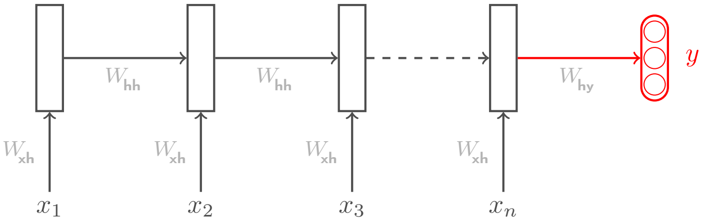
- Extract features over the input text
- Train a linear classifier

Examples:

- Topic classification
- Sentiment analysis
- Entailment recognition

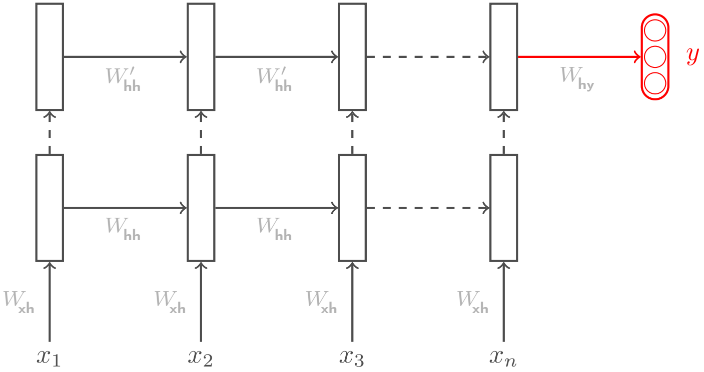
$\vdots$

Classification with RNNs



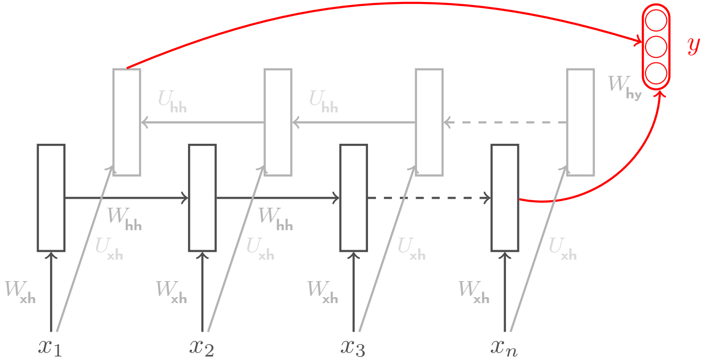
(each x_i is a sparse or dense representation of input word w_i)

Classification with deep RNNs



+ Can learn more abstract representations
– Slow computation because of recurrent dependencies

Classification with bidirectional RNNs

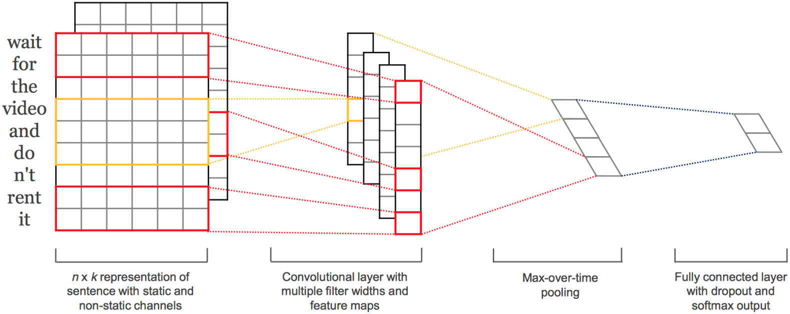


+ Less sensitive to vanishing gradients for long sequences

Classification with CNNs

Kim (2014)

Convolutional Neural Networks for Sentence Classification

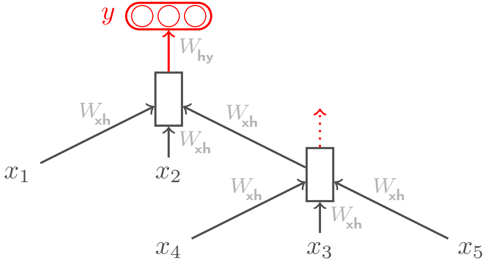


- Max-over-time pooling
- Two input embedding "channels" — one updated during training

Classification with Tree-RNNs

Tai et al (2015)

Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks



Computation graph follows dependency or constituency parse

(i) Child-sum:

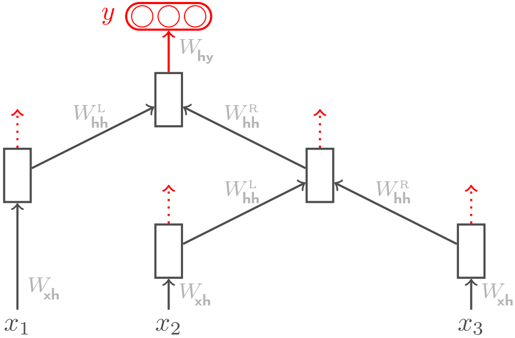
- Good for arbitrary fan-out or unordered children
- Suited to dependency trees (input x_i is head word)

(ii) N-ary

Classification with Tree-RNNs

Tai et al (2015)

Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks



Computation graph follows dependency or constituency parse

(i) Child-sum

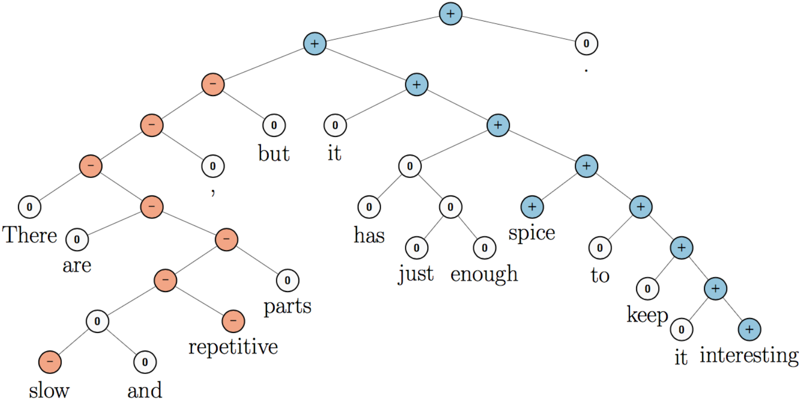
(ii) N-ary:

- Fixed number of children, each parameterized separately
- Suited to binarized constituency parses (leaves take word inputs x_i)

Classification with Tree-RNNs

Socher et al (2013)

Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank



NLP scenarios: Tagging

Given variable-length text $w_1 \cdots w_n$, label spans $z_1, \ldots, z_m$

Normal discriminative approach:

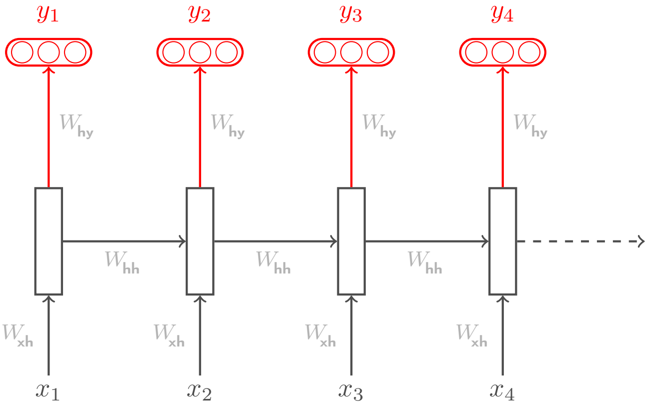
- Distribute labels over input to produce per-word labels $y_1, \ldots, y_n$
  - BIO encoding: Beginning-z, Inside-z, Outside
  - BILOU encoding: Beginning-z, Inside-z, Last-z, Outside, Unit-z
- Extract features over input words
- Train a linear-chain conditional random field

Examples:

- Part-of-speech tagging
- Chunking
- Named entity recognition
- Semantic role labeling

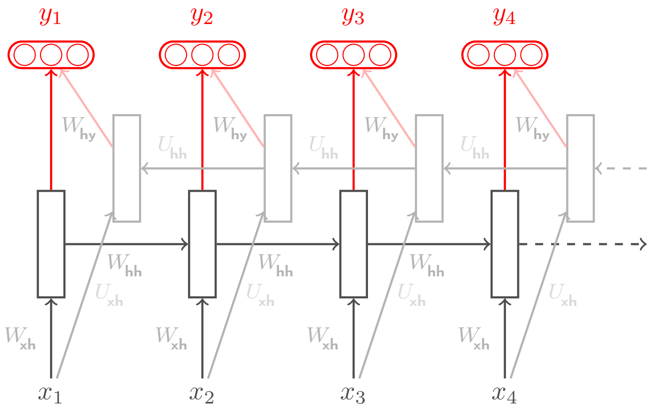
$\vdots$

Tagging with RNNs



+ Effective with BIO/BILOU label encodings

Tagging with bidirectional RNNs



+ Effective with BIO/BILOU label encodings
+ Less sensitive to vanishing gradients for long sequences

NLP scenarios: Generation

Probabilistic language modeling

- Distribution over sequences of words $p(w_1, \ldots, w_T)$ in a language
- Typically made tractable via conditional independence assumptions

$$p(w_1, \ldots, w_n) = \prod_{t=1}^{T} p(w_t | w_{t-1}, \ldots w_{t-n})$$

- n-gram counts estimated from large corpora
- Distributions *smoothed* to tolerate data sparsity, e.g., Laplace (add-one) smoothing, Kneser-Ney smoothing
- Evaluate on *perplexity* over held-out data

$$2^{\frac{1}{N}\sum_{i=1}^{N} \log_2 p\left(w_1^{(i)} \ldots w_{T_i}^{(i)}\right)}$$

NLP scenarios: Generation

Bengio et al (2003)

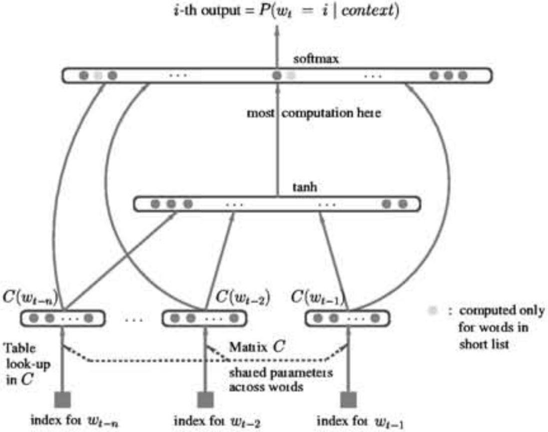
A Neural Probabilistic Language Model

Discriminative language modeling

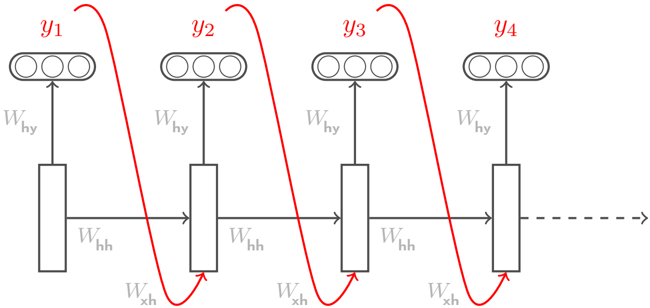
- Estimate n-gram probabilities with a discriminative model

$$p(w_t | w_{t-1}, \ldots w_1) \approx f(w_1, \ldots, w_t)$$

e.g., model $p(w_t | w_{t-1}, \ldots w_{t-n})$ with a feed-forward neural net



Language modeling with auto-regressive RNNs



- Supply one-hot encoding of output y_t as input to timestep $t + 1$
- *Curriculum learning* to overcome model initialization and speed up convergence

RNNLM

Mikolov et al (2010)

Recurrent Neural Network Based Language Model

Model $p(w_t|w_{t-1}, \ldots w_{t-n})$ with an RNN

Or an ensemble of multiple RNNs, randomly initialized

Model	Perplexity
Kneser-Ney 5-gram	141
Random forest [Xu 2005]	132
Structured LM [Filimonov 2009]	125
Feedforward NN LM	116
Syntactic NN LM [Emami 2004]	110
RNN trained by BP	113
RNN trained by BPTT	106
4x RNN trained by BPTT	98

Results on Penn Treebank corpus

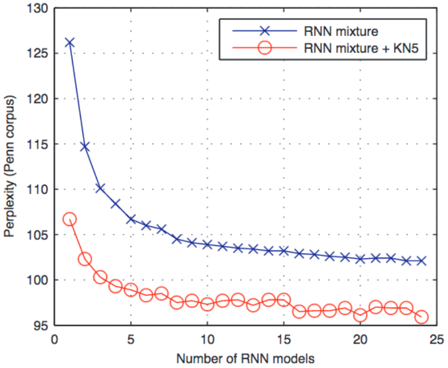
RNNLM

Mikolov et al (2010)

Recurrent Neural Network Based Language Model

Model $p(w_t|w_{t-1}, \ldots w_{t-n})$ with an RNN

Or an ensemble of multiple RNNs, randomly initialized

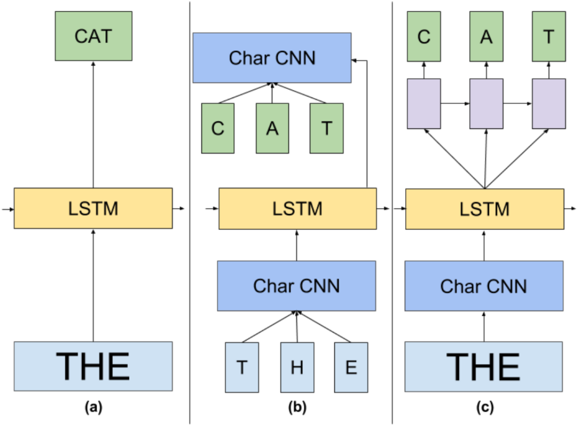


Comparison of single RNN vs RNN ensembles

RNNLMs with character CNNs

Jozefowicz et al (2015)

Exploring the Limits of Language Modeling



Recent models with character-CNN inputs and softmax alternatives

RNNLMs with character CNNs

Jozefowicz et al (2015)

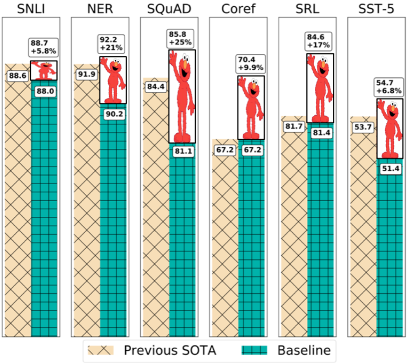
Exploring the Limits of Language Modeling

WORD	TOP-1	TOP-2	TOP-3
INCERDIBLE	INCREDIBLE	NONEDIBLE	EXTENDIBLE
WWW.A.COM	WWW.AA.COM	WWW.AAA.COM	WWW.CA.COM
7546	7646	7534	8566
TOWNHAL1	TOWNHALL	DJC2	MOODSWING360
KOMARSKI	KOHARSKI	KONARSKI	KOMANSKI

Nearest neighbors in character-CNN embedding space

Transfer learning with NNLMs

Pre-trained neural language models are useful in classification tasks



Performance gains with ELMo representations added to various models

Deep Contextualized Word Representations (Peters et al., 2018)

Universal Language Model Fine-tuning for Text Classification (Howard and Ruder, 2018)

Improving Language Understanding by Generative Pre-Training (Radford et al., 2018)

NLP scenarios: Text-to-text

Given variable-length text $x_1 \cdots x_n$ (sentence, document, etc), produce output text $y_1, \cdots, y_m$ under some transformation

Traditional approaches:

- Pipelined models
- Constrained optimization

$\vdots$

Examples:

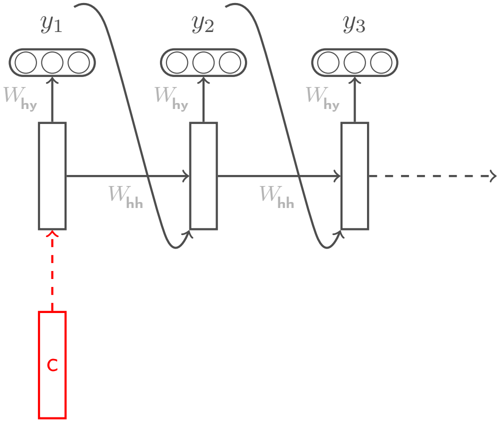
- Machine translation
- Document summarization
- Sentence simplification
- Paraphrase generation

$\vdots$

Decoding with RNNs

Input context c

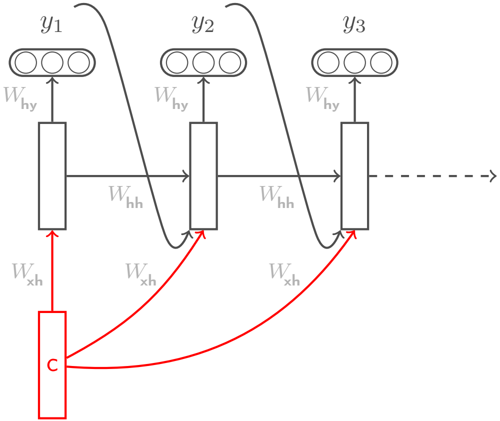
Output words $y_1, \ldots, y_m$



Decoding with RNNs

Input context c

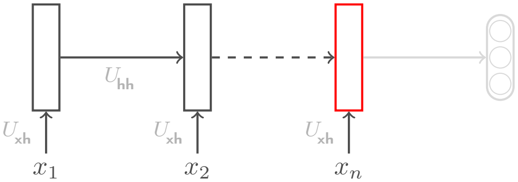
Output words $y_1, \ldots, y_m$



Encoding with RNNs

Input words $x_1, \dots, x_n$

Output representation h_n



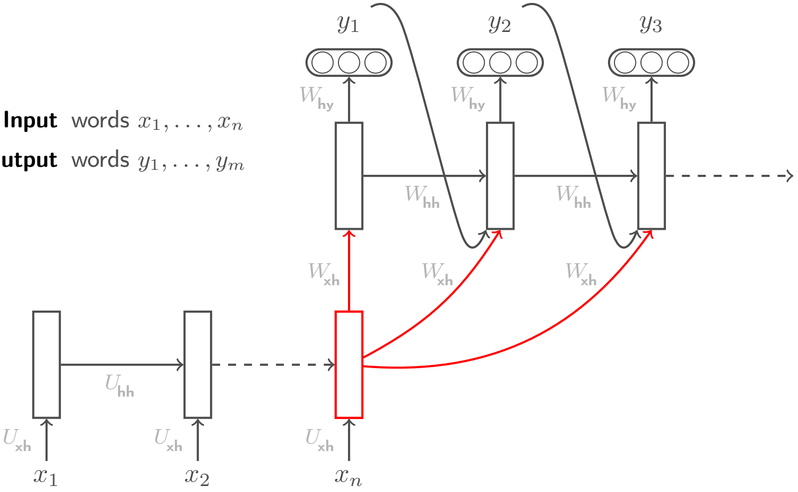
Sequence-to-sequence learning

Sutskever, Vinyals & Le (2014)

Sequence to Sequence Learning with Neural Networks

Input words $x_1, \ldots, x_n$

Output words $y_1, \ldots, y_m$



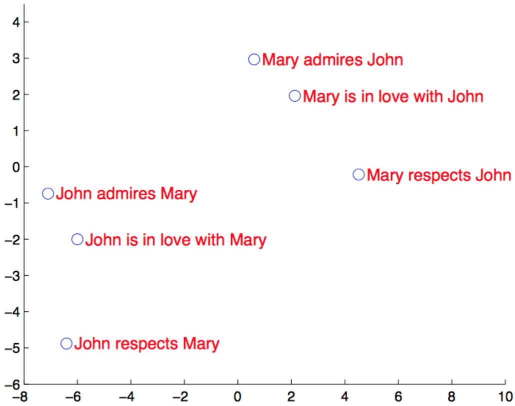
Sequence-to-sequence learning

Sutskever, Vinyals & Le (2014)

Sequence to Sequence Learning with Neural Networks

Produces a fixed length representation of input

- "sentence embedding" or "thought vector"



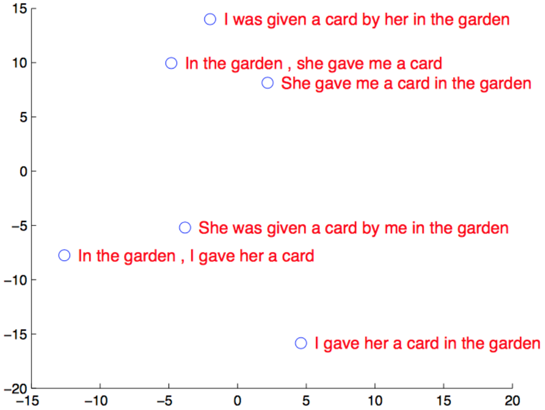
Sequence-to-sequence learning

Sutskever, Vinyals & Le (2014)

Sequence to Sequence Learning with Neural Networks

Produces a fixed length representation of input

- "sentence embedding" or "thought vector"



Overview: processing text with RNNs

Inputs

- One-hot vectors for words/characters/previous output
- Embeddings for words/sentences/context
- CNN over characters/words/sentences

⋮

Recurrent layers

- Forward, backward, bidirectional, deep
- Activations: σ, tanh, gated (LSTM, GRU), ReLU initialized with identity

⋮

Outputs

- Softmax over words/characters/labels
- Absent (i.e., pure encoders)

⋮