TTIC 31210:
Advanced Natural Language Processing

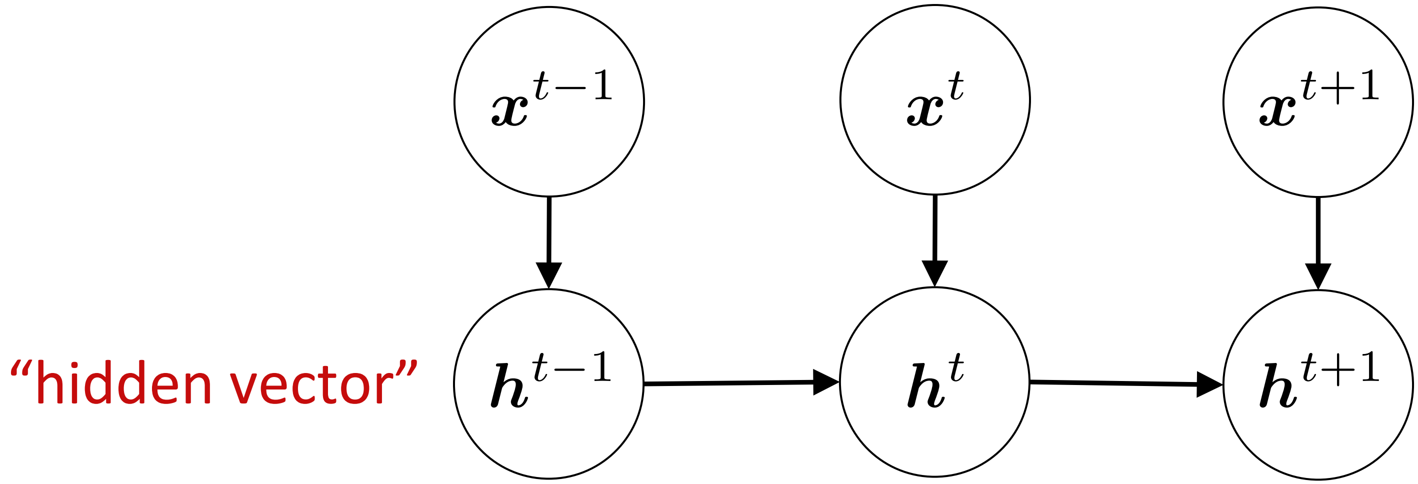
Kevin Gimpel

Spring 2017

Lecture 5:
Sequence-to-Sequence Modeling and Sentence Embeddings

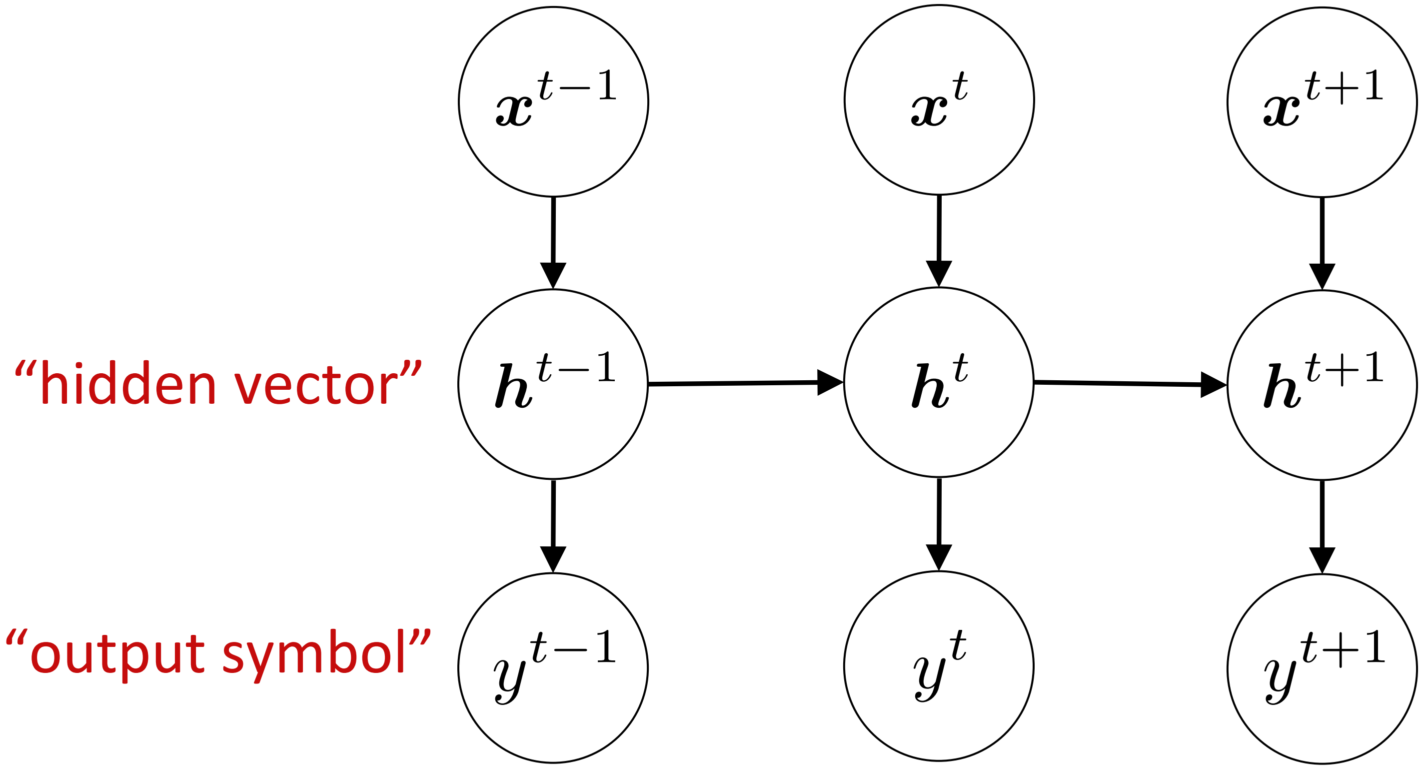
Recurrent Neural Networks

$$\boldsymbol{h}^t = \tanh\left(W^{(x)}\boldsymbol{x}^t + W^{(h)}\boldsymbol{h}^{t-1} + \boldsymbol{b}^{(h)}\right)$$



“Output” Recurrent Neural Networks

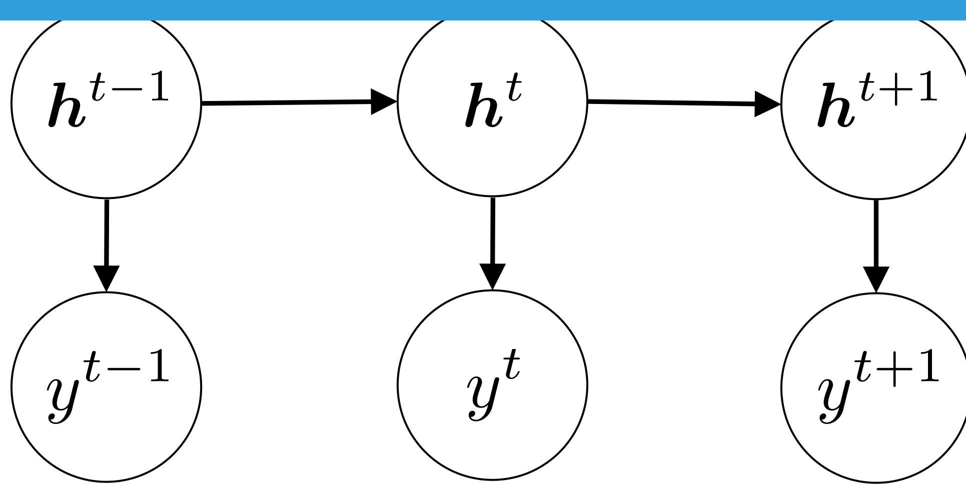
$$\boldsymbol{h}^t = \tanh\left(W^{(x)}\boldsymbol{x}^t + W^{(h)}\boldsymbol{h}^{t-1} + \boldsymbol{b}^{(h)}\right)$$



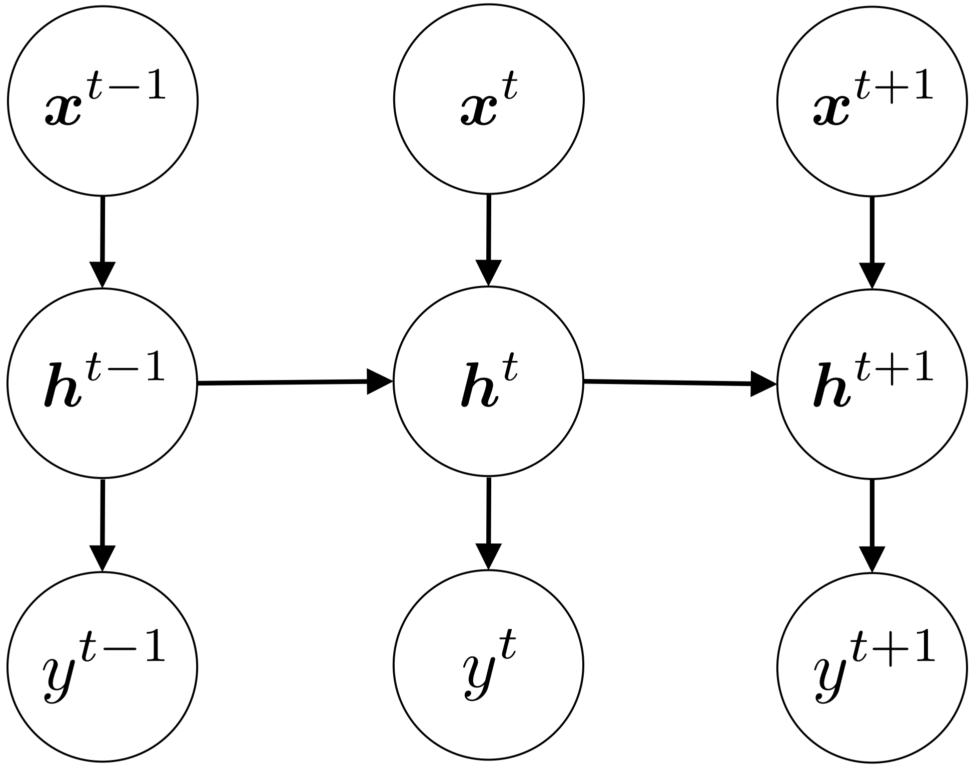
$$y^t = \operatorname*{argmax}_{y \in \mathcal{O}} \left(emb(y)^\top \boldsymbol{h}^t\right)$$

ks

- *y* is a symbol, not a vector
- *O* is the “output” vocabulary
- we have a new parameter vector *emb(y)* for each output symbol in *O*
- *emb(y)* = **x**?
- probability distribution over output symbols?



$$y^t = \operatorname*{argmax}_{y \in \mathcal{O}} \left(emb(y)^\top \boldsymbol{h}^t \right)$$

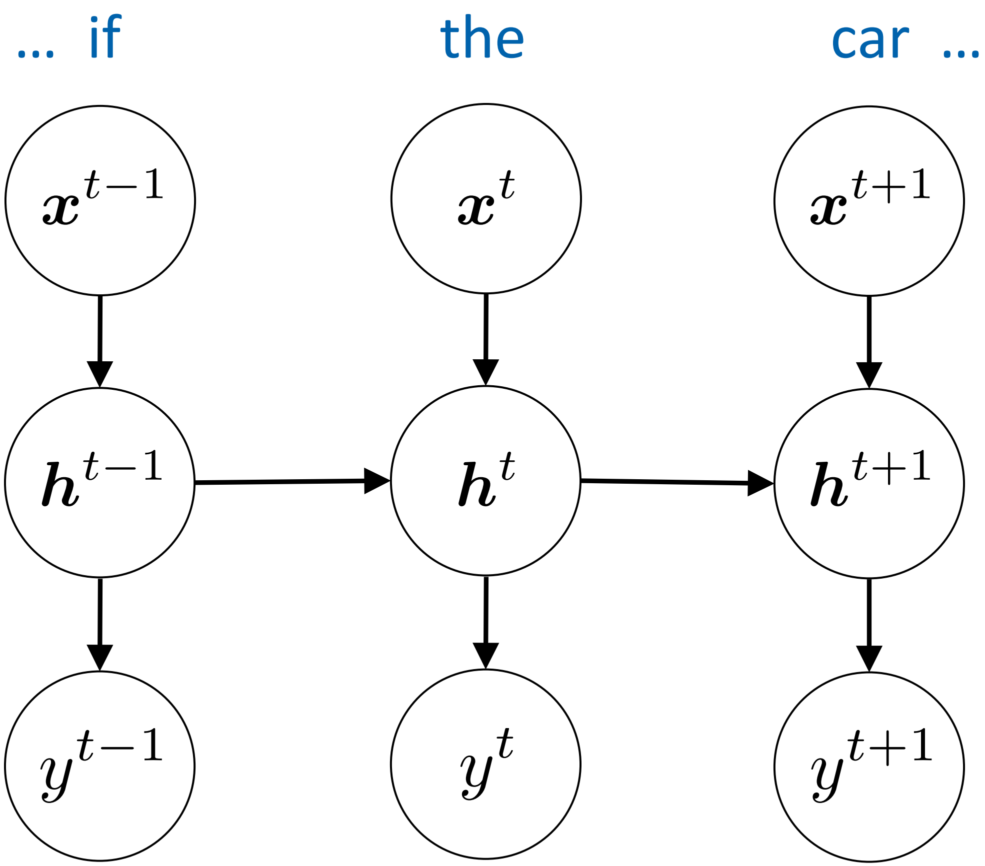


$$y^t = \operatorname*{argmax}_{y \in \mathcal{O}} \left(emb(y)^\top \boldsymbol{h}^t \right)$$

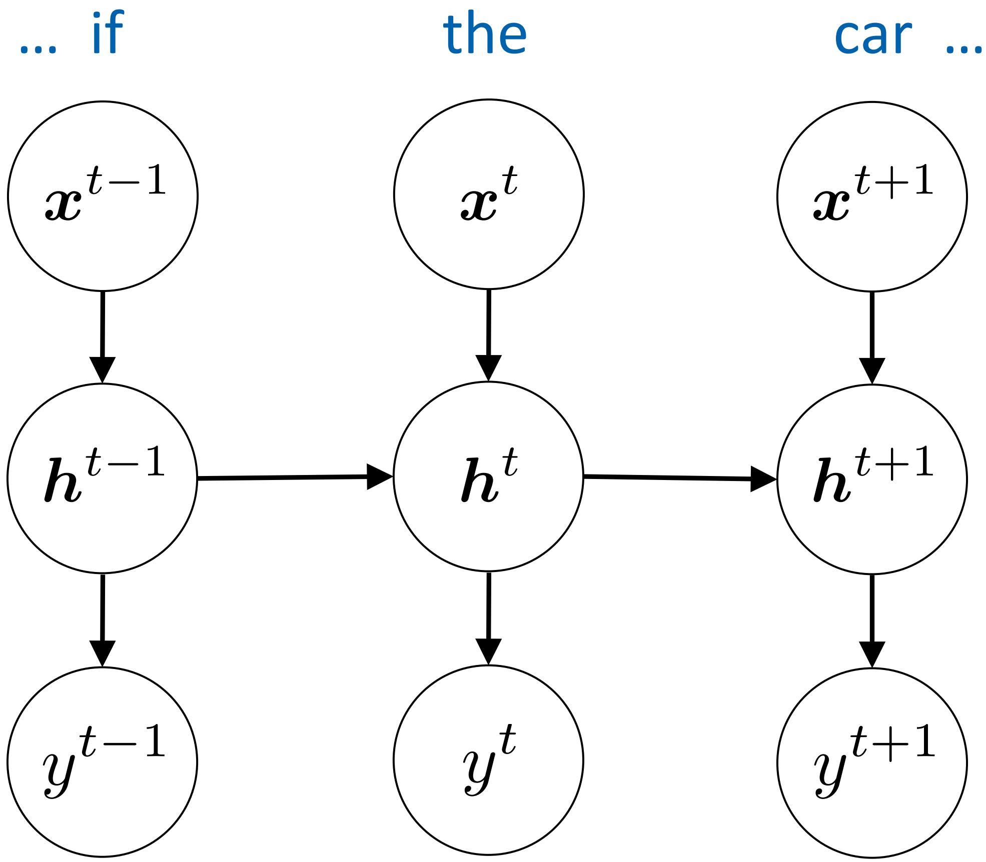
$$P(Y^t) = \operatorname{softmax}\left(W \boldsymbol{h}^t \right)$$

$$W = \left[emb(y_1)^\top ; emb(y_2)^\top ; ... ; emb(y_{|\mathcal{O}|})^\top \right]$$

Example: Language Modeling

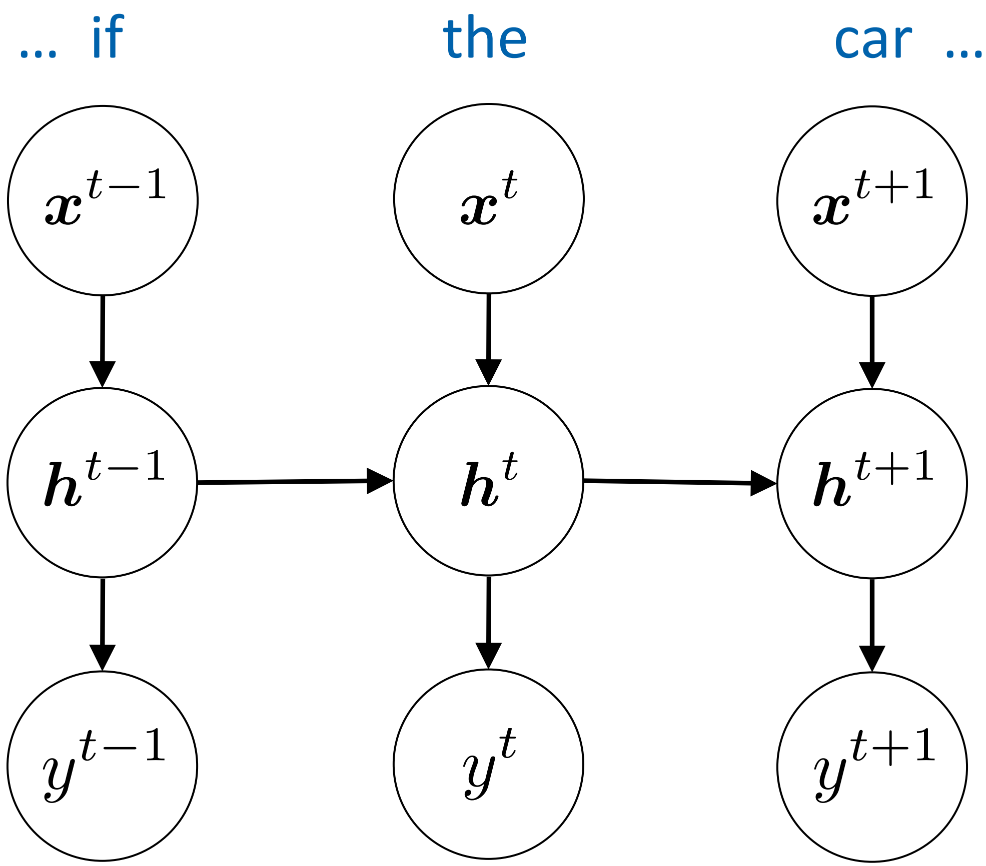


Language Modeling: Training



$$-\log P(Y^{t-1} = \quad ? \quad)$$

Language Modeling: Training



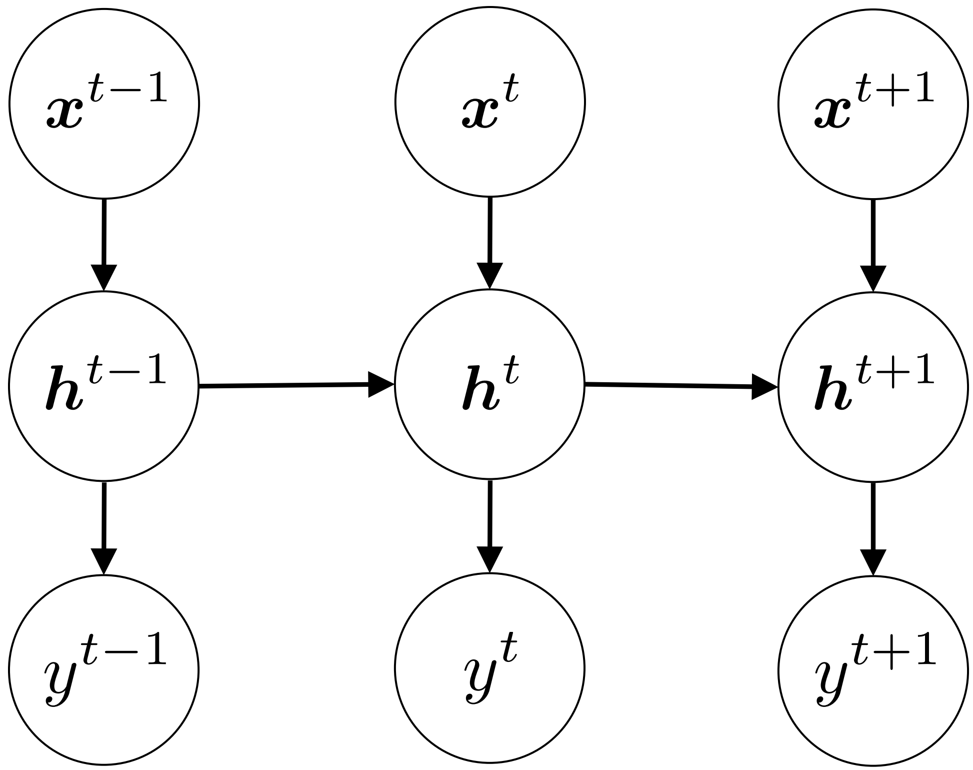
$$-\log P(Y^{t-1} = \text{“the”}) - \log P(Y^{t} = \text{“car”}) \quad \ldots$$

Language Modeling: Decoding

- we'll use the term "decoding" to mean roughly "test-time inference"
- for language modeling, decoding means "output the highest-probability sentence"

Language Modeling: Decoding

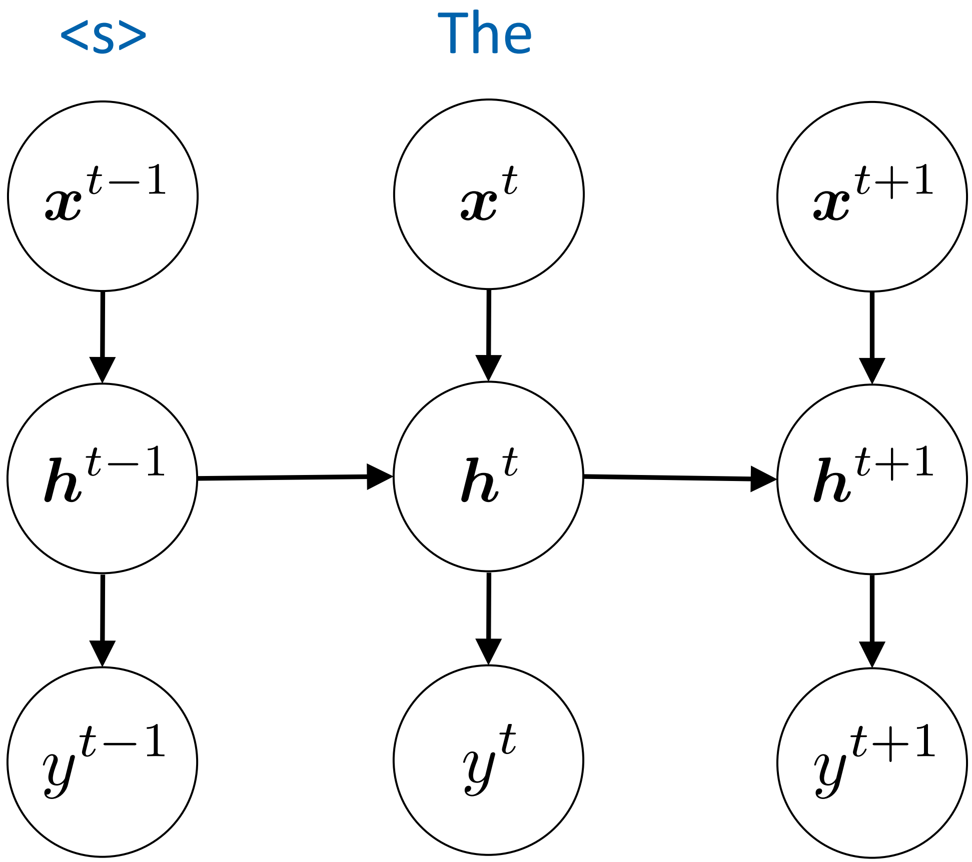
<s>



$$y^{t-1} = \underset{y \in \mathcal{O}}{\operatorname{argmax}} \left(emb(y)^\top \boldsymbol{h}^{t-1} \right)$$

"The"

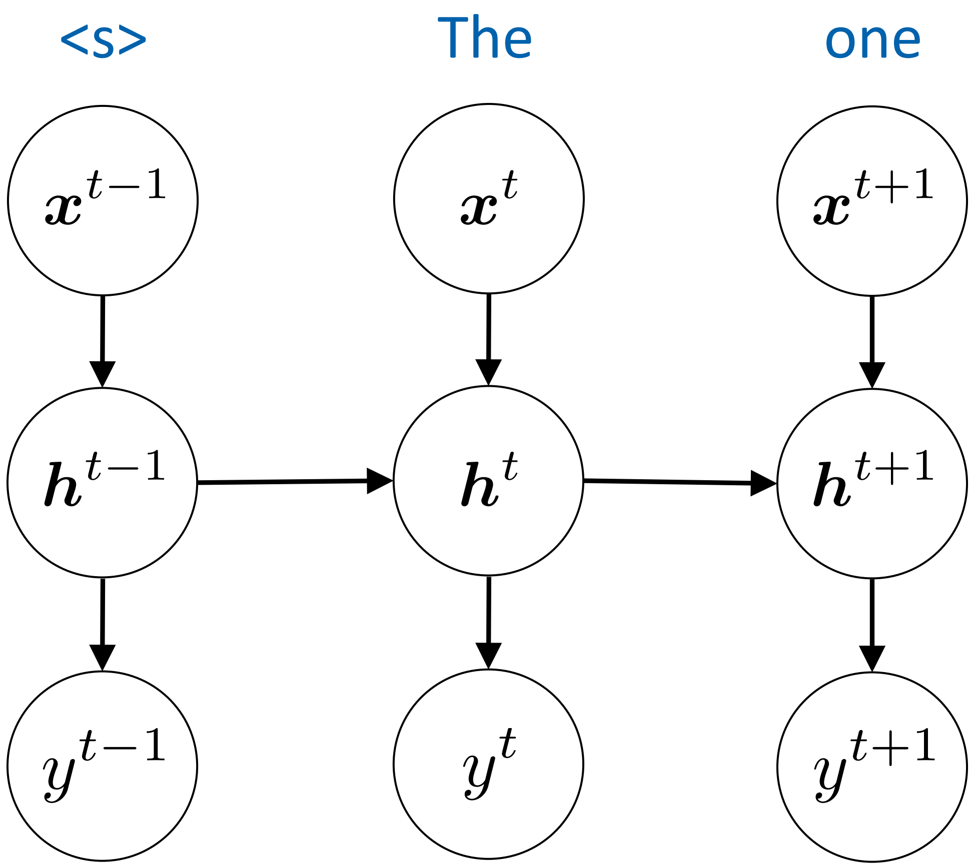
Language Modeling: Decoding



$$y^t = \operatorname*{argmax}_{y\in\mathcal{O}} \left(emb(y)^\top \boldsymbol{h}^t\right)$$

“one”

Language Modeling: Decoding



Concern

- there's a mismatch between training and test!
- (what is it?)

Sequence-to-Sequence Modeling

- data: <input sequence, output sequence> pairs
- use one neural network to encode input sequence as a vector
- use an output RNN to generate the output sequence (conditioned on the input sequence vector)
- more generally called “encoder-decoder” architectures

Recurrent Continuous Translation Models

EMNLP 2013

Nal Kalchbrenner **Phil Blunsom**
Department of Computer Science
University of Oxford

Abstract

We introduce a class of probabilistic continuous translation models called Recurrent Continuous Translation Models that are purely based on continuous representations for words, phrases and sentences and do not rely on alignments or phrasal translation units. The models have a generation and a conditioning aspect. The generation of the translation is modelled with a target Recurrent Language Model, whereas the conditioning on the source sentence is modelled with a Convolutional Sentence Model. Through various experiments, we show first that our models obtain a perplexity with respect to gold translations that is $>43\%$ lower than that of state-of-the-art alignment-based translation models.

Recurrent Continuous Translation Models

EMNLP 2013

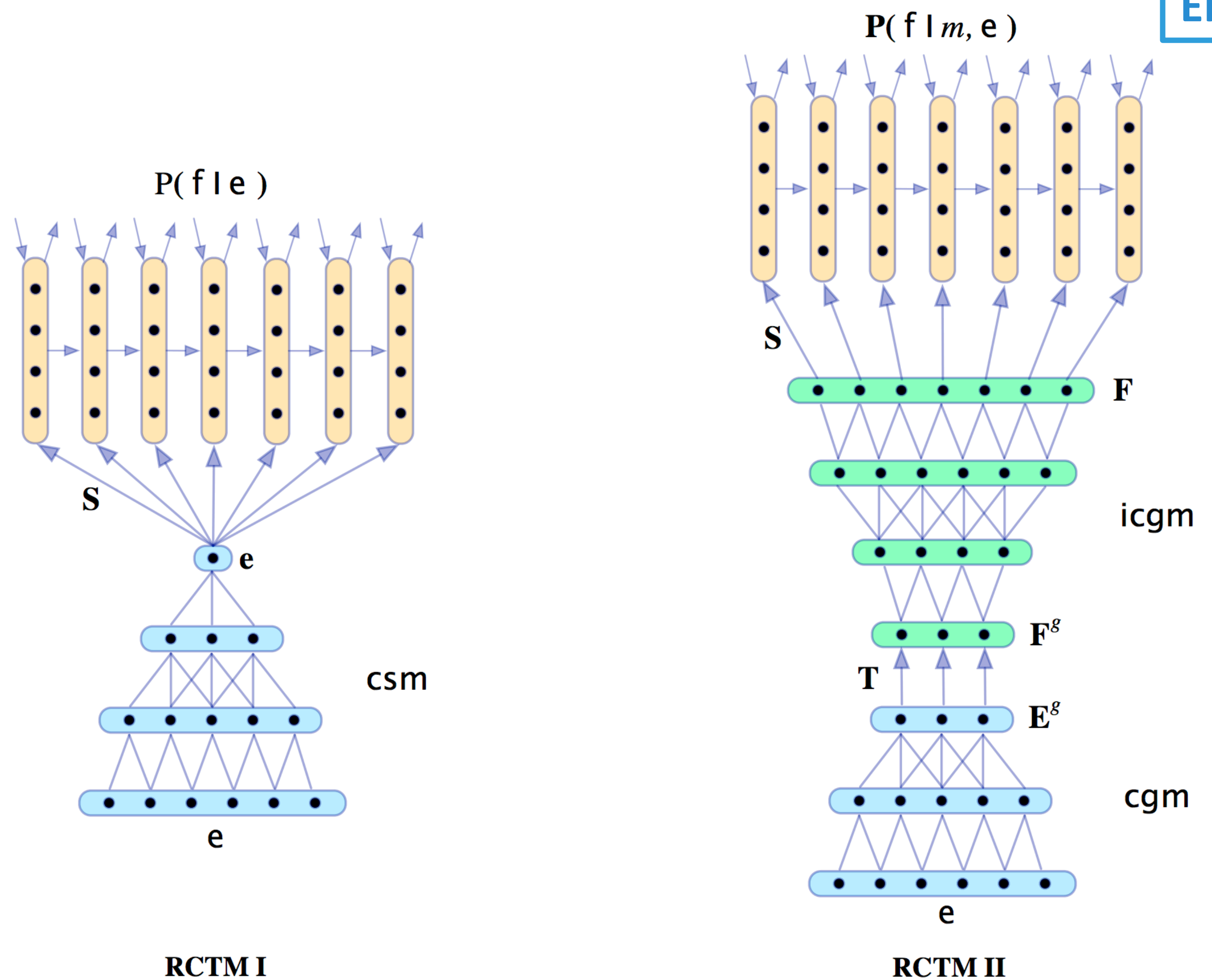


Figure 3: A graphical depiction of the two RCTMs. Arrows represent full matrix transformations while lines are vector transformations corresponding to columns of weight matrices.

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

EMNLP 2014

Kyunghyun Cho
Bart van Merriënboer **Caglar Gulcehre**
Université de Montréal
firstname.lastname@umontreal.ca

Dzmitry Bahdanau
Jacobs University, Germany
d.bahdanau@jacobs-university.de

Fethi Bougares **Holger Schwenk**
Université du Maine, France
firstname.lastname@lium.univ-lemans.fr

Yoshua Bengio
Université de Montréal, CIFAR Senior Fellow
find.me@on.the.web

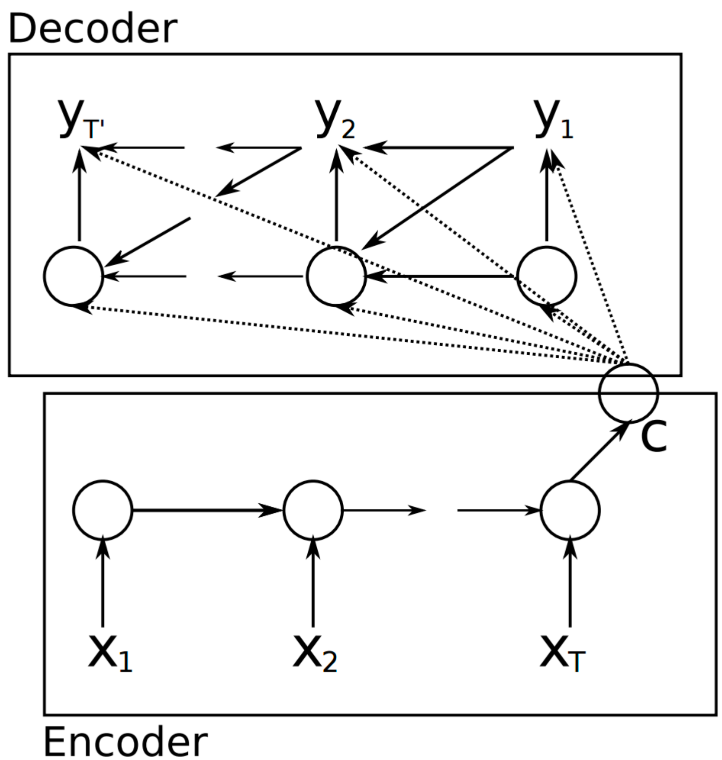


Figure 1: An illustration of the proposed RNN Encoder–Decoder.

Sequence to Sequence Learning with Neural Networks

NIPS 2014

Ilya Sutskever
Google
ilyasu@google.com

Oriol Vinyals
Google
vinyals@google.com

Quoc V. Le
Google
qvl@google.com

Abstract

Deep Neural Networks (DNNs) are powerful models that have achieved excellent performance on difficult learning tasks. Although DNNs work well whenever large labeled training sets are available, they cannot be used to map sequences to sequences. In this paper, we present a general end-to-end approach to sequence learning that makes minimal assumptions on the sequence structure. Our method uses a multilayered Long Short-Term Memory (LSTM) to map the input sequence to a vector of a fixed dimensionality, and then another deep LSTM to decode the target sequence from the vector. Our main result is that on an English to French translation task from the WMT-14 dataset, the translations produced by the LSTM achieve a BLEU score of 34.8 on the entire test set, where the LSTM's BLEU score was penalized on out-of-vocabulary words. Additionally, the LSTM did not have difficulty on long sentences. For comparison, a phrase-based SMT system achieves a BLEU score of 33.3 on the same dataset. When we used the LSTM to rerank the 1000 hypotheses produced by the aforementioned SMT system, its BLEU score increases to 36.5, which is close to the previous state of the art. The LSTM also learned sensible phrase and sentence representations that are sensitive to word order and are relatively invariant to the active and the passive voice. Finally, we found that reversing the order of the words in all source sentences (but not target sentences) improved the LSTM's performance markedly, because doing so introduced many short term dependencies between the source and the target sentence which made the optimization problem easier.

Sequence to Sequence Learning with Neural Networks

NIPS 2014

Ilya Sutskever
Google
ilyasu@google.com

Oriol Vinyals
Google
vinyals@google.com

Quoc V. Le
Google
qvl@google.com

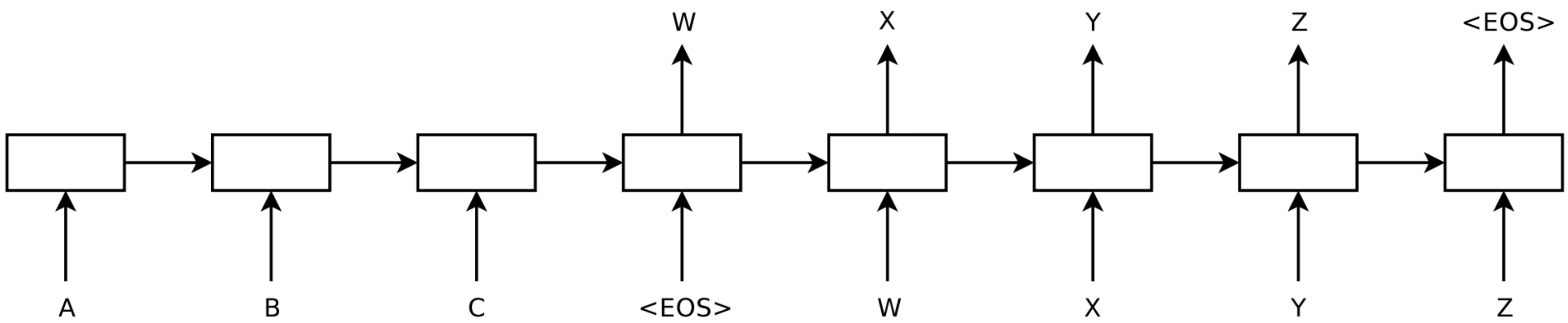


Figure 1: Our model reads an input sentence "ABC" and produces "WXYZ" as the output sentence. The model stops making predictions after outputting the end-of-sentence token. Note that the LSTM reads the input sentence in reverse, because doing so introduces many short term dependencies in the data that make the optimization problem much easier.

Unsupervised Sentence Models

- how should we evaluate sentence models?
- we consider two ways here:
  - sentence similarity:
    - two sentences with similar meanings should have high cosine similarities
    - metric: corr. between similarities & human judgments
  - sentence classification:
    - train a linear classifier using the fixed sentence representation as the input features
    - metric: average accuracy over 6 tasks

Evaluation 1: Semantic Textual Similarity (STS)

Other ways are needed.

We must find other ways.

4.4

I absolutely do believe there was an iceberg in those waters.

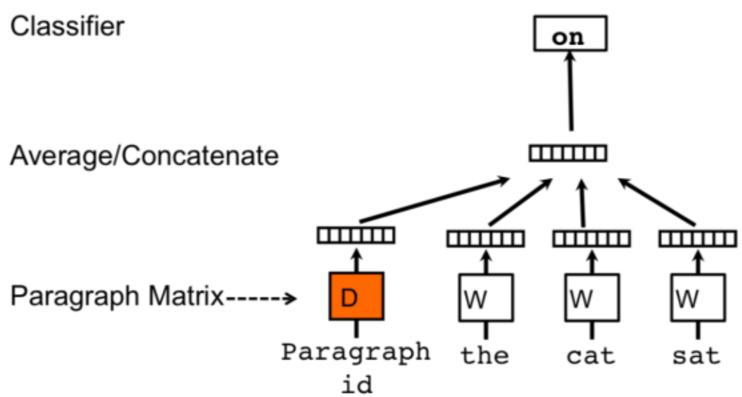
I don't believe there was any iceberg at all anywhere near the Titanic.

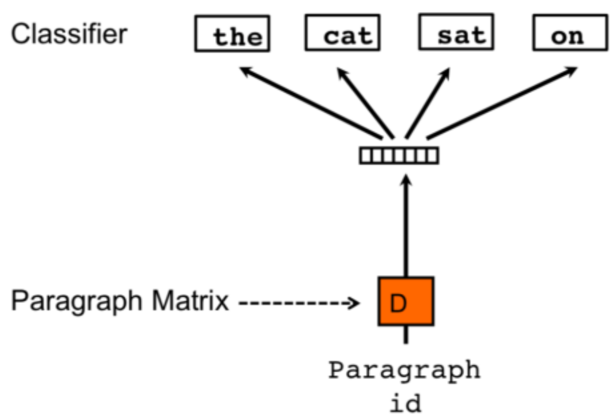
1.2

Results reported on datasets from 6 domains

Paragraph Vectors

- Represent sentence (or paragraph) by predicting its own words or context words





Le & Mikolov (2014)

Evaluation

Sentence Embedding Model	STS	Classification
Paragraph Vector	44	70.4

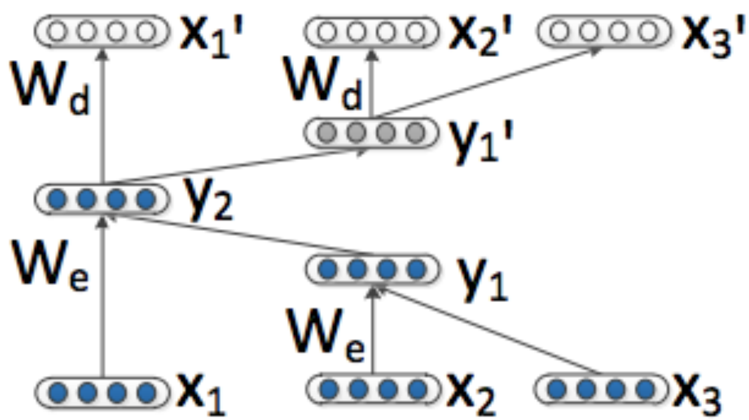
Hill, Cho, Korhonen (2016)

Sentence Autoencoders

- encode sentence as vector, then decode it
- minimize reconstruction error (using squared error or cross entropy) of original words in sentence

Recursive Neural Net Autoencoders

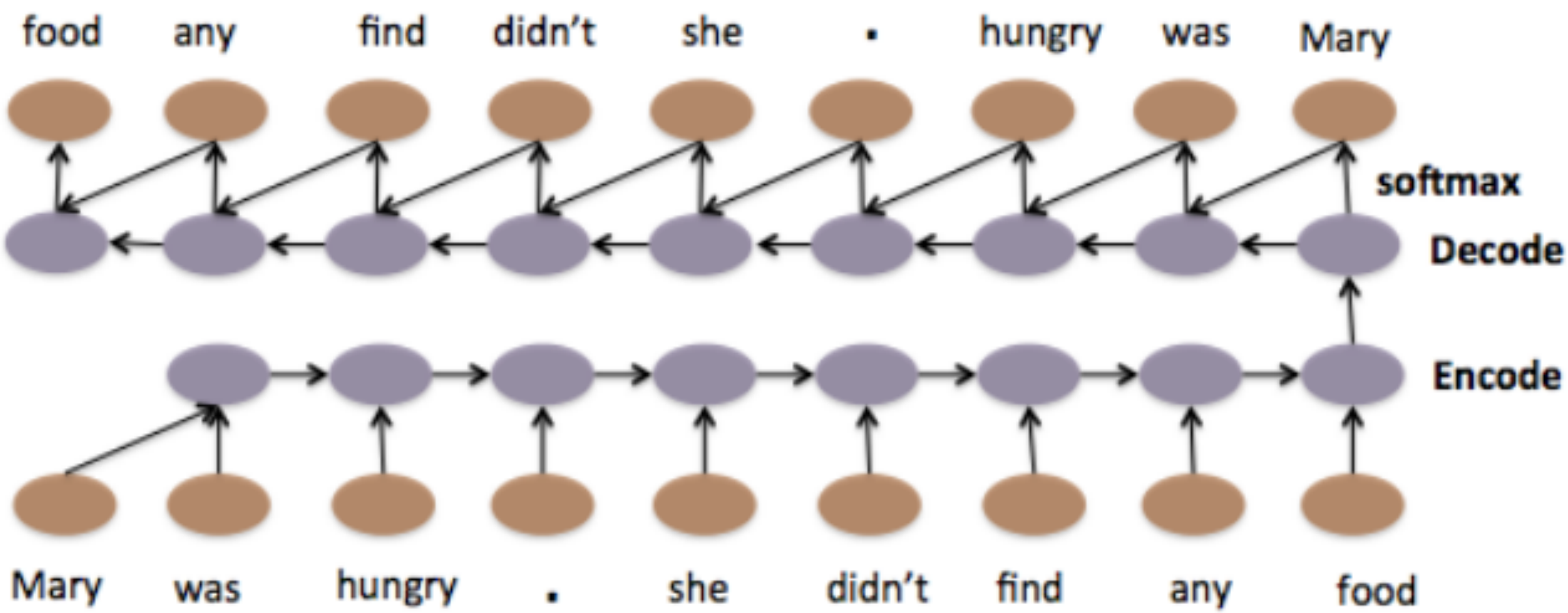
- composition based on syntactic parse



Socher, Huang, Pennington, Ng, Manning (2011)

LSTM Autoencoders

- Encode sentence, decode sentence



Li, Luong, Jurafsky (2015)

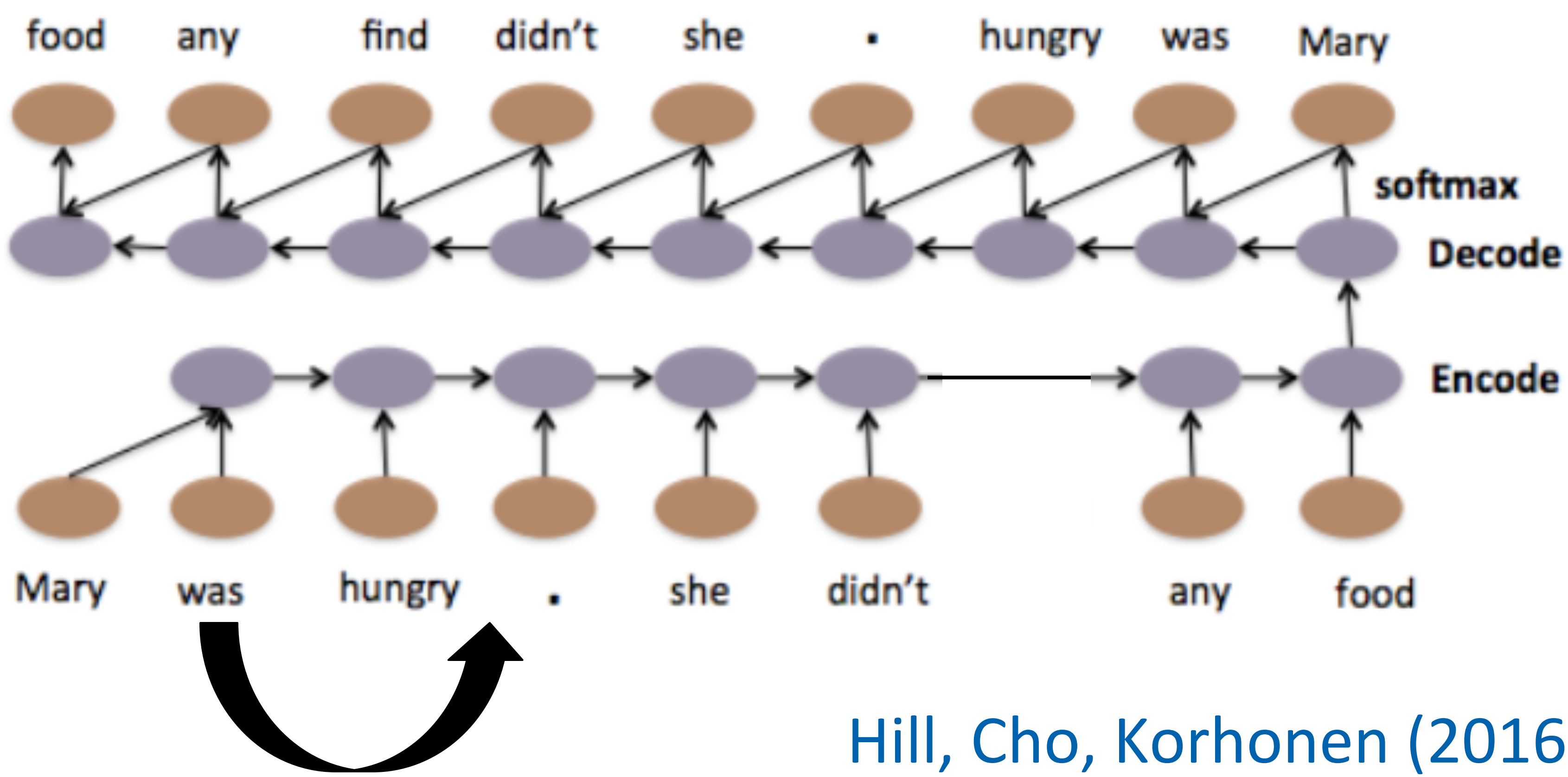
Evaluation

Sentence Embedding Model	STS	Classification
Paragraph Vector	44	70.4
LSTM Autoencoder	43	75.9

Hill, Cho, Korhonen (2016)

LSTM Denoising Autoencoders

- Encode "corrupted" sentence, decode sentence



Hill, Cho, Korhonen (2016)

Evaluation

Sentence Embedding Model	STS	Classification
Paragraph Vector	44	70.4
LSTM Autoencoder	43	75.9
LSTM Denoising Autoencoder	38	78.9

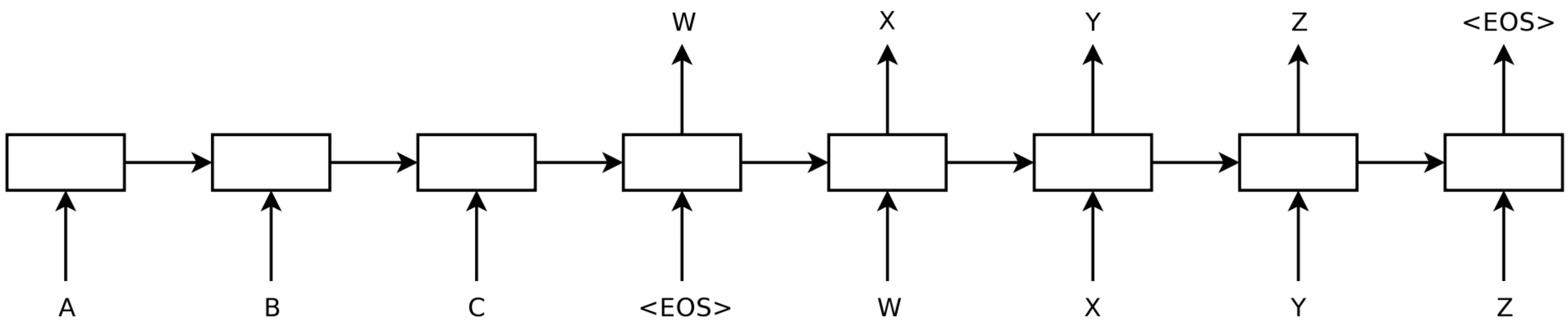
Hill, Cho, Korhonen (2016)

Other Training Criteria for Sentence Embeddings

- encode sentence, decode other things from it:
  - decode its translation
  - decode neighboring sentences
  - predict words in the sentence and predict words in neighboring sentences

Neural Machine Translation

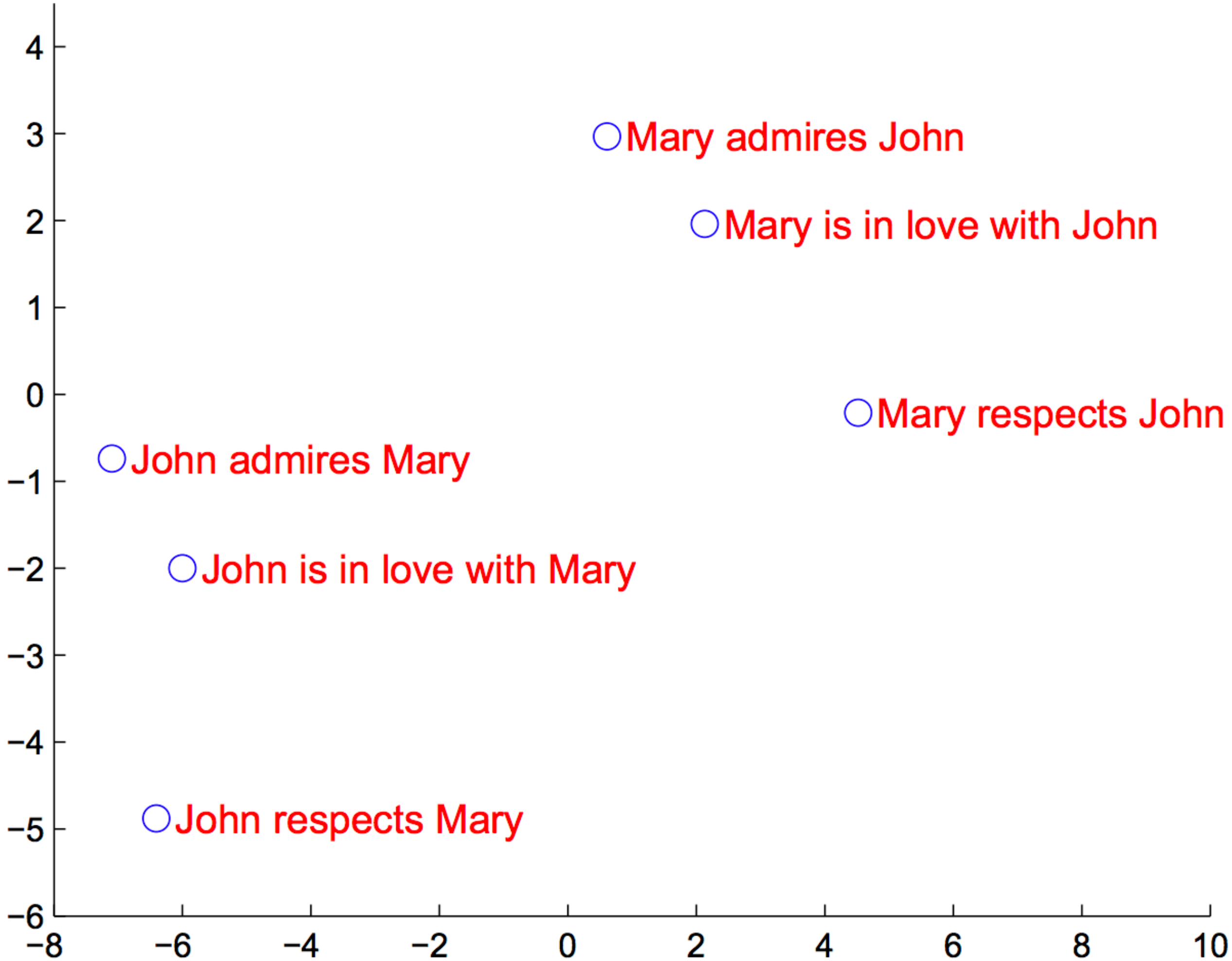
- Encode source sentence, decode translation



Sutskever, Vinyals, Le (2014)

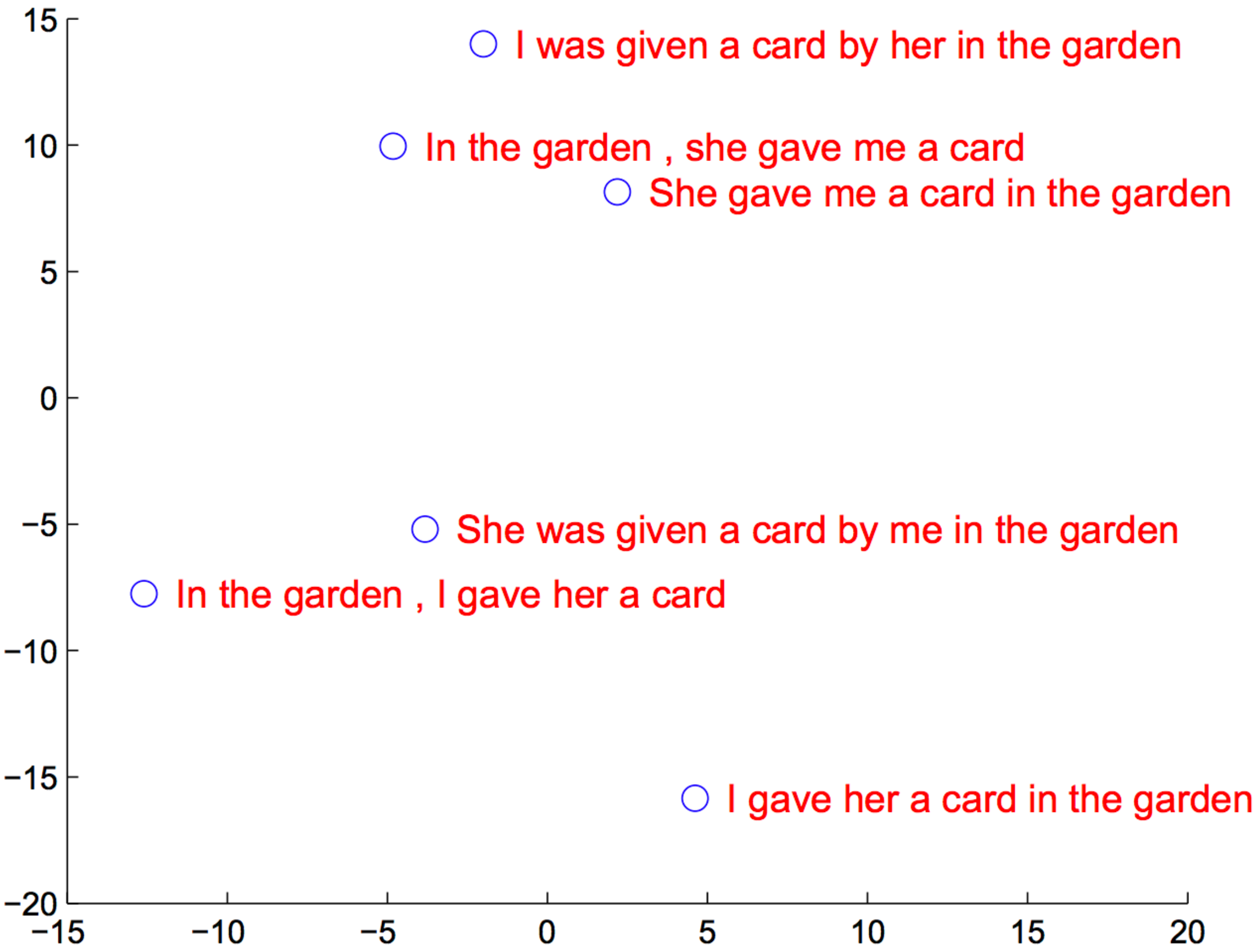
Cho, van Merrienboer, Gulcehre, Bahdanau, Bougares, Schwenk, Bengio (2014)

Encoder as a Sentence Embedding Model?

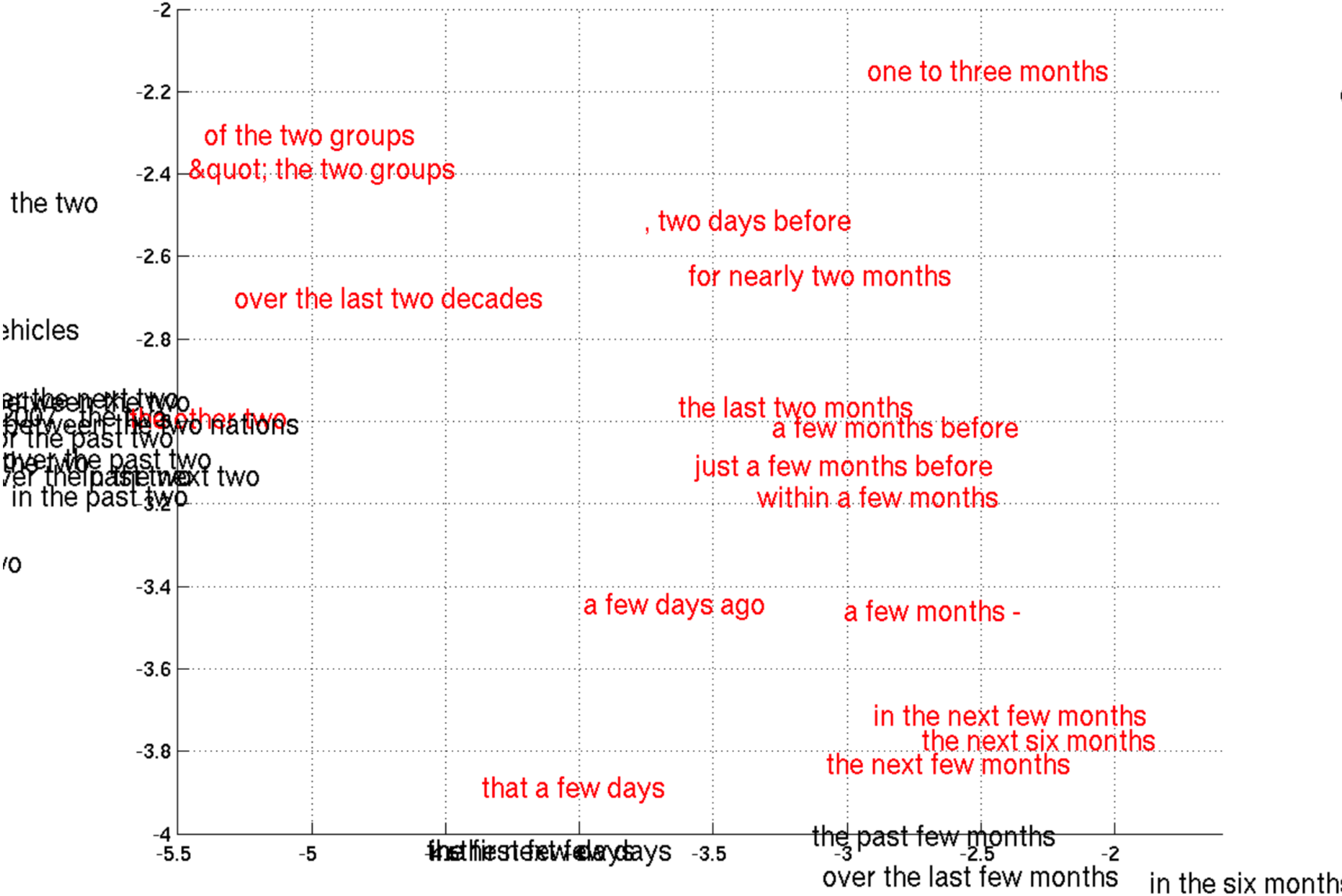


Sutskever, Vinyals, Le (2014)

Encoder as a Sentence Embedding Model?



Sutskever, Vinyals, Le (2014)



Cho, van Merrienboer, Gulcehre, Bahdanau, Bougares, Schwenk, Bengio (2014)

Evaluation

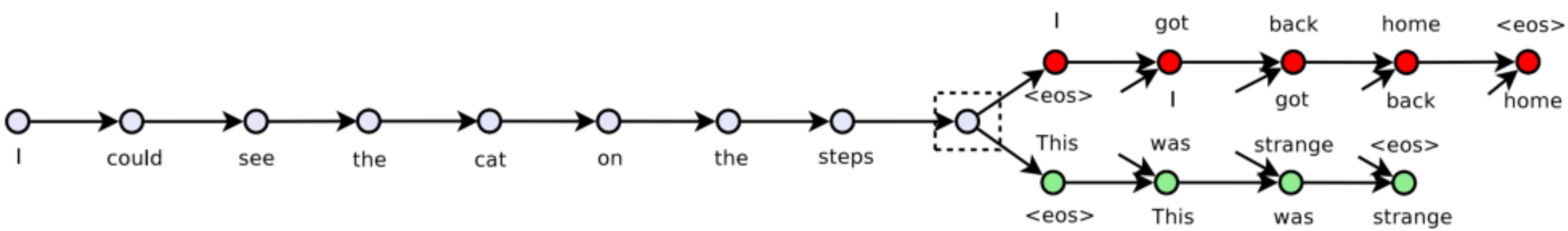
Sentence Embedding Model	STS	Classification
Paragraph Vector	44	70.4
LSTM Autoencoder	43	75.9
LSTM Denoising Autoencoder	38	78.9
Neural MT Encoder	42	76.9

Hill, Cho, Korhonen (2016)

Skip-Thoughts

- encode sentence using an RNN
- decode two neighboring sentences
- use different RNNs for previous and next sentences
- also pass center sentence vector on each decoding step

…I got back home   I could see the cat on the steps   This was strange …



Kiros, Zhu, Salakhutdinov, Zemel, Torralba, Urtasun, Fidler (2015)

Skip-Thoughts

query sentence:

im sure youll have a glamorous evening , she said , giving an exaggerated wink .

nearest neighbor:

im really glad you came to the party tonight , he said , turning to her .

Kiros, Zhu, Salakhutdinov, Zemel, Torralba, Urtasun, Fidler (2015)

Skip-Thoughts

query sentence:

if he had a weapon , he could maybe take out their last imp , and then beat up errol and vanessa .

nearest neighbor:

if he could ram them from behind , send them sailing over the far side of the levee , he had a chance of stopping them .

Kiros, Zhu, Salakhutdinov, Zemel, Torralba, Urtasun, Fidler (2015)

Evaluation

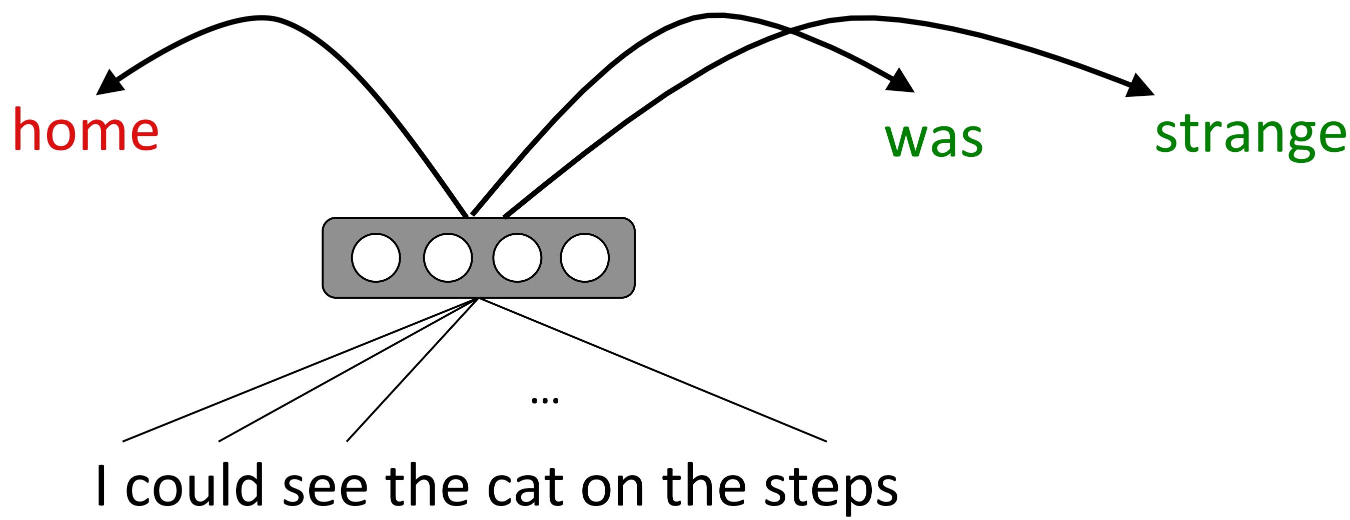
Sentence Embedding Model	STS	Classification
Paragraph Vector	44	70.4
LSTM Autoencoder	43	75.9
LSTM Denoising Autoencoder	38	78.9
Neural MT Encoder	42	76.9
Skip Thought	31	85.3

Hill, Cho, Korhonen (2016)
Wieting, Bansal, Gimpel, Livescu (2016)

FastSent

- encode center sentence using sum
- decode to predict words in neighboring sentences
- different embedding spaces for “input” and “output” words

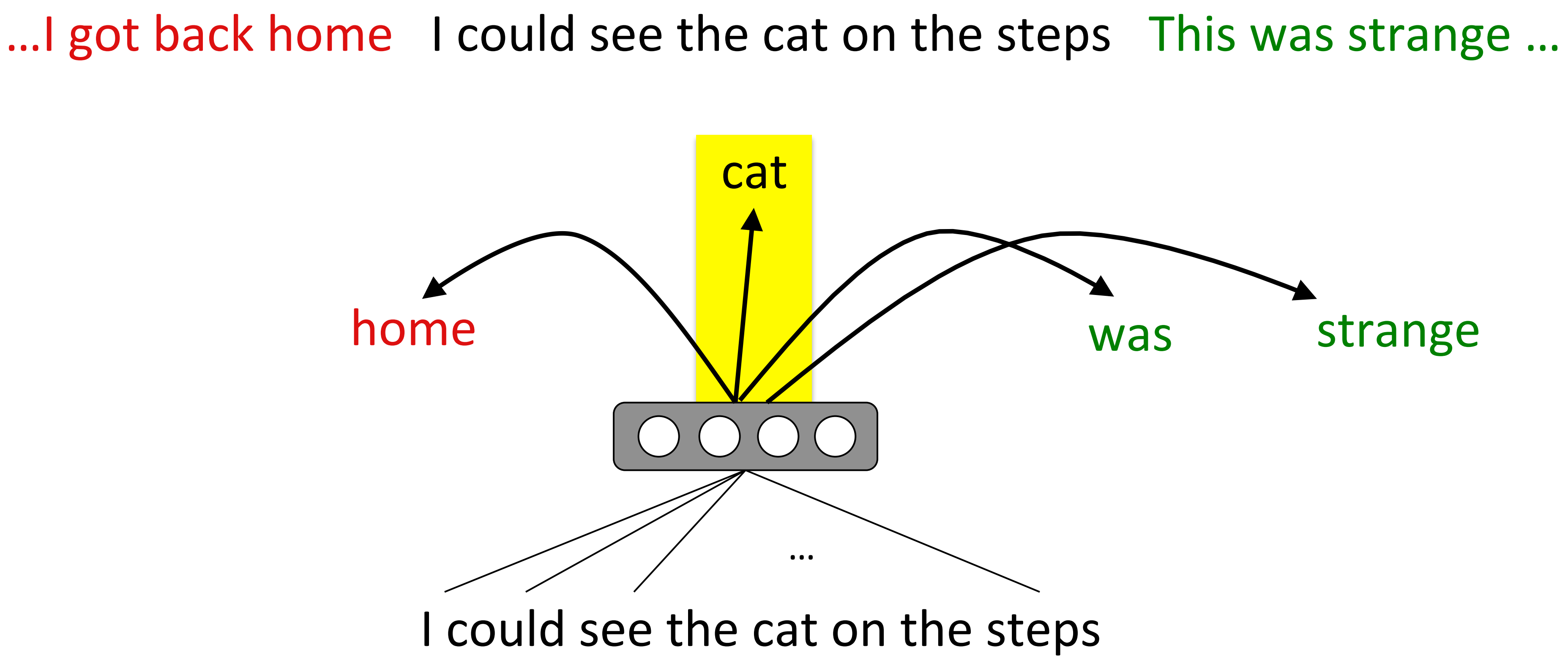
…I got back home I could see the cat on the steps This was strange …



Hill, Cho, Korhonen (2016)

FastSent + Autoencoder

- encode center sentence using sum
- decode to predict words in **current+**neighboring sentences



Hill, Cho, Korhonen (2016)

Evaluation

Sentence Embedding Model	STS	Classification
Paragraph Vector	44	70.4
LSTM Autoencoder	43	75.9
LSTM Denoising Autoencoder	38	78.9
Neural MT Encoder	42	76.9
Skip Thought	31	85.3
FastSent	64	79.3
FastSent + Autoencoder	62	79.7

Hill, Cho, Korhonen (2016)
Wieting, Bansal, Gimpel, Livescu (2016)

Sentence Embedding Model	STS	Classification
Paragraph Vector	44	70.4
LSTM Autoencoder	43	75.9
LSTM Denoising Autoencoder	38	78.9
Neural MT Encoder	42	76.9
Skip Thought	31	85.3
FastSent	64	79.3
FastSent + Autoencoder	62	79.7
C-PHRASE	67	81.7

Hill, Cho, Korhonen (2016)

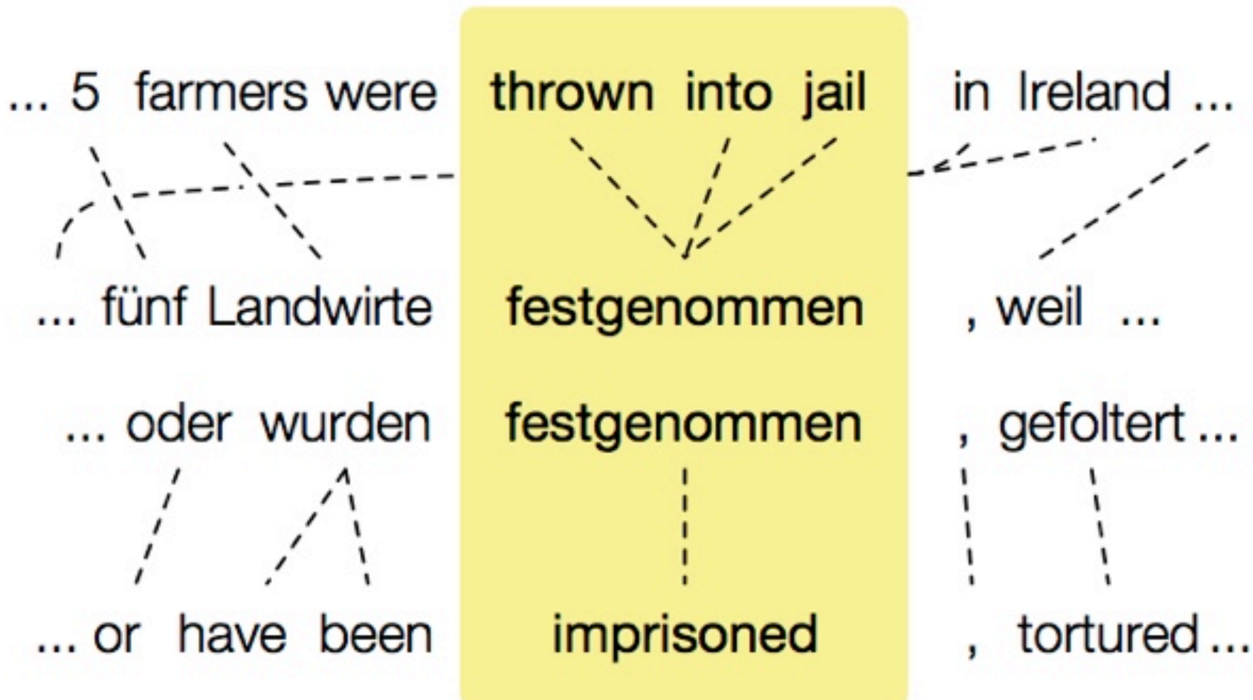
Wieting, Bansal, Gimpel, Livescu (2016)

Sentence Embedding Model	STS	Classification
Paragraph Vector	44	70.4
LSTM Autoencoder	43	75.9
LSTM Denoising Autoencoder	38	78.9
Neural MT Encoder	42	76.9
Skip Thought	31	85.3
FastSent	64	79.3
FastSent + Autoencoder	62	79.7
C-PHRASE	67	81.7
Avg. pretrained word embeddings	65	80.6

Hill, Cho, Korhonen (2016)
Wieting, Bansal, Gimpel, Livescu (2016)

Paraphrase Database (PPDB)

(Ganitkevitch, Van Durme, and Callison-Burch, 2013)



credit: Chris Callison-Burch

Training Data: phrase pairs from PPDB

good	great
be given the opportunity to	have the possibility of
i can hardly hear you .	you 're breaking up .
and the establishment	as well as the development
laying the foundations	pave the way
making every effort	to do its utmost
…	…

tens of millions more!

Sentence Embedding Model	STS	Classification
Paragraph Vector	44	70.4
LSTM Autoencoder	43	75.9
LSTM Denoising Autoencoder	38	78.9
Neural MT Encoder	42	76.9
Skip Thought	31	85.3
FastSent	64	79.3
FastSent + Autoencoder	62	79.7
C-PHRASE	67	81.7
Avg. pretrained word embeddings	65	80.6
Ours (avg. trained on PPDB)	71	N/A

Hill, Cho, Korhonen (2016)

Wieting, Bansal, Gimpel, Livescu (2016)