University of Zurich[UZH]

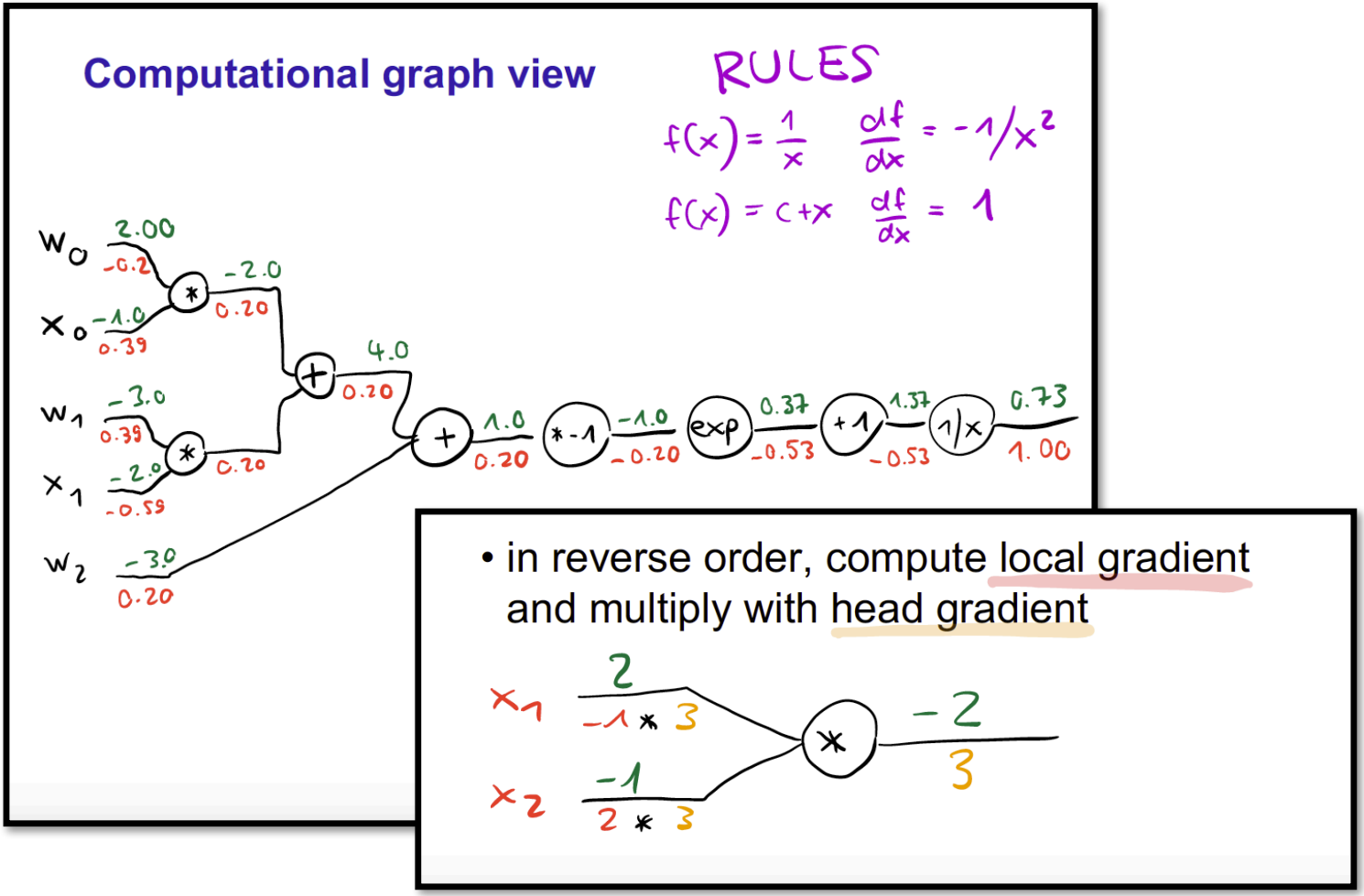
Institute of Computational Linguistics

Machine Translation

8 Word Embeddings Recurrent Neural Networks

Mathias Müller

Last Time: FFNNs plus how to train them



Topics of this lesson

- Word Embeddings
- Recurrent Neural Networks (RNNs)
- RNN Language Models

Why those topics?

- Standard NMT systems train embedding layers

- Standard NMT systems are built with recurrent neural networks

University of Zurich[UZH]

Institute of Computational Linguistics

Word Embeddings

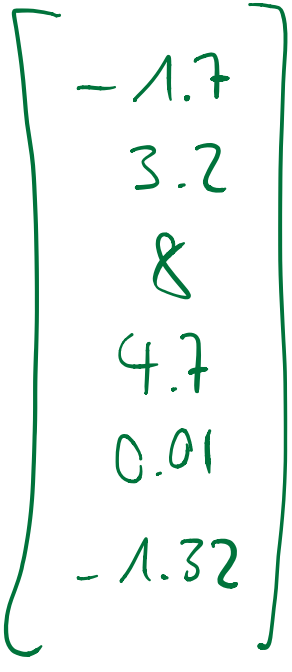
Word Embeddings

- Answer to the question: how to use text as input or output for a neural network?

- Word embeddings are vectors with floats, one embedding vector for each item in the vocabulary

What a word embedding looks like

"cauliflower" ⟶



Why can text not be used directly as input for NNs?

- Text is discrete, neural networks only take vectors of floats as input

- Why not one-hot vectors?

$V = \{$ "car", "cat", "cauliflower" $\}$

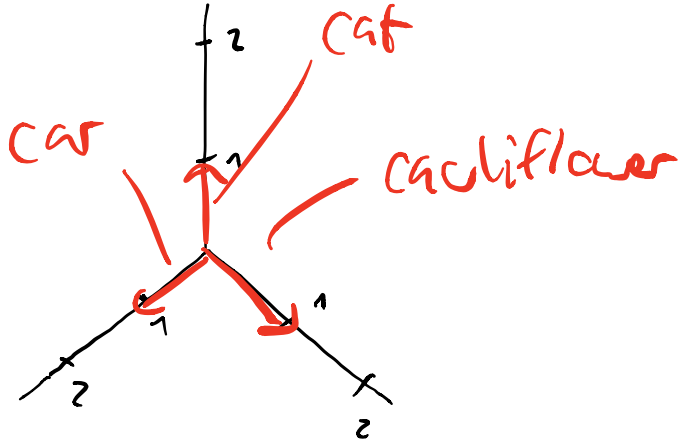
$$\text{car} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

$$\text{cat} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

$$\text{cauliflower} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

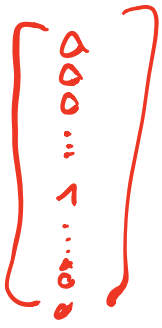
Desirable properties of representations of text

Distance in vector space ≈ distance in meaning

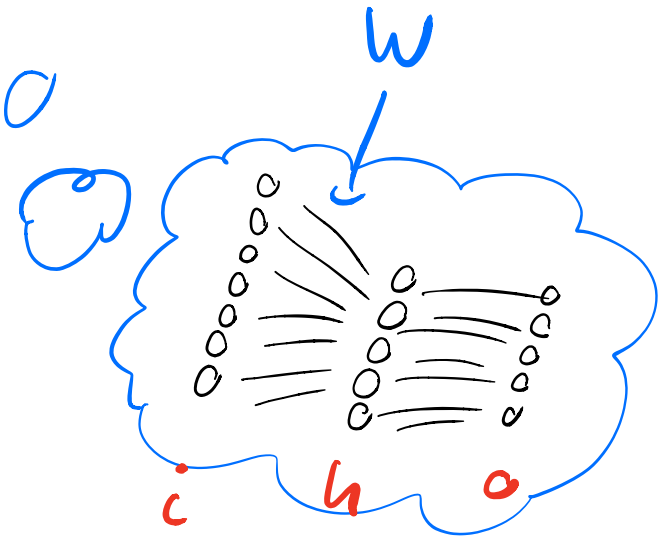


Desirable properties of representations of text

Dense representations generally less wasteful than sparse representations



$|V| = 10'000$



$h = 100$

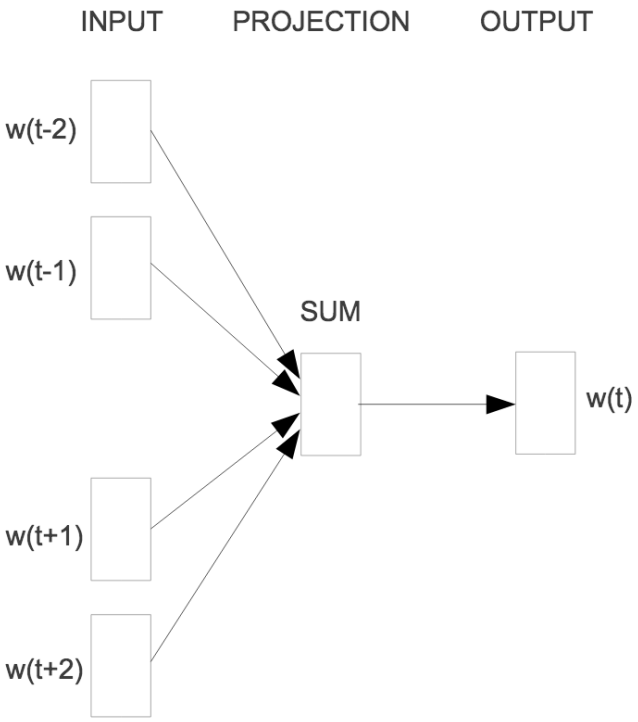
W

$100 \times 10'000$

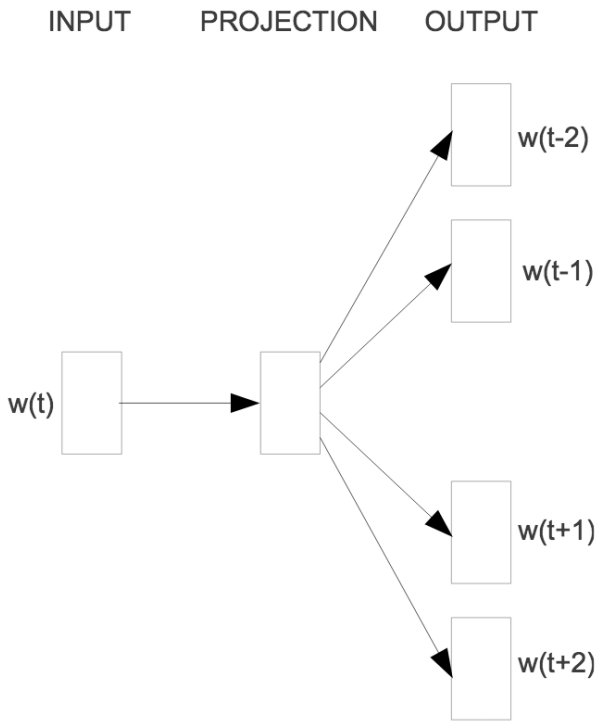
How to train word embeddings

- Trained on large monolingual corpora
- Key idea:
  - take a shallow neural network with one hidden layer
  - make it predict words in context
  - after training, the weight matrix of the hidden layer will be the embeddings

Original algorithms of word2vec



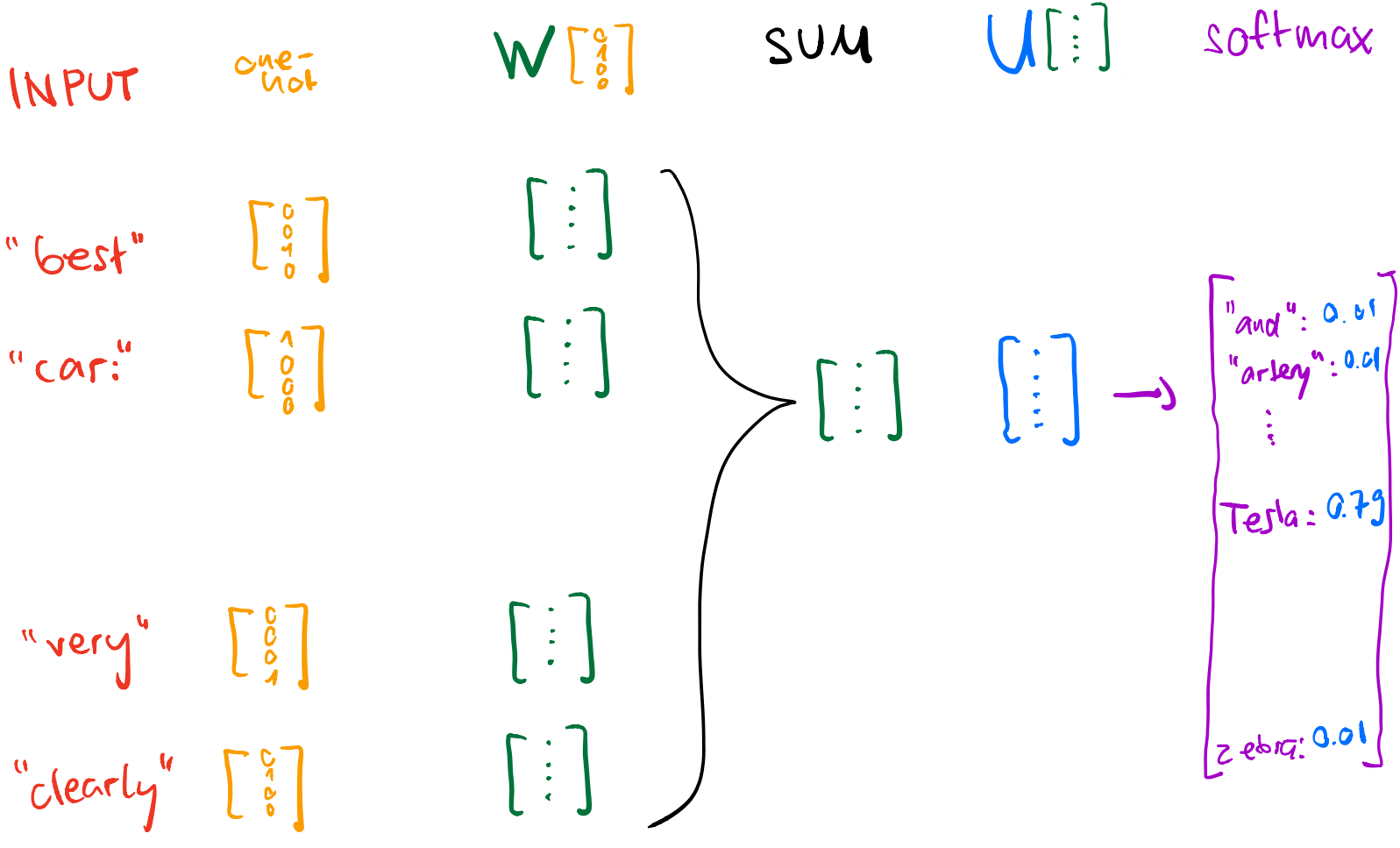
CBOW



Skip-gram

Glove FastText ELMo Bert

CBOW: Computation in more detail



Q&A that helps clarify word embeddings

- What if a word was not seen during training?
- What does “embedding size” mean?
- Why is this called “embedding”?

bit.ly/2TØGDeK

Summary Word Embeddings

- One-hot vectors usually unsuitable as input for neural networks
- Embeddings are a learnable **representation** for text, e.g. words
- Embeddings are trained on monolingual corpora, by predicting words in context

University of Zurich[UZH]

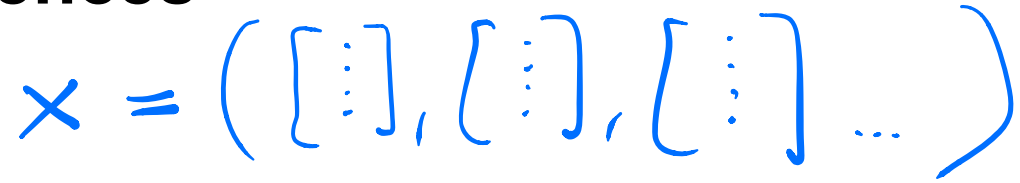
Institute of Computational Linguistics

Recurrent Neural Networks

Recurrent neural networks (RNNs)

- Extension of feed-forward networks

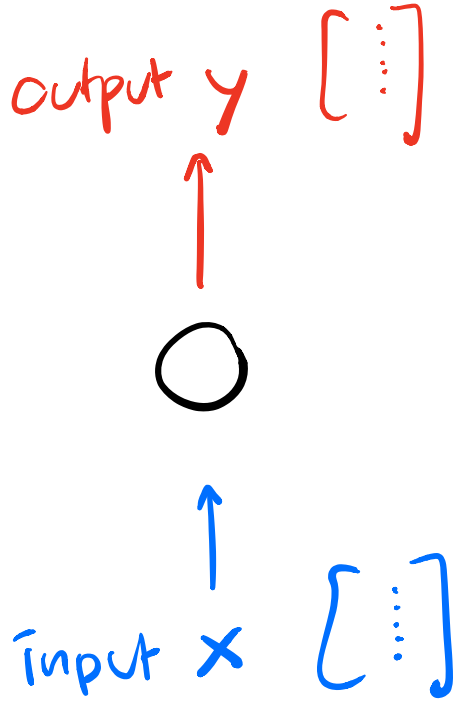
- Main purpose: read or write **ordered sequences**



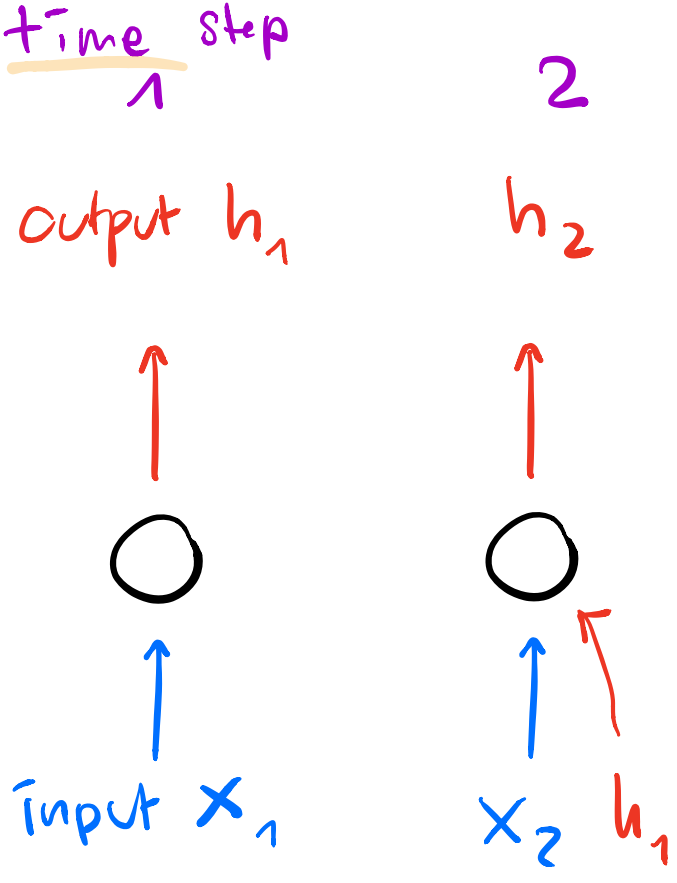
- Is **recurrent**: takes as input what is has computed before

→ time dimension!

Recurrence



non-recurrent



recurrent

Elman ("Vanilla") RNN

$$h_t = \text{RNN}(x_t, h_{t-1})$$

Elman ("Vanilla") RNN

$$h_t = \mathrm{RNN}(x_t, h_{t-1})$$

$$h_t = \sigma_h\left(\underbrace{W_h x_t + U_h h_{t-1}}_{[\vdots]} + b_h\right)$$

---

dimensions

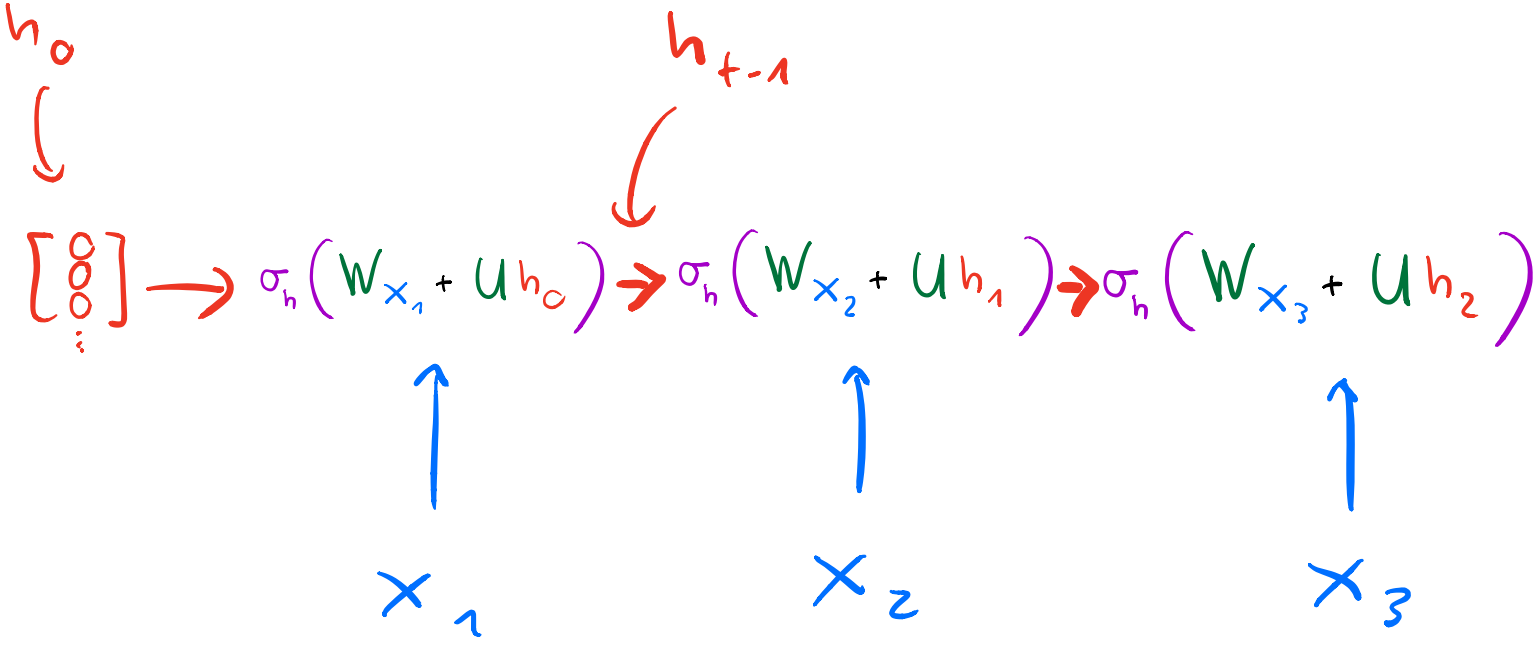
h	x	W_h	U_h	b_h
$\mathbb{R}^m$	$\mathbb{R}^n$	$m \times n$	$m \times m$	$\mathbb{R}^m$

How to train RNNs

- RNNs are trained with usual gradient-based optimization
- Loops are eliminated by **unrolling** the recurrent graph
- Backpropagation variant: **backpropagation through time** (BPTT)

Unfolding / unrolling RNN graph

Key observation: RNNs are just very deep FFNNs where layers **share weights**



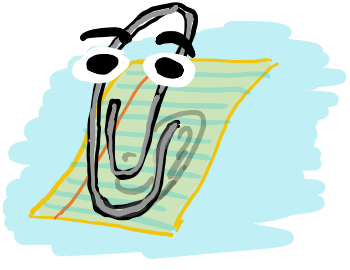
Problems with Elman RNNs

1. Gradient flow difficult because of depth:
   - Gradient can vanish

e.g. W_h used many times!

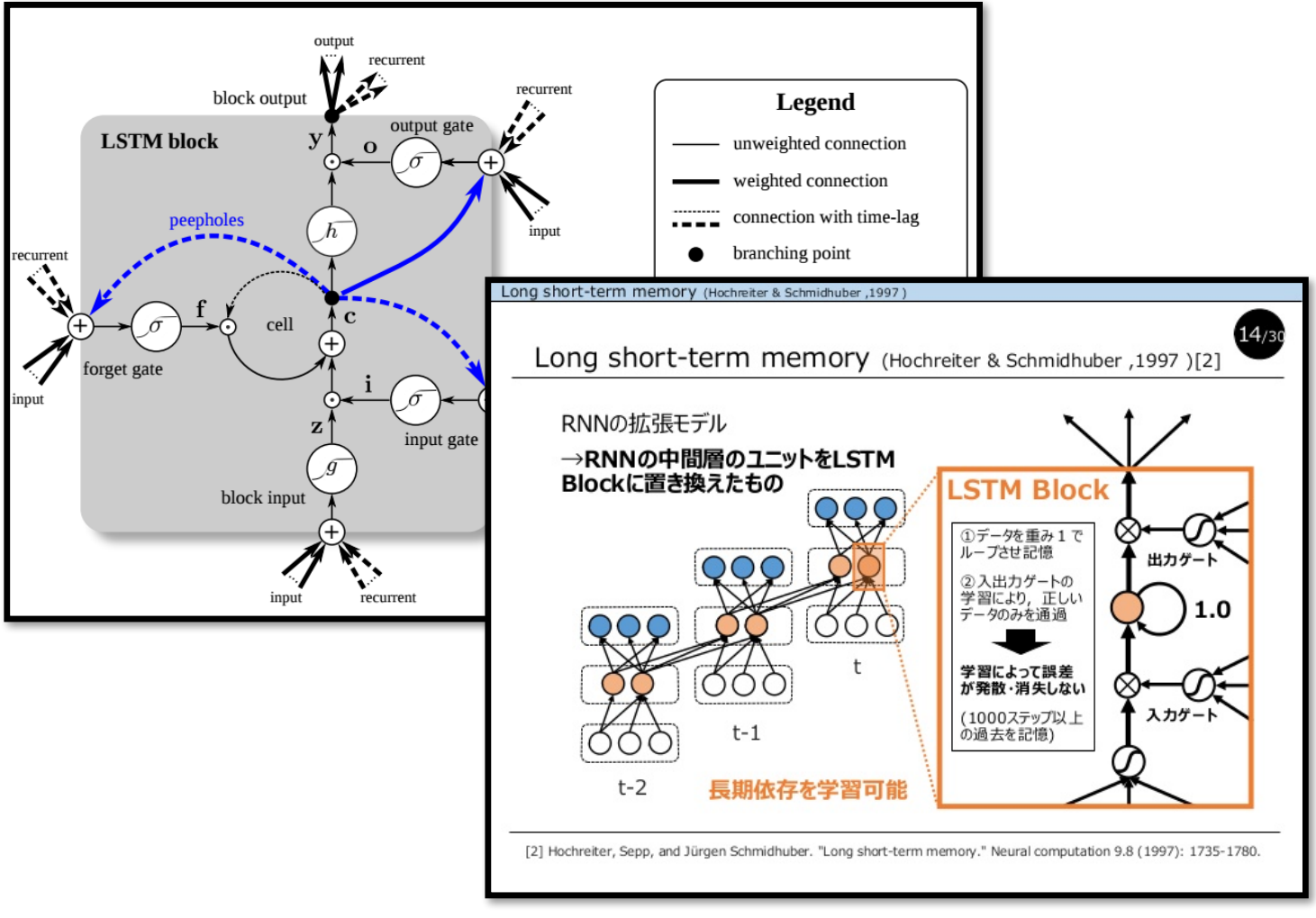
   - Gradient can explode

$$\begin{bmatrix} 7.1 \\ 6 \\ 3 \end{bmatrix} \xrightarrow{5!} \begin{bmatrix} 5 \\ 5 \\ 3 \end{bmatrix}$$



2. No built-in mechanisms to remember or forget

Long-short-term memory RNNs (LSTMs)



LSTMs

$$c_t, h_t = \text{LSTM}(x_t, c_{t-1}, h_{t-1})$$

- inputs: x_t
  c_{t-1} cell state
  h_{t-1} hidden state
- intermediate results (gates): gates = vectors

  i_t o_t f_t
- outputs: c_t cell state
  h_t hidden state

LSTMs

$\circ$ elementwise multiplication

gates

$$f_t = \sigma_g(W_f x_t + U_f h_{t-1} + b_f)$$
$$i_t = \sigma_g(W_i x_t + U_i h_{t-1} + b_i)$$
$$o_t = \sigma_g(W_o x_t + U_o h_{t-1} + b_o)$$

output

$$c_t = f_t \circ c_{t-1} + i_t \circ \sigma_c(W_c x_t + U_c h_{t-1} + b_c)$$
$$h_t = o_t \circ \sigma_h(c_t)$$

- $x_t \in \mathbb{R}^d$: input vector to the LSTM unit
- $f_t \in \mathbb{R}^h$: forget gate's activation vector
- $i_t \in \mathbb{R}^h$: input gate's activation vector
- $o_t \in \mathbb{R}^h$: output gate's activation vector
- $h_t \in \mathbb{R}^h$: hidden state vector also known as output vector of the LSTM unit
- $c_t \in \mathbb{R}^h$: cell state vector
- $W \in \mathbb{R}^{h \times d}$, $U \in \mathbb{R}^{h \times h}$ and $b \in \mathbb{R}^h$: weight matrices and bias vector parameters which need to be learned during training

where the superscripts d and h refer to the number of input features and number of hidden units, respectively.

LSTMs

~~Bill~~ gates:

f_t forget

i_t input

o_t output

$$c_t = f_t \circ c_{t-1} + i_t \circ \sigma\left(W_c x_t + U_c h_{t-1} + b_c\right)$$

$$h_t = o_t \circ \sigma_h(c_t)$$

Summary RNNs

- RNNs are special-purpose neural networks used for **sequence data**
- RNNs are **stateful**, turing-complete models
- The most successful and widely used variant of RNNs are **LSTMs**

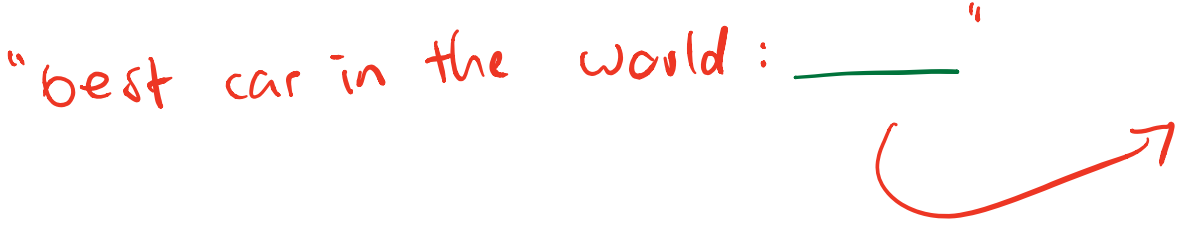
University of Zurich[UZH]

Institute of Computational Linguistics

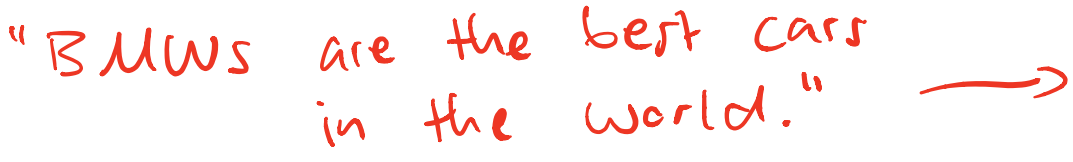
RNN Language Models

RNN Language Models

- Review: what language models do:
  - given some prefix text, predict the next word



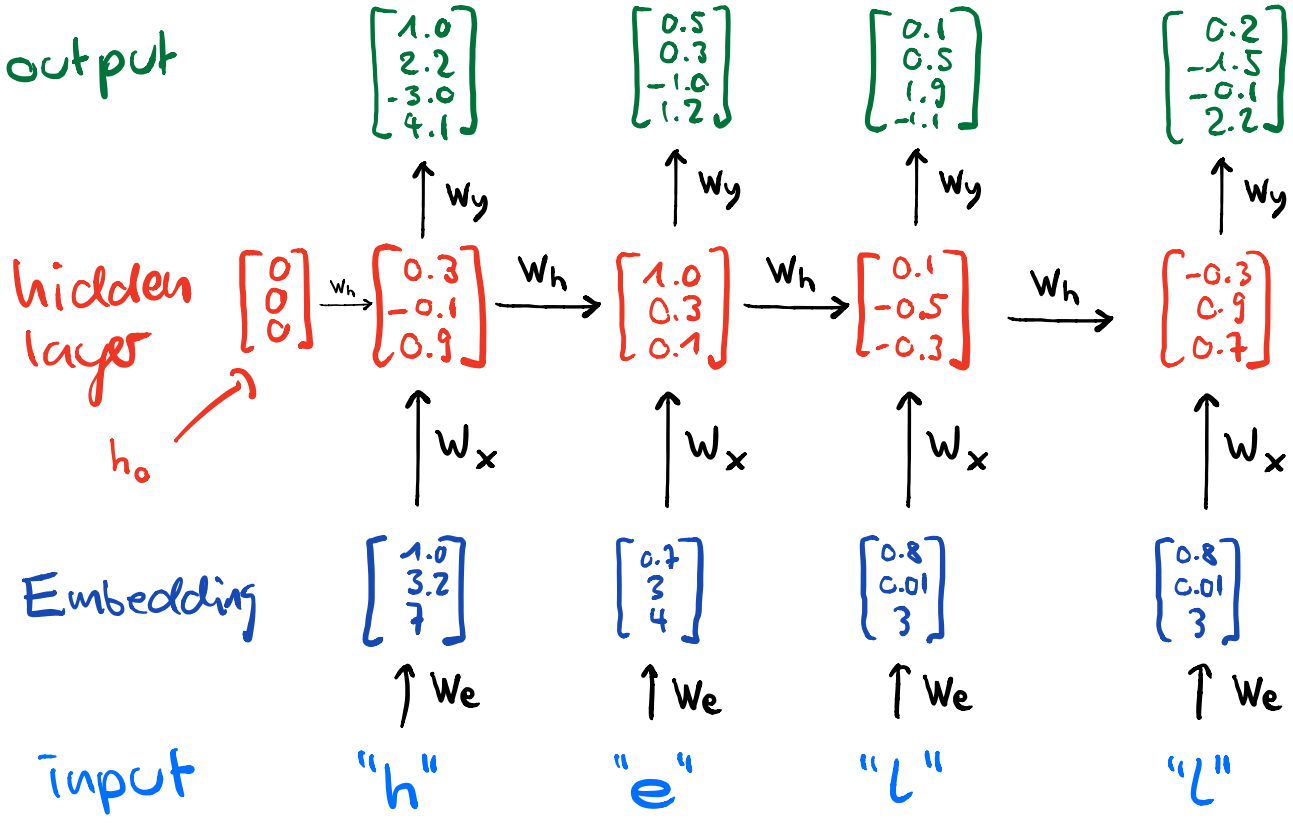
  - given some text, compute its probability



How to use an RNN as a language model

Vocab:	ID	one-hot
"h"	1	$\begin{bmatrix}1\\0\\0\\0\end{bmatrix}$
"e"	2	$\begin{bmatrix}0\\1\\0\\0\end{bmatrix}$
"l"	3	$\begin{bmatrix}0\\0\\1\\0\end{bmatrix}$
"o"	4	$\begin{bmatrix}0\\0\\0\\1\end{bmatrix}$

How to use an RNN as a language model



Summary

- **Word embeddings:** method to represent meaning of text as a vector of numbers

- **Recurrent neural networks**: kind of neural network that maintains state. Most popular variant: LSTMs

- RNNs can be used as **language models** - with infinite history!

Further Reading / Useful Links (again, lots!)

- A fun blog post about character RNNs: http://karpathy.github.io/2015/05/21/rnn-effectiveness/
- Chris Olah's blog: http://colah.github.io/posts/2015-08-Understanding-LSTMs/
- Stanford CS231n lecture about RNNs: https://www.youtube.com/watch?v=yCC09vCHzF8
- Skymind A.I. Wiki: https://skymind.ai/wiki/lstm
- Forward passes of different flavours of RNNs, both numpy and tensorflow: https://github.com/bricksdont/rnn-recipes
- Great visual explanations by Jay Alammar: http://jalammar.github.io/illustrated-word2vec/
- Chollet, F. (2017). Deep learning with Python. Manning Publications. **Kapitel 6**.
- Goldberg, Y. (2017). Neural network methods for natural language processing. Synthesis Lectures on Human Language Technologies, 10(1), 1-309. **Kapitel 14-17**.
- Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). Deep Learning. Cambridge: MIT Press. **Kapitel 10**.

Next Time