

VL Deep Learning for Natural Language Processing

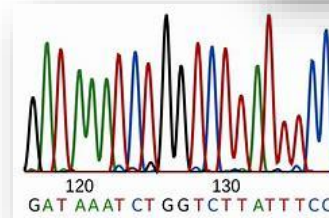
11. Language Models

Prof. Dr. Ralf Krestel
AG Information Profiling and Retrieval

Sequential Data



- So far: feed forward nets
 - Fully-connected layers
 - Every input data point independent
 - Input, e.g. a whole move review
 - No storing of states accross training samples
 - A sequence can only be processed as a whole not one-by-one
- Task with sequential data
 - Speech recognition
 - Music generation
 - Sentiment classification
 - DNA analysis
 - Machine translation
 - Scene description



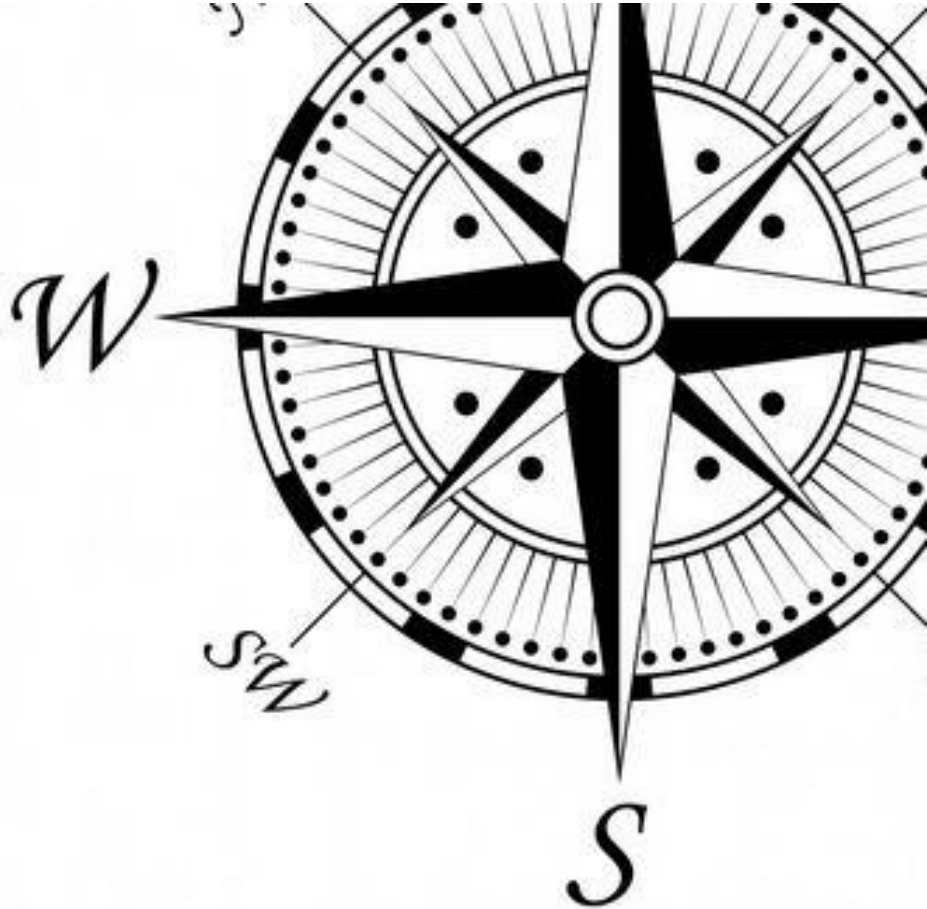
Learning Goals for this Chapter



- Understand the difference between feed-forward-nets and recurrent network architectures
 - Know application areas
-
- Relevant chapters:
 - P6.2
 - S5 (2021) <https://www.youtube.com/watch?v=PLryWeHPcBs>

Topics Today

1. **Language Models**
2. A RNN Language Model



Language Model Definition



- The goal of **language modeling** is to model a language, i.e., build a model of a language.
- With a good model you can make predictions:
 - „In five minutes, I will go _____“
 - ☐ home
 - ☐ Berlin
 - ☐ out of town
 - ☐ supermarket
- Formal: Given a sequence of words $x^{(1)}, x^{(2)}, \dots, x^{(t)}$, compute a probability distribution over the next word $x^{(t+1)}$: $P(x^{(t+1)} = w_j | x^{(t)}, \dots, x^{(1)})$
 - where w_j is a word from vocabulary $V = \{w_1, \dots, w_{|V|}\}$
- „Language modeling“ denotes the task.
- A system which solves this task is called **a language model**.

Examples



warum| Erweiterte Suche Sprachoptionen

warum **ist der himmel blau**
warum **liegt hier stroh**
warum
warum **ist die banane krumm**
warum **soll ein längeres starkes gefälle nicht mit getretener kupplung durchfahren werden**
warum **bin ich so fröhlich**
warum **kann das befahren dieser ungleichmäßig beleuchteten straße gefährlich werden**
warum **musste robert das bb haus verlassen**
warum **will thoas iphigenie heiraten**
warum **ist scharfes anfahren zu vermeiden**

→ I'll meet you at the →

☰ cafe airport office

1	2	3	4	5	6	7	8	9	0
q	w	e	r	t	y	u	i	o	p
@	#	&	*	-	+	=	()	
a	s	d	f	g	h	j	k	l	
↑	_	£	"	'	:	;	/		x
123	🔊							!/?	↩
,								.	😊

A Bad Language Model



„In five minutes, I will go _____“

- How to learn a language model?
 - By counting **n-grams**!
- An n-gram denotes consecutive words
 - n=1: unigrams: „In“, „five“, „minutes“, „I“, „will“, „go“
 - n=2: bigrams: „In five“, „five minutes“, „minutes I“, ...
 - n=3: trigrams: „In five minutes“, „five minutes I“, ...
 - n=4: fourgrams: „In five minutes I“, „five minutes I will“, ...
- Side note: There are also character n-grams
 - n=2: „_I“, „In“, „n “, „ f“, „fi“, „iv“, „ve“, „e “, ...

N-Gram Language Model II



- Assumption:

$$P(x^{(t+1)} = w_j | x^{(t)}, \dots, x^{(1)}) = P(x^{(t+1)} = w_j | x^{(t)}, \dots, x^{(t-n+2)})$$

- Conditional Probability:

$$P(x^{(t+1)} = w_j | x^{(t)}, \dots, x^{(t-n+2)}) = \frac{P(x^{(t+1)}, x^{(t)}, \dots, x^{(t-n+2)})}{P(x^{(t)}, \dots, x^{(t-n+2)})}$$

- Estimate the probabilities:
 - Large, representative corpus

$$P(x^{(t+1)} = w_j | x^{(t)}, \dots, x^{(1)}) \approx \frac{\text{count}(x^{(t+1)}, x^{(t)}, \dots, x^{(t-n+2)})}{\text{count}(x^{(t)}, \dots, x^{(t-n+2)})}$$

Example Four-Gram Language Model



„It is very cold outside, I will go _____“

$$P(w_j | i \text{ will go}) = \frac{\text{count}(i \text{ will go } w_j)}{\text{count}(i \text{ will go})}$$

- In the corpus:
 - „i will go“ occurs 1000 times.
 - „i will go home“ occurs 400 times.
 - $P(\text{home} | i \text{ will go}) = 0.4$
 - „I will go indoors“ occurs 10 times.
 - $P(\text{indoors} | i \text{ will go}) = 0.01$

- Problem:

- Context too small
 - But for any $n > 5$ too sparse
 - Memory need increases exponentially with n ($O(\exp(n))$)

Smoothing: E.g. Laplace

Backoff: E.g. Katz

Smoothing



- Maximum Likelihood Estimate (MLE)

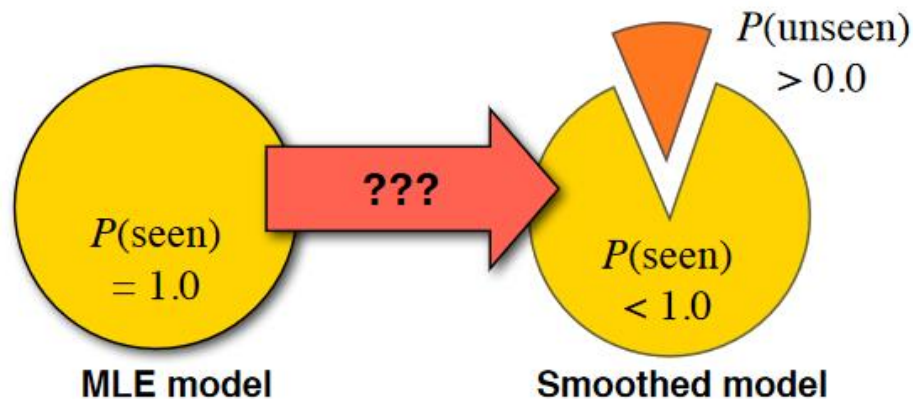
$$P(w_i) = \frac{C(w_i)}{\sum_j C(w_j)} = \frac{C(w_i)}{N}$$

- with N =count of all tokens
- $P(\text{seen})=1$

- Smoothing

- Assign some probability to unseen n-grams
- Laplace (add-1) smoothing

$$P(w_i) = \frac{C(w_i) + 1}{N + V}$$



Katz's Back-Off Model

- Idea
 - If count for n-gram is zero, take shorter n-gram instead
- Non-linear method
- The estimate for an n-gram is allowed to back off through progressively shorter histories.
- The most detailed model that can provide sufficiently reliable information about the current context is used.
- Trigram version (simplified):
 - if $C(w', w'', w) > 0$ $P^*(w | w', w'') = P(w | w', w'')$
 - else if $C(w'', w) > 0$ $P^*(w | w', w'') = P(w | w'')$
 - else if $C(w) > 0$ $P^*(w | w', w'') = P(w)$
 - else $P^*(w | w', w'') = 1 / \text{\#words}$

Katz's Back-Off Model Example



- Smoothing of Conditional Probabilities

$P(\text{Angeles} \mid \text{to}, \text{Los})$

- If „to Los Angeles“ is not in the training corpus, the smoothed probability $P(\text{Angeles} \mid \text{to}, \text{Los})$ is identical to $P(\text{York} \mid \text{to}, \text{Los})$.
- However, the actual probability is probably close to the bigram probability $P(\text{Angeles} \mid \text{Los})$.

Generative Language Model I



```
from nltk.corpus import reuters
from collections import Counter
counts = Counter(reuters.words())
total_count = len(reuters.words())
print counts.most_common(n=20)
# [(u'.', 94687), (u',', 72360), (u'the',
58251), (u'of', 35979), (u'to', 34035),
(u'in', 26478), (u'said', 25224), (u'and',
25043), (u'a', 23492), (u'mln', 18037),
(u'vs', 14120), (u'-', 13705), (u'for',
12785), (u'dlrs', 11730), (u'"', 11272),
(u'The', 10968), (u'000', 10277), (u'1',
9977), (u's', 9298), (u'pct', 9093)]
for word in counts:
    counts[word] /= float(total_count)
print sum(counts.values())
# 1.0
```

<https://nlpforhackers.io/language-models/>

```
import random
text = []
for _ in range(100):
    r = random.random()
    accumulator = .0
    for word, freq in counts.iteritems():
        accumulator += freq
        if accumulator >= r:
            text.append(word)
            break
print ' '.join(text)
# tax been its and industrial and vote "
decision rates elimination and 2 . base Ltd one
merger half three division trading it to company
before CES mln may to . . , and U is - exclusive
affiliate - biggest its Association [...]
from operator import mul
print reduce(mul, [counts[w] for w in text],
1.0)
# 3.0290546883e-32
```

Generate a text with 100 words
(=unigram language model)

Probability of generated text

Generative Language Model II



```
from nltk.corpus import reuters
from nltk import bigrams, trigrams
from collections import Counter, defaultdict
first_sentence = reuters.sents()[0]
print first_sentence # [u'ASIAN', u'EXPORTERS', u'FEAR', u'DAMAGE', u'FROM'...
print list(bigrams(first_sentence)) # [(u'ASIAN', u'EXPORTERS'), (u'EXPORTERS', u'FEAR'),...
print list(bigrams(first_sentence, pad_left=True, pad_right=True))
print list(trigrams(first_sentence, pad_left=True, pad_right=True))
model = defaultdict(lambda: defaultdict(lambda: 0))
for sentence in reuters.sents():
    for w1, w2, w3 in trigrams(sentence, pad_right=True, pad_left=True):
        model[(w1, w2)][w3] += 1
print model["what", "the"]["economists"] # "economists" follows "what the" 2 times
print model["what", "the"]["nonexistingword"] # 0 times
print model[None, None]["The"] # 8839 sentences start with "The"
for w1_w2 in model:
    total_count = float(sum(model[w1_w2].values()))
    for w3 in model[w1_w2]:
        model[w1_w2][w3] /= total_count
```



- What is the probability of the following sentence „*the weather is nice*“ under a bigram language model learnt from the corpus below?

the weather was bad yesterday
today the weather will be better
it is nice today

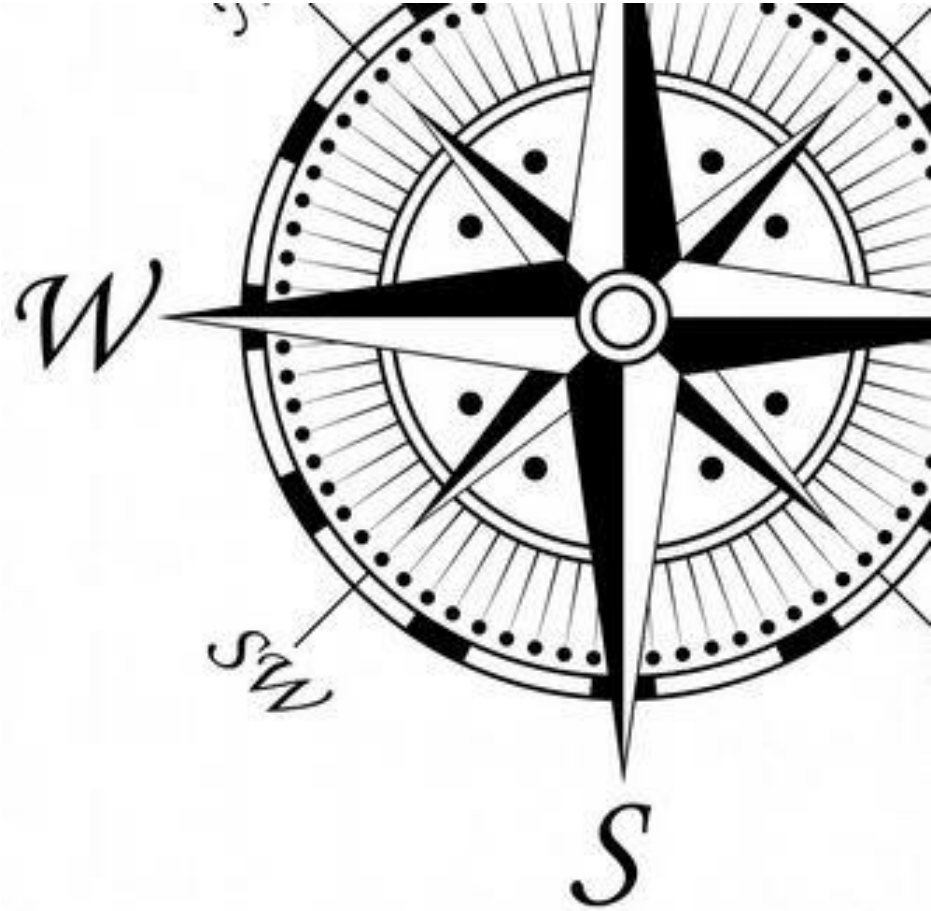
- Use Laplace smoothing!
- Laplace-smoothed bigrams:

$$P(w_n | w_{n-1}) = \frac{C(w_{n-1}w_n) + 1}{C(w_{n-1}) + V}$$



Topics Today

1. Language Models
2. **A RNN Language Model**



Neural Language Model With Fixed Window Size I



- Output=probability distribution

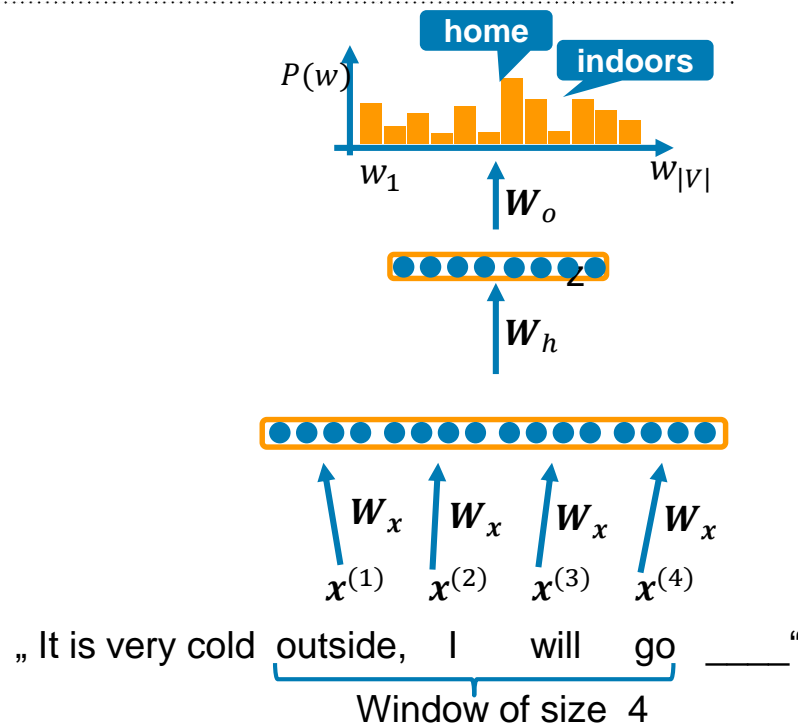
$$\hat{y} = \text{softmax}(\mathbf{W}_o \mathbf{h} + \mathbf{b}_o) \in \mathbb{R}^{|V|}$$

- Hidden layer

- $\mathbf{h} = f(\mathbf{W}_h \mathbf{e} + \mathbf{b}_h)$

- Concatenated word embeddings

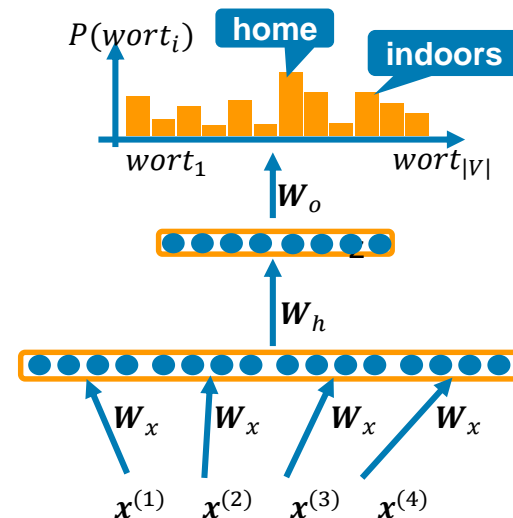
- $\mathbf{e} = [\mathbf{e}^{(1)} = \mathbf{W}_x \mathbf{x}^{(1)}; \mathbf{e}^{(2)} =$



Neural Language Model With Fixed Window Size II



- Advantage over n-gram language model
 - No sparsity problem
 - Model size is in $O(n)$ not $O(\exp(n))$
- Not yet solved:
 - Fixed window size too small
 - Increasing window size increases W_h
 - Window will never be large enough!
 - Weights are not shared among $x^{(i)}$
- We need a model that can process input sequences of different lengths.



RNN Language Model



- Probability distribution

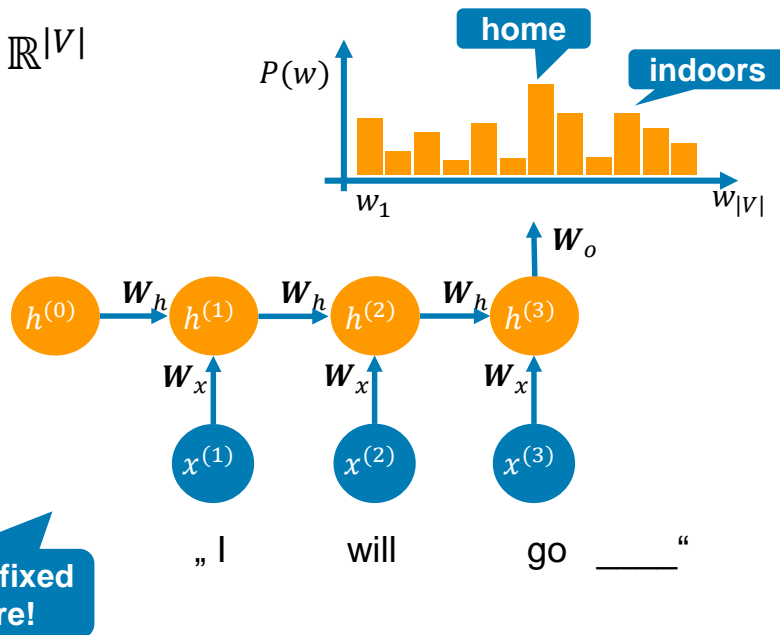
$$\hat{y} = \text{softmax}(\mathbf{W}_o \mathbf{h}^{(T)}) \in \mathbb{R}^{|V|}$$

- Hidden layer

$$\mathbf{h}^{(t)} = f(\mathbf{W}_h \mathbf{h}^{(t-1)} + \mathbf{W}_x \mathbf{x}^{(t)})$$

- Word embedding vectors

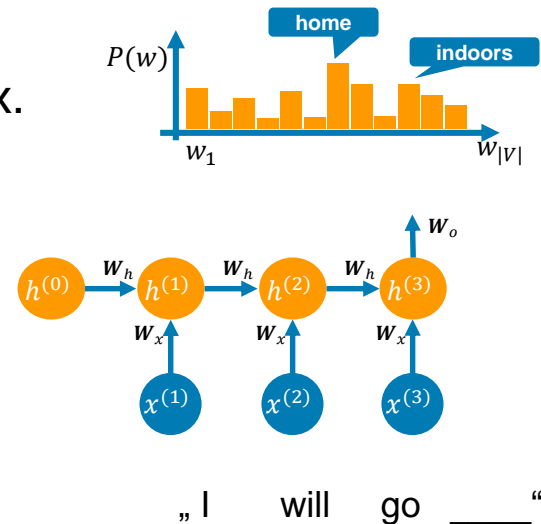
$$\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}, \mathbf{x}^{(4)}$$



RNN Language Model



- Advantage over fixed window
 - Input data is processed sequentially.
 - Input can be of variable length.
 - Weights are shared across time steps in a state matrix.
 - (Theoretically) access to information at time step t from many time steps before
- Disadvantages of RNNs
 - Computation accross many time steps very slow
 - In practice, it is hard to access old information



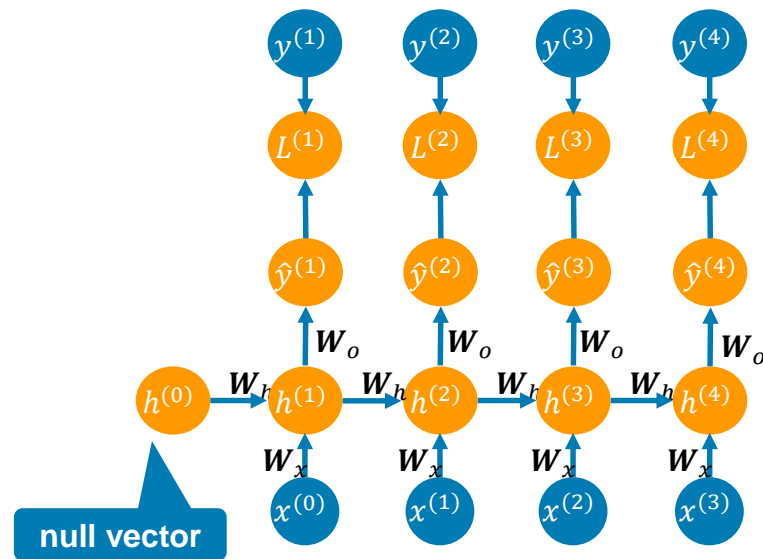
Learning the Weights

- Take a large corpus
 - Sequence of words $x^{(1)}, \dots, x^{(T)}$
- Compute for each word $x^{(t)}$ a probability distribution $\hat{y}^{(t)}$ given all previous words
- Loss function for step t is the cross entropy between predicted distribution $\hat{y}^{(t)}$ and actual next word $y^{(t)} = x^{(t+1)}$:

$$L^{(t)}(\theta) = CE(\hat{y}^{(t)}, y^{(t)}) = - \sum_{j=1}^{|V|} y_j^{(t)} \log \hat{y}_j^{(t)}$$

- Total loss is average:

$$L(\theta) = \frac{1}{T} \sum_{t=1}^T L^{(t)}(\theta)$$

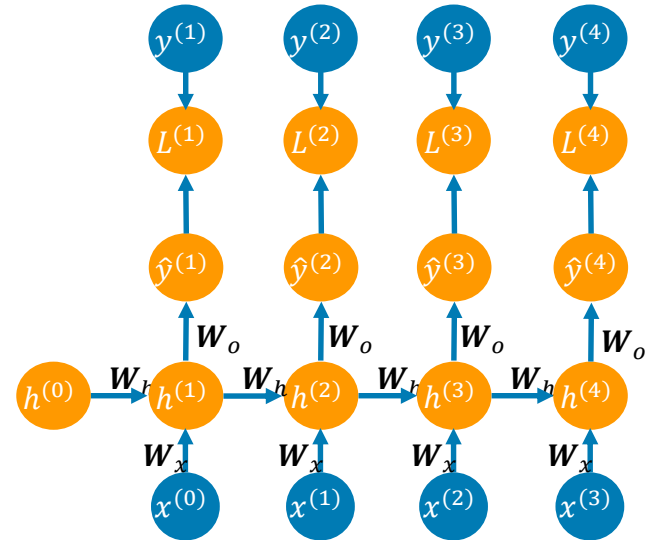


Train RNNs

- Computing the loss function and the gradients for the whole corpus is way too expensive!
- Stochastic gradient descent to the rescue
 - Update weights using small samples
- → computation per sentence
 - $\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(T)}$ is a sentence

$$L(\theta) = \frac{1}{T} \sum_{t=1}^T L^{(t)}(\theta)$$

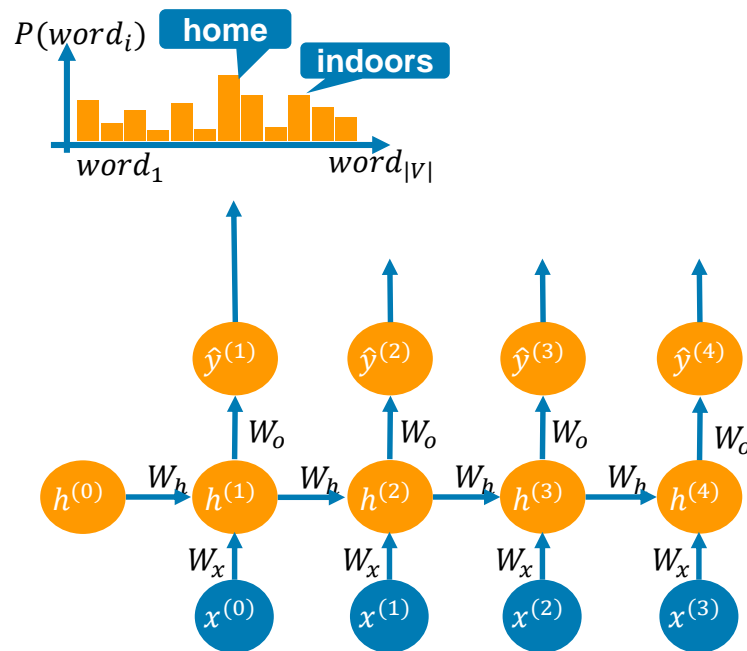
- Computation of $L(\theta)$ for one sentence:
 1. Computation of the gradients
 2. Update the weights
 3. Continue with next sentence



Generation of Sentences

- Analogous to n-gram language model
 - Repeated sampling of words
 - The sampled words in one step become the input for the next step
 - At some point there will be a `<eos>`-token sampled
 - In case a `<unk>`-token is sampled, ignore and sample again

`np.random.choice`



Examples I

- The kind of text that is generated depends on the training data



Good morning. And as we mark the fact that they can stand with their companies that are consistent to the state of Pakistan and the United States of America.

With the financial system we can do that. And the people of the United States will not be able to continue to support the people of the greatest problem of the American people to stay in the

White House. And that's why I've got to recognize the private sector that there is no doubt that we've got to continue to shape the painful realisation that we are the United States of America.

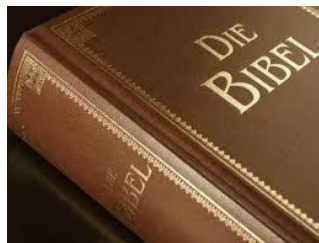
23:2 And the vision of the breaking thereof shall be in rubbick, and they shall take away the stones out of the land.

24:11 Thus saith the LORD of hosts; Ask now this stones are for the righteous and the children of Israel.

https://twitter.com/RNN_Bible

<https://www.avclub.com/a-bunch-of-comedy-writers-teamed-up-with-a-computer-to-1818633242>

<https://medium.com/@samim/obama-rnn-machine-generated-political-speeches-c8abd18a2ea0>



JERRY:

Well the elevator opens and wrong side of the door... I thought maybe the door's not waiting, but it said "going down" and Kramer couldn't help me move it. I just wanted to get out of it, just get out. (He slams his hand on the door.)

KRAMER enters dancing with garbage.

KRAMER:

Hey hey hey, great idea for a big sponge: Make it so large you think it's got a fat clock in the middle.

JERRY

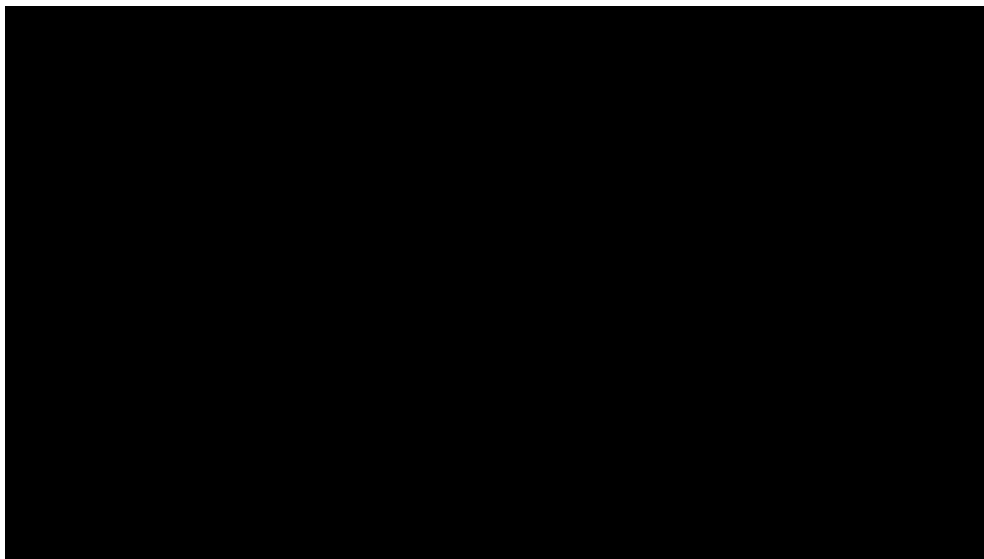
(takes off his bones)

Kramer, do you have a fun flashback to do?

Example II



- Movie scripts written by AI (with some human help, e.g., selection)



https://www.youtube.com/watch?v=5qPgG98_CQ8



Sunspring | A Sci-Fi Short Film Starring Thomas Middleditch

<https://www.youtube.com/watch?v=LY7x2lhqjmc>

Evaluation of Language Models

- Typically computation of **perplexity** on test set $W = w_1 \dots w_T$

$$PP(W) = P(w_1 \dots w_T)^{-\frac{1}{T}}$$

Normalization over number of words

$$PP(W) = \left(\prod_{i=1}^T \frac{1}{P(w_i | w_1 \dots w_{i-1})} \right)^{\frac{1}{T}}$$

– Lower is better!

- Or **log-likelihood**

$$\sum_{i=1}^T \log P(w_i | w_1 \dots w_{i-1})$$

– Higher is better!

Model	Perplexity
Interpolated Kneser-Ney 5-gram (Chelba et al., 2013)	67.6
RNN-1024 + MaxEnt 9-gram (Chelba et al., 2013)	51.3
RNN-2048 + BlackOut sampling (Ji et al., 2015)	68.3
Sparse Non-negative Matrix factorization (Shazeer et al., 2015)	52.9
LSTM-2048 (Jozefowicz et al., 2016)	43.7
2-layer LSTM-8192 (Jozefowicz et al., 2016)	30
Ours small (LSTM-2048)	43.9
Ours large (2-layer LSTM-2048)	39.8

One billion parameters

<https://research.fb.com/building-an-efficient-neural-language-model-over-a-billion-words/>

Generative Language Models



- Are automatically generated texts useful?
 - If so, in which context?
- Which are applications where language models can be used in a meaningful way?
 - (Except to generate text)
- Ethical, legal considerations?
 - Copyright, authorship, ...
 - Fake news, „truth“, ...



Learning Goals for this Chapter



- Understand the difference between feed-forward-nets and recurrent network architectures
 - Know application areas
-
- Relevant chapters:
 - P6.2
 - S5 (2021) <https://www.youtube.com/watch?v=PLryWeHPcBs>