Für $k = 2$ bekommen wir:

p(Er ölt) =

p(E|⟨s⟩,⟨s⟩) p(r|⟨s⟩,E) p(_ |E,r) p(ö|r,_) p(l|_,ö) p(t|ö,l) p(⟨s⟩|l,t)

$$p(w_i|w_{i-k}^{i-1}) = \frac{\max(0, f(w_{i-k}^{i}) - \delta_k)}{f(w_{i-k}^{i-1})} + \alpha(w_{i-k}^{i-1})\ p(w_i|w_{i-\underline{k+1}}^{i-1})$$

p(Buch | das, rote) = p* (Buch | das, rote) + α(das, rote) · p_backoff(Buch | ~~das,~~ rote)

p_backoff (Buch | rote) = (p* (Buch | rote) + α(rote) · p_backoff (Buch | ~~rote~~))
p_backoff (Buch) = p* (Buch | rote)
= f(Buch) / f() oder N

p*(Buch | das, rote) = (f (das, rote, Buch) - δ2) / f(das, rote)
p* (Buch | rote) = (f(rote, Buch) - δ 1) / f(rote)

....

$$p(w_i|w_{i-k}^{i-1}) = \frac{\max(0, f(w_{i-k}^{i}) - \delta_k)}{f(w_{i-k}^{i-1})} + \alpha(w_{i-k}^{i-1})\ p(w_i|w_{i-\underline{k+1}}^{i-1})$$

p(w3| w1,w2) = (f (w1, w2,w3) - δ2) / f(w1,w2)+ α(w1,w2) · p_backoff(w3 | _ w2)

3-gram-freq

('das', 'rote', 'buch')	:	5
('dieses', 'rote', 'buch')	:	2
('gute', 'rote', 'buch')	:	4
('das', 'gelbe', 'buch')	:	1
('das', 'rote', 'kleid')	:	2
('dieses', 'rote', 'kleid'	:	2
('das', 'rote', 'haus')	:	8

We read the training corpus and get this 3-gram freq.
Next we want to compute p_smoothed(ngram) for all ngrams.
These will be our trained parameters.

When we have a new text. We convert the text in to a seq of ngram and compute the product of p_smoothed(ngram).

Goal
3-gram-smoothed_prob

('das', 'rote', 'buch')	:	0.001
('dieses', 'rote', 'buch')	:	0.002
('gute', 'rote', 'buch')	:	0.0004
('das', 'gelbe', 'buch')	:	0.0231
('das', 'rote', 'kleid')	:	0.33
('dieses', 'rote', 'kleid'	:	0.002
('das', 'rote', 'haus')	:	0.0018

p*

$$p(w_i|w_{i-k}^{i-1}) = \frac{\max(0, f(w_{i-k}^{i}) - \delta_k)}{f(w_{i-k}^{i-1})} + \alpha(w_{i-k}^{i-1})\, p(w_i|w_{i-k+1}^{i-1})$$

p(w3| w1,w2) = **(f (w1, w2,w3) - δ2) / f(w1,w2)**+ α(w1,w2) · p_backoff(w3 | _ w2)

3-gram-freq

(**'das', 'rote'**, 'buch')	:	5
('dieses', 'rote', 'buch')	:	2
('gute', 'rote', 'buch')	:	4
('das', 'gelbe', 'buch')	:	1
(**'das', 'rote',** 'kleid')	:	2
('dieses', 'rote', 'kleid'	:	2
(**'das', 'rote'**, 'haus')	:	8

How to get f (w1, w2,w3)?
look up at the 3-gram-freq

How to get f(w1,w2)?

Folien 80

$$f(w_{i-k}^{i-1}) = \sum_w f(w_{i-k}^{i-1} w)$$

to get the freq of the context “das, rote” of a 3-gram, you look at the 3-gram-freq. Go through all 3-gram, and ask if does this 3-gram has the context “das, rote”?
It does not matter what word the 3-gram has. We only look at the context of it.

Then you take the freq of those 3-gram and sum them up.
This is how we compute the context of each 3-gram.
(do it for all 3 gram and store the context freq in a dict)

context_freq (3-gram)

'das', 'rote'	:	5+2+8
'dieses', 'rote'	:	2 +2
'gute', 'rote'	:	4
'das', 'gelbe'	:	1

how to compute the discount (3-gram /k = 2) ?

$$\delta = \frac{N_1}{N_1 + 2N_2}$$

3-gram-freq

('das', 'rote', 'buch')	:	5
('dieses', 'rote', 'buch')	:	**2**
('gute', 'rote', 'buch')	:	4
('das', 'gelbe', 'buch')	:	**1**
('das', 'rote', 'kleid')	:	**2**
('dieses', 'rote', 'kleid'	:	**2**
('das', 'rote', 'haus')	:	8

You look up in the 3-gram-freq again. …

Now we can compute p*(das rote Buch)

$$p(w_i|w_{i-k}^{i-1}) = \frac{\max(0, f(w_{i-k}^{i}) - \delta_k)}{f(w_{i-k}^{i-1})} + \alpha(w_{i-k}^{i-1})\, p(w_i|w_{i-k+1}^{i-1})$$

How can you compute the backoff of the context “das, rote”?

$$\alpha(C) = 1 - \sum_{w:f(C,w)>0} \frac{f(C,w) - \delta}{f(C)}$$

You need p*(3-gram) of all 3-gram.
We first will compute p* for all 3-grams from the traning corpus.

3-gram-freq

('das', 'rote', 'buch')	:	5
('dieses', 'rote', 'buch')	:	2
('gute', 'rote', 'buch')	:	4
('das', 'gelbe', 'buch')	:	1
('das', 'rote', 'kleid')	:	2
('dieses', 'rote', 'kleid'	:	2
('das', 'rote', 'haus')	:	8

now that we already have f(3gram), f(context), discount (3-gram), we can compute p*

for each 3-gram, we compute p*
and store it in a new dictionary.
In Übung4 solution, p* is “prob”.

p*

(**'das', 'rote'**, 'buch')	:	0.001
('dieses', 'rote', 'buch')	:	0.001
('gute', 'rote', 'buch')	:	0.005
('das', 'gelbe', 'buch')	:	0.006
(**'das', 'rote'**, 'kleid')	:	0.022
('dieses', 'rote', 'kleid'	:	0.003
('das', 'rote', 'haus')	:	0.042

Now we can compute p*.

We have all p*, now we can compute the backoff factor (context)
-we will compute the backoff factor for all context

C = “das rote”

p*

$$\alpha(C) = 1 - \sum_{w:f(C,w)>0} \frac{f(C,w)-\delta}{f(C)}$$

For each context, for example “das rote”, look in p* dict, for 3-gram that has this exact context.

Sum up p* of such 3-gram and subtract the result from 1

p*

(**'das', 'rote'**, 'buch')	:	0.001
('dieses', 'rote', 'buch')	:	0.001
('gute', 'rote', 'buch')	:	0.005
('das', 'gelbe', 'buch')	:	0.006
(**'das', 'rote'**, 'kleid')	:	0.022
('dieses', 'rote', 'kleid'	:	0.003
('das', 'rote', 'haus')	:	0.042

backoff(context 3-gram)

('das', 'rote')	:	1 - [p*(das, rote, buch)+ p*(das rote kleid)+ + p*(das rote haus)]
('dieses', 'rote')	:	1 - [p*(dieses, rote, buch)+ p*(dieses rote kleid)]
('gute', 'rote')	:	1 - [p*(gute, rote, buch)]
('das', 'gelbe')	:	1 - [p*(das, gelbe, buch)]

$$p(w_i|w_{i-k}^{i-1}) = \frac{\max(0, f(w_{i-k}^{i}) - \delta_k)}{f(w_{i-k}^{i-1})} + \alpha(w_{i-k}^{i-1})\, p(w_i|w_{i-k+1}^{i-1})$$

p(w3| w1,w2) = (f (w1, w2,w3) - δ2) / f(w1,w2)+ α(w1,w2) · **p_backoff(w3 | _ w2)**

p_backoff(w3 | _ w2) = f(w2,w3)- discount(k=1) / f(w2) + backoff(w2) * p(w3)

3-gram-freq

('das', **'rote', 'buch'**)	:	5
('dieses', **'rote', 'buch'**)	:	2
('gute', **'rote', 'buch'**)	:	4
('das', 'gelbe', 'buch')	:	1
('das', 'rote', 'kleid')	:	2
('dieses', 'rote', 'kleid'	:	2
('das', 'rote', 'haus')	:	8

How to compute f(w1,w2) ?

$$p_{backoff}(w|C) = \frac{f^*(C, w)}{\sum_{w'} f^*(C, w')}$$ Folie 87

We need a 2-gram freq which we will derive from the existing 3-gram freq
hier we have 2 options for how to compute the freq

For example, f(rote, buch)

$$f(C, w) = \sum_{w'} f(w', C, w)$$

$$f^*(C, w) = \sum_{w'} \mathbf{1}_{f(w', C, w) > 0}$$

0.

2-gram-freq (standard)

('rote', 'buch')	:	5+2+4
('gelbe', 'buch')	:	1
('rote', 'kleid')	:	2 +2
('rote', 'haus')	:	8

2-gram-freq (kneser-ney)

('rote', 'buch')	:	1+1+1
('gelbe', 'buch')	:	1
('rote', 'kleid')	:	1 +1
('rote', 'haus')	:	1

Now that we have the 2-gram freq, to compute f(rote buch), we can simply look up in the 2-gram freq dict.

Next, how can we compute the context freq f(rote) ?

$$p_{backoff}(w|C) = \frac{f^*(C, w)}{\sum_{w'} f^*(C, w')}$$

To compute f(rote), we look up in the 2-gram freq dict, and look for those with context rote. Sum the freq of them up. Below we will use the standard 2-gram-freq to compute this.

2-gram-freq (standard)
(**'rote'**, 'buch')   :   11
('gelbe', 'buch')   :   1
(**'rote'**, 'kleid')   :   4
(**'rote'**, 'haus')   :   8

2-gram context
rote : 11 + 4 +8
gelb : 1

Übung
- compute discount for the bigram (the same method as shown in 3-gram step)
-then compute p*(bigram)
-then compute backoff_factor(bigram_context)

Do the same for unigram
(To check your solution, run the file “Example_code_markov_model” in the tutorium website)

p*

backoff (context 2gram)