Sprachverstehen

Summer term 2016

Elmar Nöth
Lehrstuhl für Informatik 5
(Mustererkennung)

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Teil VII

Language Models with Markov Chains

Why do we need syntactic structure?

Wouldn’t the sentence
“I want to put a hyphen between the words Fish and And and And and Chips in my Fish–And–Chips sign”
have been clearer if quotation marks had been placed
before Fish, and between Fish and and, and and and And, and And and and, and and and And, and And and and, and and and Chips, as well as after Chips?

Yes, it makes sense

Why do we need syntactic structure?

John where James had had had had had had had had had had been correct

John, where James had had “had”, had had “had had”;
“had had” had been correct.

vs.

John, where James had had “had had”, had had “had”;
“had had” had been correct.

To Know What Was Said Without Listening

- utterance: “We must resolve some problems”
- we hear NOTHING!
- we count on 91243 sentences of the Wall Street Journal (WSJ) which words are at the beginning of a sentence, and how often
- the text contains 2265839 words; (47343 different)
- can we guess the correct utterance?
- the utterance is not contained in the data set

Which Words Start a Sentence? (7170)

Rank	Count	Word	Rank	Count	Word
1	14844	THE	11	881	THIS
2	3877	IN	12	849	I
3	3801	BUT	13	822	THEY
4	3307	MR.	14	820	AS
5	1776	HE	15	798	**WE**
6	1742	A	16	789	IF
7	1722	QUOTE	17	751	THAT
8	1631	IT	18	733	AT
9	1290	AND	19	622	SOME
10	1184	FOR	20	568	IT'S

Which Words follow a “we” in WSJ? 416!

Rank	Count	Word	Rank	Count	Word
1	295	HAVE	11	50	WOULD
2	245	ARE	12	42	BELIEVE
3	119	DON’T	13	39	KNOW
4	108	WERE	14	39	DO
5	101	WILL	15	37	CAN’T
6	95	CAN	16	36	COULD
7	70	HAD	17	34	DIDN’T
8	66	WANT	18	33	EXPECT
9	55	THINK	19	31	SHOULD
10	52	NEED	20	29	**MUST**

Which Words follow a “must” in WSJ? 247!

Rank	Count	Word	Rank	Count	Word
1	26	HAVE	11	6	COME
2	15	PAY	12	5	SHOW
3	15	MAKE	13	5	DECIDE
4	11	TAKE	14	5	CLEAR
5	11	ALSO	15	4	STILL
6	10	APPROVE	16	4	REMAIN
7	8	NOT	17	4	PROVIDE
8	8	MEET	18	4	KNOW
9	6	GO	19	4	GET
10	6	FIND	..	..	..
			109	1	**RESOLVE**

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Which Words follow a “resolve” in WSJ? 36!

Rank	Count	Word	Rank	Count	Word
1	18	THE	11	2	ANY
2	6	A	12	1	TODAY
3	5	TO	13	1	THROUGH
4	3	THEIR	14	1	THESE
5	3	ITS	15	1	TECHNICALLY
6	3	DISPUTES	16	1	TECHNICAL
7	3	CASES	17	1	**SOME**
8	2	THIS	18	1	SOCIAL
9	2	OUR	19	1	SEVERAL
10	2	DIFFERENCES	20	1	REMAINING

Which Words follow a “some” in WSJ? 1364!

Rank	Count	Word	Rank	Count	Word
1	667	OF	11	31	COMPANIES
2	152	ANALYSTS	12	28	TRADERS
3	58	TIME	13	24	INDUSTRY
4	57	PEOPLE	14	23	U.
5	56	CASES	15	23	OBSERVERS
6	49	INVESTORS	16	23	ARE
7	47	OTHER	17	21	KIND
8	36	POINT	18	21	IN
9	34	ONE	19	20	ECONOMISTS
10	31	TWO	20	20	BIG
			..	..	..
			112	5	**PROBLEMS**

N-Gram Language Models

- For the Bayes formula we need the prior probability

$$P(\boldsymbol{w}) = P(w_1, .., w_N)$$

= Probability that the sequence of words $w_1, .., w_N$ is uttered

- Expand $P(\boldsymbol{W})$ using the chain rule

$$P(\boldsymbol{w}) = P(w_1, .., w_N) = P(w_1) \cdot P(w_2|w_1) \cdot \ldots \cdot P(w_N|w_1, \ldots, w_{N-1})$$

- The probability of w_i depends on all words (= history) uttered before
- $P(w_i|w_1, \ldots, w_{i-1})$ cannot be estimated for large i
- Remedy: define equivalence classes for the preceding words

N-Gram Language Models

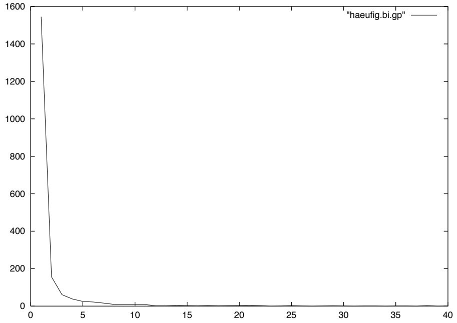
- Equivalence classes:
  w_i only depends on a shortened history $\Phi(w_1, ..., w_{i-1})$:
  $P(\boldsymbol{w}) = P(w_1) \cdot \prod_{i=2}^{N} P(w_i | \Phi(w_1, ..., w_{i-1})$
- Unigram language model: All histories are equivalent:
  $P(\boldsymbol{w}) = \prod_{i=1}^{N} P(w_i)$
- Bigram language model: all histories which end in the same word are equivalent:
  $P(\boldsymbol{w}) = P(w_1) \cdot \prod_{i=2}^{N} P(w_i | w_{i-1})$
- Trigram language model: all histories with the same last two words are equivalent:
  $P(\boldsymbol{w}) = P(w_1) \cdot P(w_2 | w_1) \cdot \prod_{i=1}^{N} P(w_i | w_{i-2}, w_{i-1})$

ML Estimation of N-Gram Language Models

- Compute the relative frequencies $f(w_i|w_{i-2}, w_{i-1})$, $f(w_i|w_{i-1})$, $f(w_i)$ from a training corpus
- e.g., $f(w_i|w_{i-2}, w_{i-1}) = \frac{\#(w_{i-2}, w_{i-1}, w_i)}{\#(w_{i-2}, w_{i-1})}$
  $\#$ stands for the number of occurrences in the training corpus
- ML estimate of $P(w_i|w_{i-n+1}, w_{i-1}) = f(w_i|w_{i-n+1}, w_{i-1})$
- Problem: Many trigrams do not occur in the training corpus $\rightarrow P(\boldsymbol{w}) = 0$
  $\Rightarrow$ recognition error
- Example: Size of vocabulary $= |\mathcal{V}| = 20000 \Rightarrow 8 \cdot 10^{12}$ trigrams are possible
- In a training corpus with 100000 words a maximum of 10^5 different n-grams can be observed, in reality significantly less
- Remedy: Smoothing of the estimates or using categories

Frequencies of Bigrams

The plot shows a histogram of absolute frequencies of bigrams in a sample of about 52000 words (1780 words in the vocabulary)



Smoothing of N-Grams

- Most simple possibility to prevent a situation where $\#(w_{i-n+1}, \ldots, w_i) = 0$ is the **Jeffrey smoothing**: Set $\#^{'}(w_{i-n+1}, \ldots, w_i) = 1 + \#(w_{i-n+1}, \ldots, w_i)$

$$P(w_i | w_{i-n+1}, \ldots, w_{i-1}) = \frac{\#^{'}(w_{i-n+1}, \ldots, w_i)}{\#^{'}(w_{i-n+1}, \ldots, w_{i-1})} = \frac{1 + \#(w_{i-n+1}, \ldots, w_i)}{|\mathcal{V}| + \sum_{w_i \in \mathcal{V}} \#(w_{i-n+1}, \ldots, w_i)}$$

- Problem: Impossible n-grams also receive a relatively high probability
- Example above: Even if — having 1780 words (Types) and 500 000 words running text (Tokens) — each bigram occurs exactly once in the training corpus, the probability mass is increased from about 500 000 to about 3 670 000, for 52 000 tokens from about 50 000 to 3 200 000.

Smoothing of N-Grams: Good-Turing Estimation

- Idea: All n-grams with the same frequency r in the training corpus should be assigned the same probability
- Good-Turing estimate assigns to each n-gram, which occurred r times in the training corpus, a value r^*

$$r^* = (r+1)\frac{n_{r+1}}{n_r}$$

- r^* is the expected value of the frequency for this n-gram in another corpus with the size of the training corpus
- n_l is the number of n-grams, which occurred exactly l in the training corpus
- Transformation into a probability: Probability that an n-gram occurs k-times in the data

$$p_k = \frac{r^*}{N} \quad \text{with } N = \text{number of n-grams in the training corpus}$$

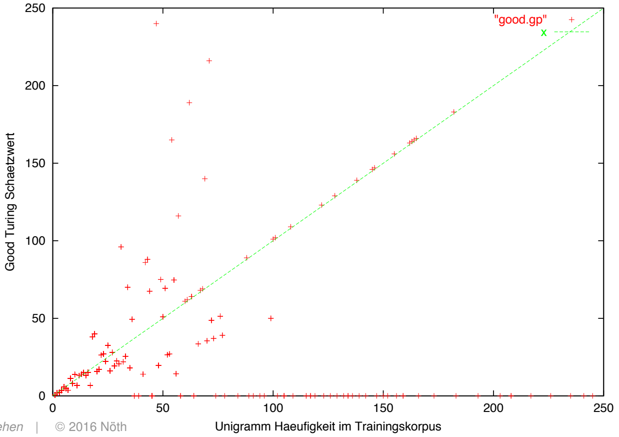
Smoothing of N-Grams: Good-Turing Estimation

- The Good-Turing estimation does not work if $n_r = 0$, i.e., before applying the Good-Turing smoothing, another smoothing has to be used such that $n_r \neq 0$ for all n-grams
- important: even for the new, colored frequency values r^* the sum of all frequencies is the same:

$$(N)^* = \sum_{k=1}^{K} (N_k)^* = \sum_{r=0}^{\infty} n_r r^* = \sum_{r=0}^{\infty} (r+1) n_{r+1} = \sum_{r=1}^{\infty} r n_r = N$$

- Thus — with the Good-Turing estimation — the frequencies are not changed arbitrarily, only **redistributed**

Smoothing of N-Grams: Good-Turing Estimation



Smoothing of N-Grams: Back-Off Strategies

- Idea: N-grams, which do not occur in the training corpus, are mapped onto a lower order n-gram (= with a shortened history)
- General Back-off strategy (= Back-off-Smoothing):

$$P_{smooth}(w \mid \boldsymbol{v}) = \begin{cases} \hat{q}(w \mid \boldsymbol{v}) & \text{if } \#(\boldsymbol{v}w) > 0 \\ \beta(\boldsymbol{v}) \cdot P_{smooth}(w \mid \boldsymbol{v}') & \text{if } \#(\boldsymbol{v}w) = 0 \end{cases}$$

- $\hat{q}(\cdot)$ colored estimation of the finer models
- **Redistribution** of the saved probability mass proportional to $P_{smooth}(w|\boldsymbol{v}')$
- $\boldsymbol{v}'$ is the **reduced** antecedent of the more coarse models
- The weight $\beta(\boldsymbol{v})$ guarantees the stochasticity of $P_{smooth}(w|\boldsymbol{v})$

Smoothing of N-Grams: Katz-Smoothing

- Combination of the Idea of Good-Turing Estimation with back-off strategy
- Approach:
  - The relative frequency of frequent n-grams with $r > k$ ($k = 5..8$) is not changed
  - The relative frequency of n-grams with $k \geq r > 0$ is lowered in favour of n-grams, which are not observed $r = 0$ (**discounting**)
  - Each n-gram with $r = 0$ receives some probability mass; if the corresponding $(n-1)$-gram is observed very often in the corpus, a higher probability is assigned, if it is observed rarely, a lower probability is assigned
- Katz-Smoothing:

$$P_{katz}(w \mid \boldsymbol{v}) = \begin{cases} \frac{\#(\boldsymbol{v},w)}{\#(\boldsymbol{v})} & \text{if } r > k \\ d_r \cdot \frac{\#(\boldsymbol{v},w)}{\#(\boldsymbol{v})} & \text{if } k \geq r > 0 \\ \alpha(\boldsymbol{v}) \cdot P(w \mid \boldsymbol{v}') & \text{if } r = 0 \end{cases}$$

Smoothing of N-Grams: Katz-Smoothing

- The exact formulae are the result of the following conditions:
  - The stochasticity has to be fulfilled
  - The factor d_r should be proportional to the Good-Turing estimation
  - The probability mass which is taken from the frequent n-grams should be equal to the probability mass which is assigned to the not observed
- There is a unique solution:

$$d_r = \frac{\frac{r^*}{r} - \frac{(k+1)n_{k+1}}{n_1}}{1 - \frac{(k+1)n_{k+1}}{n_1}} \quad \text{and} \quad \alpha(\boldsymbol{v}) = \frac{1 - \sum\limits_{w|\#(\boldsymbol{v},w)>0} P_{katz}(w|\boldsymbol{v})}{1 - \sum\limits_{w|\#(\boldsymbol{v},w)>0} P(w|\boldsymbol{v}')}$$

Smoothing of N-Grams: Interpolation

- Motivation: For small r the ML estimation is very imprecise, even if $r > 0$
- Approach: Inclusion of statistics of lower order even if $r > 0$
- linear interpolation:

$$P_I(w_n|w_1, .., w_{n-1}) =$$
$$\rho_o \cdot \frac{1}{|\mathcal{V}|} + \rho_1 \cdot f(w_n) + \rho_2 \cdot f(w_n|w_{n-1}) + \cdots + \rho_n \cdot f(w_n|w_1..w_{n-1})$$
$$\text{with } \sum_i \rho_i = 1$$

- The weights ρ_i can be decided upon with, e.g., the EM algorithm on a separate validation set

Other Important Interpolation methods

Not treated because of time constraints:

- Kneser-Ney interpolation: non-linear interpolation
- Interpolation using the principle of maximal entropy
- log-linear interpolation

Categorisation

- Another approach to prevent unreliable estimation values
- $L = 1000$ words $\Longrightarrow$ 1 000 000 000 trigrams!

$\rightarrow$ reduce the effective size of the vocabulary

- Groups of words with similar functional and statistical properties are merged into **categories**

$$\mathcal{C} = \{C_1, \ldots, C_N\} \quad \text{with} \quad \bigcup_{k=1}^{N} C_k = \mathcal{V}$$

- e.g. all numerals, names of cities, persons, and months each form a category
- Categories should be disjunct $\rightarrow$ categorisation is unique
  problem: e.g. is 'Essen' the name of a city or not?

$$w \in \mathcal{V} \longrightarrow C(w) \in \mathcal{C} \quad \text{with the property } w \in C(w)$$

Categorisation

- e.g. a categorical bigram language model is built as follows:
  $P(\boldsymbol{w}) \;=\; P(w_1) \cdot P(w_1 \mid C(w_1)) \cdot \prod_{i=2}^{m} P(w_i \mid C(w_i)) \cdot P(C(w_i) \mid C(w_{i-1}))$
- Necessary statistics:
  - The $N^2 - 1$ bigram probabilities $P(C_i)$, $P(C_j \mid C_i)$ for the transitions of categories
  - $|\mathcal{V}| - N$ probabilities to belong to a categories $P(W_k \mid C(W_k))$
- More complex: overlapping categories $\rightarrow$ category membership is a not observable state $\rightarrow$ HMM
- All interpolation and back-off strategies can be transferred to category systems

Cache Models

- Approach, to dynamically **adapt** a language model to the current subject
- A stochastic language model P_{static} is linearly interpolated with a cache model P_{cache}

$$P_{total}(w_i|w_{i-n+1},..,w_{i-1}) =$$
$$\rho_c P_{static}(w_i|w_{i-n+1},..,w_{i-1}) + (1-\rho_c)P_{cache}(w_i|w_{i-1})$$

- The cache model P_{cache} is estimated on the **last** spoken words
- Typically only a bi- or trigram, since there is only little data available for the cache model
- The interpolation weight ρ_c varies with the size of the Cache
- Significant improvement, especially with dictation systems

Perplexity

- The perplexity is a measure for the probabilities which are assigned to the sentences of the test set by the language model
- Given: A language model, that assigns a probability $P(\boldsymbol{W})$ to the sentences $\boldsymbol{W}$ of the test set
- Approximation of the entropy $H(\boldsymbol{W})$ of the model $P(w_i|w_{i-n+1}, .., w_{i-1})$ to the data $\boldsymbol{W}$:

  $$H(\boldsymbol{W}) = -\frac{1}{N_W} \log_2 P(\boldsymbol{W}) \quad N_W \text{ is the number of words in } \boldsymbol{W}$$
- The (empirical) test-set perplexity $PP(\boldsymbol{W})$ is defined as

  $$PP(\boldsymbol{W}) = 2^{H(\boldsymbol{W})}$$
- Can be estimated with $PP(\boldsymbol{W}) = P(\boldsymbol{W})^{-\frac{1}{N_W}}$
- The smaller the empirically approximated perplexity of the language model, the better the test set can be predicted and the better it is suited

Perplexity

- The real perplexity PP is always smaller than the empirical approximation $PP(\boldsymbol{W})$: $PP(\boldsymbol{W}) \geq PP$ (because of the Jensen Inequality)
- Real perplexity $PP = 2^H$ with

$$H = \lim_{m\to\infty} H_m = -\lim_{m\to\infty} \frac{1}{m} \sum_{\boldsymbol{w}\in\mathcal{V}^m} P(\boldsymbol{w}) \cdot \log_2 P(\boldsymbol{w})$$
$$= -\lim_{m\to\infty} \frac{1}{m} E[\log_2 P(\boldsymbol{w})] = -\lim_{m\to\infty} \frac{1}{m} \log_2 P(\boldsymbol{w})$$

- A random process has a maximal entropy $H = \log_2 L$ if $P(v|\boldsymbol{w}) = \frac{1}{L}$ with $L = |\mathcal{V}|$
- The perplexity can therefore also be seen as the average branching of the language
- The better the words of a language can be predicted the lower the perplexity

Perplexity

- The real perplexity of the English language is assumed to be about 30-50
- The empirical perplexity of a language model for trigrams is typically significantly smaller than for bigrams and unigrams
- The empirical perplexity of a language model strongly depends on the data:
  - For newspaper texts (Wall Street Journal) with a vocabulary of 5000 words a typical language model perplexity of about 128 (trigram) and 176 (bigram) is achieved
  - For spoken dialogue systems (e.g., ATIS– Air Travel Information, EVAR–Zugauskunft) the perplexity of the trigrams is under 20

Perplexity

- Because the perplexity is independent of the acoustic confusability of the words, the performance of a recogniser can be bad even with low perplexity
- There are approaches to, e.g., design the categories such that acoustically confusable words fall into different categories and can thus be better modeled by the language model

Limits of the Statistical Language Modeling

„Dann das Problem des Nach–, eh, des Alters der Kinder, des, eh, wenn sie, eh,Nachzugsalters, dann kommt aber fünftens und der sechste Punkt, gleichgeschlechtlich,nicht gleichgeschlechtlich, sondern, ob ich auch Asylgründe schaffe, eh, außerhalb der Verfolgung, als auch Gründe, wenn aus, wenn andere Gründe sozusagen, aus dem Geschlecht oder Ähnlichem, Frauen, die wegen ihres Frausein verfolgt werden.“

Stoiber with Ms. Christiansen, EN, 25.1.02