N-gram Model

Hiroshi Nakagawa

(Information Technology Center; Mathematical Informatics, Graduate School of Information Science and Technology; Graduate School of Interdisciplinary Information Studies, The University of Tokyo)

Statistical Model of String

- One-dimentional string $c_1\, c_2\, c_3, \ldots, c_n = C_1^n$
- One-dimentional word sequence $w_1\, w_2\, w_3, \ldots, w_n = w_1^n$
  - The description below refers to the word sequence.
- The occurrence probability of the word sequence w_1^n is represented in the following conditional probability model:

$$P(w_1^n) = P(w_1)P(w_2 \mid w_1)P(w_3 \mid w_1 w_2), \ldots, P(w_n \mid w_1^{n-1})$$

- The conditional probability of each member w_n consists a model for the string. When this probability depends on the most approximate N-word, the model is called "N-gram Model".

$$P(w_n \mid w_1^{n-1}) = P(w_n \mid w_{n-N+1}^{n-1})$$

Bag of Words Model

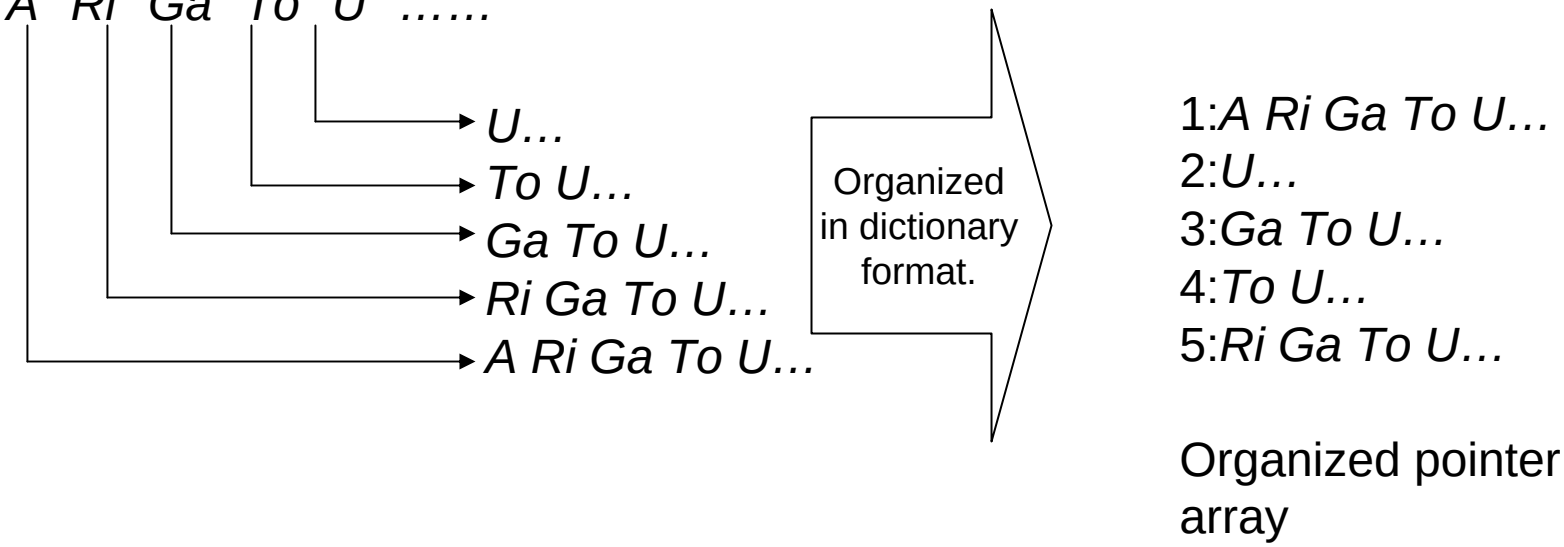
- In 1-gram model, $P(w_n)$ is the only one determiner for the linguistic model, which is the occurrence probability (in a corpus) of each word.
- A rough calculation though easier computation and less load.
- A basic model for information search.

N-gram

- An N-gram is a simple method (a linguistic model) to characterize languages.
- Certain linguistic units (phoneme, characteristics, word, POS, etc) are selected. "N-gram" is the Nth consecutive strings of the linguistic units represented in Markov Model. In particular, those explicitly included linguistic units are referred to as "linguistic unit name N-gram" (e.g. "word 2-gram").
- A single linguistic unit is called a "unigram", pairs are "bigrams", and triplets are "trigrams". (A model in which all words appear in the same frequency is called a "zero-gram".)
- Let’s find the total number of different words.
  - (1)Total number of character 2-grams in English
  - (2)Total number of mora 2-grams in Japanese
  - Mora time is one phonetic unit of one *hiragana* character. E.g. *n, ttu,* and *"–"* are one mora .
  - Syllable is "(consonant)(half vowel) vowel (mora phonemes)".
  - (3)Total number of character 2-grams in Japanese
  - (4)Total number of word 2-grams in Japanese
  - (5)Total number of POS 2-grams in Japanese

N-gram Analysis

- In many cases, the total number of N-grams observed in a language is too large to process. A language (and its subsets) is characterized based on the N-gram observed in real texts. It is, therefore, necessary to use an N-gram analysis in text.
- *A Ri Ga To U ……*



- An organized pointer array is called a "suffix array". Arrays with the same strings at the head come in conjunction or adjacent to each other.
- A statistical analysis is simply performed to find how many N-grams exist in its vicinity.

KWIC (Key Word In Context)

- KWIC is a list showing in what kind of context employ specific linguistic expressions in a corpus given.
- This list can be created by using the pointer arrays for text (found in N-gram analysis) listed by dictionary format.
- KWIC for N-grams on the slide "N-gram analysis" are as follows:

Precedent Context	**Key Word**	Antecedent Context
The total number of	**N-grams**	observed in a language is too large…
A language (and its subsets) is characterized based on the	**N-grams**	observed in real text…
It is, therefore, necessary to employ	**N-gram**	analysis in text…

- KWIC gives information regarding with which words and expressions these key words co-occur. The trend of co-occurrence becomes essential information of natural language processing.

N-gram Probability Model

- N-gram theory is a chain model based on N linguistic units. Various types such as characters, words, and POS can be used as linguistic units.
- Firstly, a chain of N linguistic units is $C(w_1 w_2 \ldots . w_n)$, *C* as a frequency of occurrence in the corpus given.
- This model should be able to predict the next possible character based on preceding *N* characters (an N-th order Markov process for some characters in the corpus). In general, N-th order Markov chain model is a probability process in which a current status is determined by preceding *N* characters.
- Therefore, it is… $p(w_i \mid w_{i-n} \ldots w_{i-1})$
- A conditional probability: $$p(w_i \mid w_{i-n} \ldots w_{i-1}) = \frac{C(w_{i-n} \ldots w_{i-1} w_i)}{C(w_{i-n} \ldots w_{i-1})}$$

Obtain occurrence probabilities in N-gram (vol.1)

- Maximum Likelihood Estimation

P as N - 1st order Markov process for characters:

$$p(w_n \mid w_1 \ldots w_{n-1}) = \frac{C(w_1 \ldots w_{n-1} w_n)}{C(w_1 \ldots w_{n-1})}$$

Approximate the occurrence probability of N-gram with C as relative frequency provided.

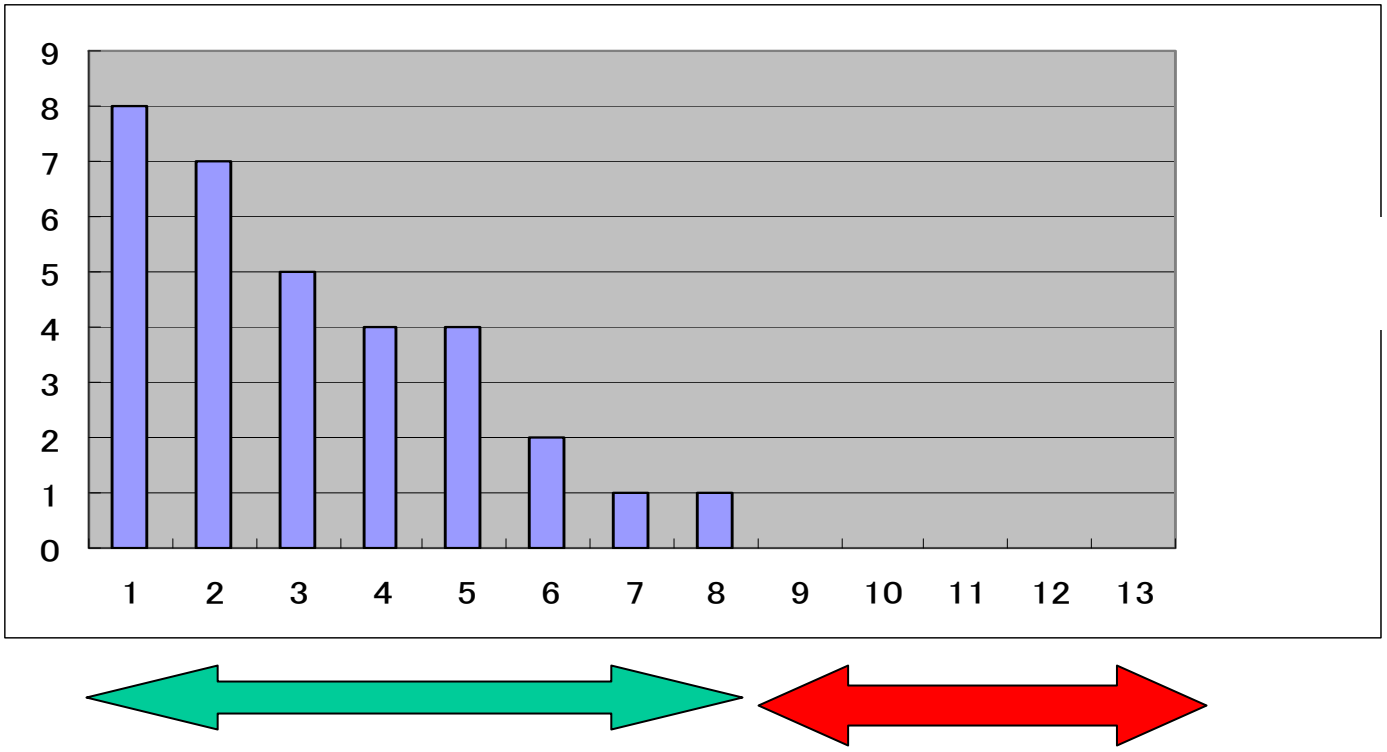
- A reliable N-gram cannot be obtained if N is too large.
- The relative frequency of given N-grams is 0 in many cases (the data sparseness issue).
- Addition: add any number to the numerator and the denominator.

$$P(w_n \mid w_1 \ldots w_{n-1}) = \frac{C(w_1 \ldots w_{n-1} w_n) + \delta}{N + V\delta} = \lambda \frac{C(w_1 \ldots w_{n-1} w_n)}{N} + (1-\lambda)\frac{1}{V}$$

$$where \quad \lambda = \frac{N}{N + \delta V}$$

Simply give δ when the numerator is 0. This is a handy method, but a low reliability. *V* is given as the total number of different words in the corpus.

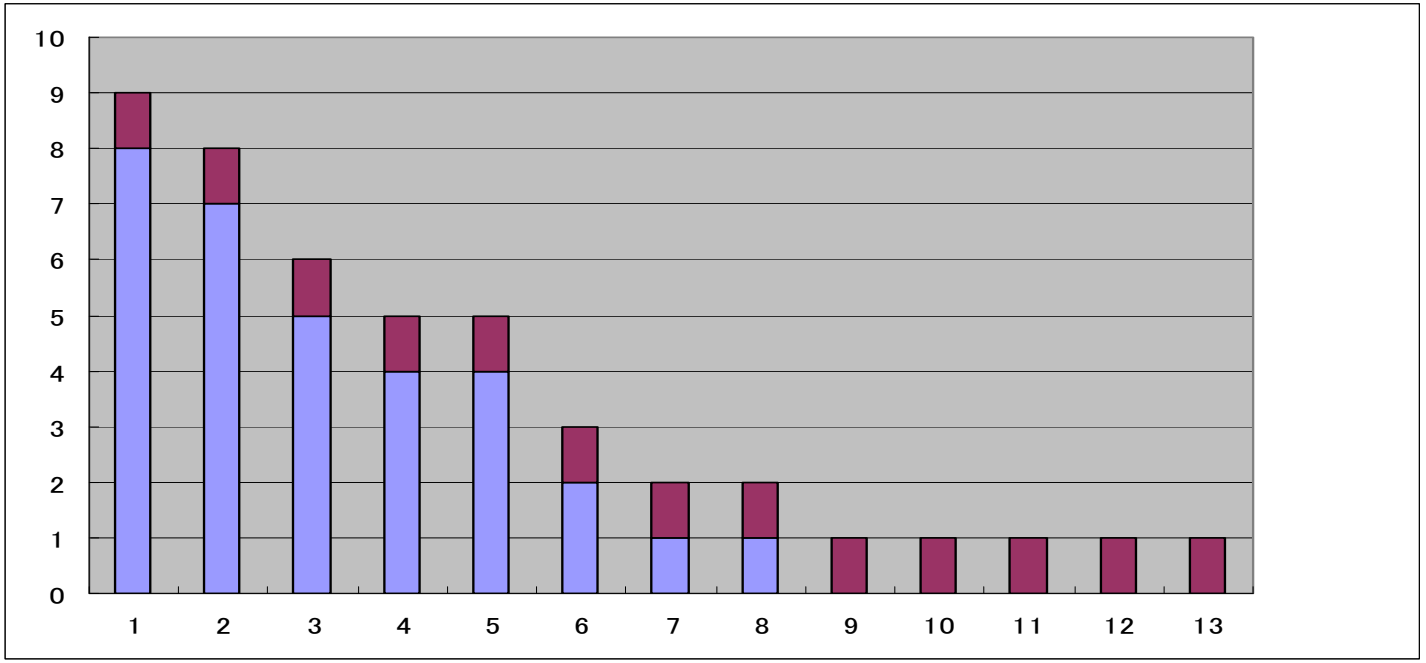
Back-off Smoothing (Frequency in Raw Data)

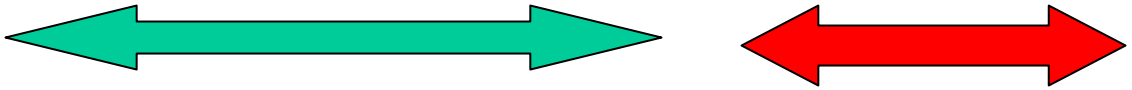


Words observed (8 words)

Words that are not observed yet, but will possibly appear (5 words)

Add δ (=1) to the frequency of each word



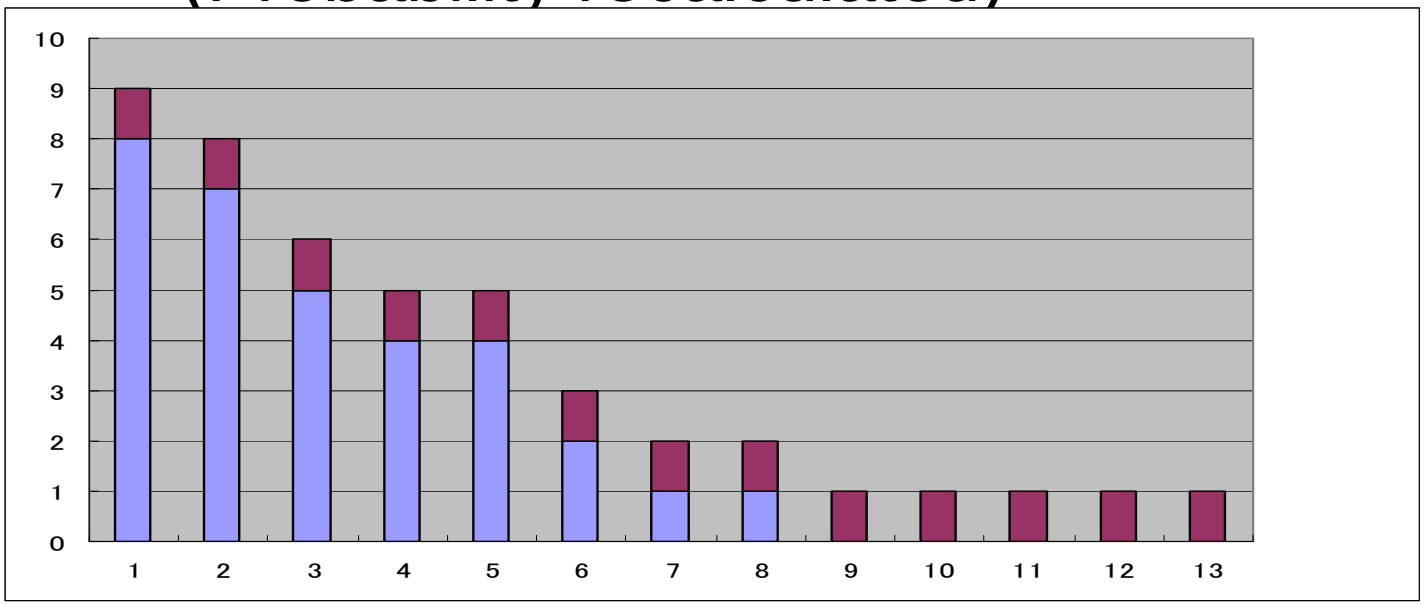


Words observed
(8 words)

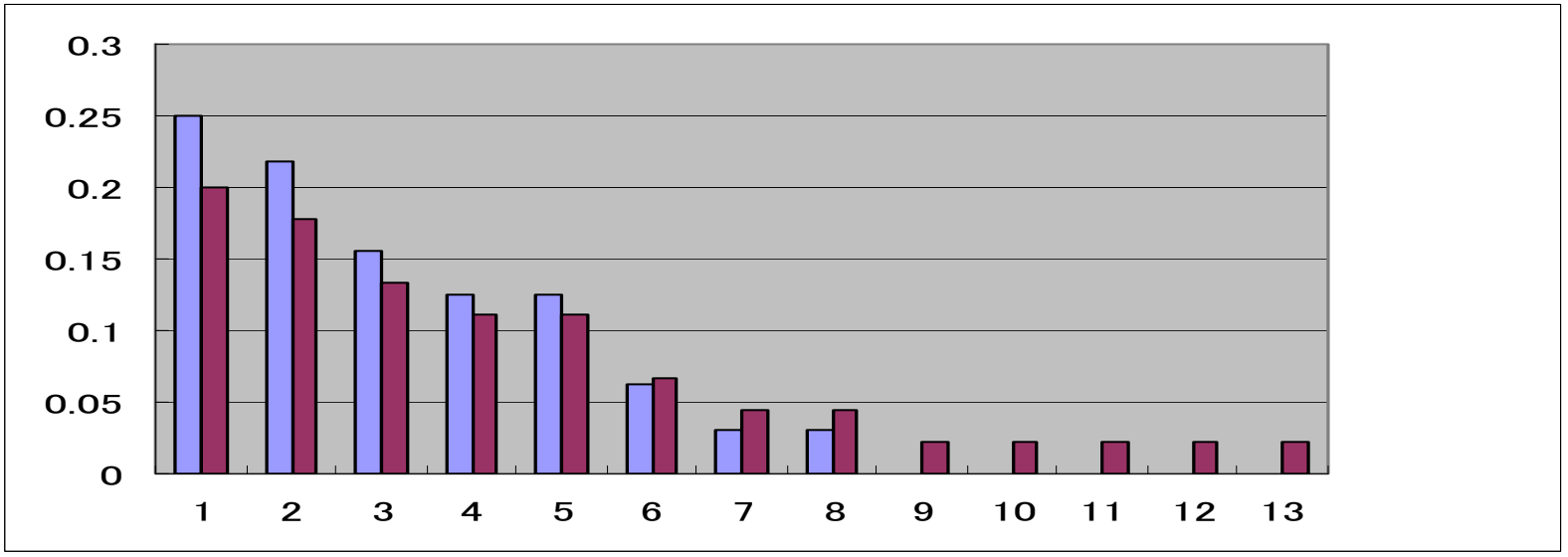
Words that are not observed yet, but will possibly appear (5 words)

Back-off Smoothing
(Probability recalculated)

Raw Data



Probability



Obtain occurrence probabilities in N-gram (vol.2)

Good-Turing Estimation

- Good-Turing probability estimation

  The total number of different words n_r in a corpus, the number of words N given:

$$N = \sum_{r>0} r n_r = n_1 + 2n_2 + 3n_3 + \ldots$$

  Good-Turing estimators use word w that appear r times in the corpus in the following equation to give you a probability:

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

- Good-Turing estimation

n_r is defined as the number of words that appear r times in a corpus including N words.

$$N = \sum_{r>0} r n_r = n_1 + 2n_2 + 3n_3 + \ldots$$

Good-Turing estimation will use the following equation to find the probability of word w that appear r times in the corpus:

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

- Probability expectation 0^* for words which occur at zero time:

$$0^* = \frac{n_1}{n_0} = \frac{n_1}{\text{(\# of words)} - \text{(\# of words found in the corpus)}}$$

- The total of relative frequencies of the words that appear more than once:

$$\sum_{r>0} \frac{n_r r^*}{N} = 1 - \frac{n_1}{N}$$

Thus, $\frac{n_1}{N}$ is the total estimated probability of frequencies for all words that do not appear in the corpus.

- $d = \frac{r^*}{r}$ is a discount coefficient.

Derivation in Good-Turing Estimation

- *M* as the number of different words in population.
- $P(w_i)$ as the occurrence probability of word w_i in population.
- $C(w_i)$ as the occurrence in the corpus of size *N* in which word w_i appear. Therefore, $\sum_{i=1}^{M} C(w_i) = N$
- When word *w* appears in the corpus *r* times, the expectation of occurrence and frequency in the population of *w* is:

$$E[P(w) \mid C(w) = r] = \sum_{i=1}^{M} P(w = w_i \mid C(w) = r) P(w_i) \qquad -(1)$$

$$r^* = E[r \mid C(w) = r] = E[P(w) \mid C(w) = r] N \qquad -(2)$$

- The probability distribution of a set of binary variables for words in a corpus of size *N* is:

$$P(w=w_i \mid C(w)=r)=\frac{P(C(w_i)=r)}{\sum_{i=1}^{M} P(C(w_i)=r)}$$

$$=\frac{{}_N C_r P(w_i)^r (1-P(w_i))^{N-r}}{\sum_{i=1}^{M} {}_N C_r P(w_i)^r (1-P(w_i))^{N-r}} \quad -(3)$$

Substitute in equation (1) with the previous result given:

$$E[P(w) \mid C(w)=r]=\frac{\sum_{i=1}^{M} {}_N C_r P(w_i)^{r+1} (1-P(w_i))^{N-r}}{\sum_{i=1}^{M} {}_N C_r P(w_i)^r (1-P(w_i))^{N-r}} \quad -(4)$$

The expectation of the total number of words observed *r* times in the corpus of size *N* is:

$$E_N[N_r] = \sum_{i=1}^{M} P(C(w_i) = r) = \sum_{i=1}^{M} {}_N C_r P(w_i)^r (1 - P(w_i))^{N-r}$$

Then, use ${}_N C_r = \frac{r+1}{N+1} {}_{N+1} C_{r+1}$ to rephrase the equation (4) as follows:

$$E[P(w) \mid C(w) = r] = \frac{r+1}{N+1} \frac{E_{N+1}(N_{r+1})}{E_N(N_r)} \qquad -(5)$$

Substitute in equation (2) to find r^*:

$$r^* = E[P(w) \mid C(w) = r]N = N \frac{r+1}{N+1} \frac{E_{N+1}(N_{r+1})}{E_N(N_r)} \qquad -(6)$$

Obtain a rough approximate solution of $E_N(N_r)$ and frequency N_r with a good size of *N* given:

$$r^* = (r+1) \frac{N_{r+1}}{N_r}$$

Obtain occurrence probabilities in N-gram (vol.4)
(Back-off Smoothing)

- A frequency estimation of bigrams with a frequency (= 0) based on Good-Turing estimation is:

$$p(w_2|w_1)=\frac{C^*(w_1,w_2)}{C(w_1)}=\frac{C^*(w_1,w_2)}{C\ (w_1,w_2)}\times\frac{C(w_1,w_2)}{C(w_1)}=d_{C(w_1,w_2)}\frac{C(w_1,w_2)}{C(w_1)}$$

- According to the equation, the total number of bigram probabilities calculated from the corpus is less than one.

$$\beta(w_1)=1-\sum_{w_2:C(w1,w2)>0} p(w_2|w_1)$$

- $\beta(W_1)$ is the total number of conditional probabilities for all W_2 that satisfy $C(W_1, W_2) = 0$.

- Substitute the result in the equation to find $P(W_1 | W_2)$ for word sequences that meet $C(W_1, W_2) = 0$.

$$p(w_2 \mid w_1)=\frac{\beta(w_1)p(w_2)}{\sum_{w:C(w1,w_2)=0} p(w_2)}=\frac{1-\sum_{w_2:C(w1,w_2)>0} p(w_2 \mid w_1)}{1-\sum_{w_2:C(w1,w)>0} p(w_2)}\times p(w_2)$$

Expansions of N-gram (vol.1): Class-based Models

- An N-gram class language model induces word classes used to predict the next word, such as POS. A POS-word class-based bigram model in a first-order Markov model given the provided word *w* and POS *c* is:

$$p(w_n \mid w_{n-1}) = p(w_n \mid c_n)p(c_n \mid c_{n-1})p(c_{n-1} \mid w_{n-1})$$
$$= p(w_n \mid c_n)p(c_n \mid c_{n-1})$$
$$\Theta\ p(c_{n-1} \mid w_{n-1}) = 1$$

C_i is the class to which word W_i belongs (e.x. POS)

In general, words have multiple POS. Thus, the following equation is introduced:

$$p(w_n \mid w_{n-1}) = \sum_{c_n} p(w_n \mid c_n)p(c_n \mid c_{n-1})$$

The total number of different POS is far smaller than the total number of different words when POS are used as class categories. Thus, N-gram can be processed with a large *N* provided.