Математические модели в морфологии

Энграммные языковые модели.

Алексей Андреевич Сорокин

ОТИПЛ МГУ,
осенний семестр 2017–2018 учебного года
10 октября 2017 г.

Вероятность текста

- Часто требуется оценивать вероятность текста:
- Автоматическая генерация текста (чатботы),

Вероятность текста

- Часто требуется оценивать вероятность текста:
- Автоматическая генерация текста (чатботы),
- Модель канала связи: $\widehat{t} = \operatorname{argmax} p(t|s) = \operatorname{argmax} p(s|t)p(t)$.
  - Например, s — предложение на исходном языке, t — на целевом.

Вероятность текста

- Часто требуется оценивать вероятность текста:
- Автоматическая генерация текста (чатботы),
- Модель канала связи: $\widehat{t} = \operatorname{argmax} p(t|s) = \operatorname{argmax} p(s|t)p(t)$.
  - Например, s — предложение на исходном языке, t — на целевом.
  - Или s — последовательность графем, t — последовательность фонем.

Вероятность текста

- Часто требуется оценивать вероятность текста:
- Автоматическая генерация текста (чатботы),
- Модель канала связи: $\widehat{t} = \operatorname{argmax} p(t|s) = \operatorname{argmax} p(s|t)p(t)$.
  - Например, s — предложение на исходном языке, t — на целевом.
  - Или s — последовательность графем, t — последовательность фонем.
  - s — последовательность слов, t — последовательность морфологических меток
- Везде нужно считать $p(t)$.

Вероятность текста

- Часто требуется оценивать вероятность текста:
- Автоматическая генерация текста (чатботы),
- Модель канала связи: $\widehat{t} = \operatorname{argmax} p(t|s) = \operatorname{argmax} p(s|t)p(t)$.
  - Например, s — предложение на исходном языке, t — на целевом.
  - Или s — последовательность графем, t — последовательность фонем.
  - s — последовательность слов, t — последовательность морфологических меток
- Везде нужно считать $p(t)$.
- Формула условной вероятности:
  $p(w_1 \dots w_N) = p(w_1)p(w_2|w_1)p(w_3|w_1 w_2) \dots p(w_N|w_1 \dots w_{N-1})$.

Базовая энграммная модель

- Предположение энграммной модели: каждое слово зависит только от $n-1$ предыдущего.

$$p(w_N|w_1 \dots w_{N-1}) = p(w_N|w_{N-n+1} \dots w_{N-1})$$

Базовая энграммная модель

- Предположение энграммной модели: каждое слово зависит только от $n-1$ предыдущего.

$$p(w_N|w_1 \dots w_{N-1}) = p(w_N|w_{N-n+1} \dots w_{N-1})$$

- Чаще всего берут $n \leqslant 3$ ($n = 1$ — униграммы, $n = 2$ — биграммы, $n = 3$ — триграммы).
- Как считать энграммные вероятности?

Базовая энграммная модель

- Предположение энграммной модели: каждое слово зависит только от $n-1$ предыдущего.

$$p(w_N|w_1 \dots w_{N-1}) = p(w_N|w_{N-n+1} \dots w_{N-1})$$

- Чаще всего берут $n \leqslant 3$ ($n = 1$ — униграммы, $n = 2$ — биграммы, $n = 3$ — триграммы).
- Как считать энграммные вероятности?
- Наивный подход: $p(w_n|\mathbf{w}_{1,n-1}) = \dfrac{c(\mathbf{w}_{1,n})}{c(\mathbf{w}_{1,n-1})}$.
- Здесь и далее $\mathbf{w}_{1,n} = w_1 \dots w_n$.

Базовая энграммная модель

- Предположение энграммной модели: каждое слово зависит только от $n-1$ предыдущего.

$$p(w_N|w_1 \dots w_{N-1}) = p(w_N|w_{N-n+1} \dots w_{N-1})$$

- Чаще всего берут $n \leqslant 3$ ($n = 1$ — униграммы, $n = 2$ — биграммы, $n = 3$ — триграммы).
- Как считать энграммные вероятности?
- Наивный подход: $p(w_n|\mathbf{w}_{1,n-1}) = \dfrac{c(\mathbf{w}_{1,n})}{c(\mathbf{w}_{1,n-1})}$.
- Здесь и далее $\mathbf{w}_{1,n} = w_1 \dots w_n$.
- Недостаток: нулевые вероятности.

Пример

я читал		1864			
я читал	книгу	19	$\frac{19}{1864}$	$\approx$	0.010
я читал	газету	3	$\frac{3}{1864}$	$\approx$	0.002
я читал	лекцию	11	$\frac{11}{1864}$	$\approx$	0.006
я читал	доклад	0	$\frac{0}{1864}$	$=$	0?
я читал	инструкцию	0	$\frac{0}{1864}$	$=$	0?

Аддитивное сглаживание

- Можно применить аддитивное сглаживание:

$$p(t_n|t_1 \dots t_{n-1}) = \frac{c(t_1 \dots t_{n-1} t_n) + \alpha}{c(t_1 \dots t_{n-1} \bullet) + \alpha|D|},$$

где D — словарь (множество возможных униграмм),
$\alpha > 0$ — сглаживающее слагаемое

Аддитивное сглаживание

- Можно применить аддитивное сглаживание:

$$p(t_n|t_1 \dots t_{n-1}) = \frac{c(t_1 \dots t_{n-1} t_n) + \alpha}{c(t_1 \dots t_{n-1} \bullet) + \alpha|D|},$$

  где D — словарь (множество возможных униграмм),
  $\alpha > 0$ — сглаживающее слагаемое
- Теперь уже нет нулевых вероятностей. Но как выбирать значение α?

Аддитивное сглаживание

- Можно применить аддитивное сглаживание:

$$p(t_n|t_1 \dots t_{n-1}) = \frac{c(t_1 \dots t_{n-1} t_n) + \alpha}{c(t_1 \dots t_{n-1} \bullet) + \alpha|D|},$$

  где D — словарь (множество возможных униграмм),
  $\alpha > 0$ — сглаживающее слагаемое
- Теперь уже нет нулевых вероятностей. Но как выбирать значение α?
- Маленькая α — риск переподгонки под обучающую выборку.
- Большая α — не учитываем наблюдаемые вероятности.

Аддитивное сглаживание

- Можно применить аддитивное сглаживание:

$$p(t_n|t_1 \dots t_{n-1}) = \frac{c(t_1 \dots t_{n-1} t_n) + \alpha}{c(t_1 \dots t_{n-1} \bullet) + \alpha|D|},$$

  где D — словарь (множество возможных униграмм),
  $\alpha > 0$ — сглаживающее слагаемое
- Теперь уже нет нулевых вероятностей. Но как выбирать значение α?
- Маленькая α — риск переподгонки под обучающую выборку.
- Большая α — не учитываем наблюдаемые вероятности.
- Нужно что-то среднее, на чём достигается наилучшее качество.
- Кстати, а как мерить качество?

Перплексия

- Алгоритм определения качества языковой модели:
  - Разбить данные на обучающую и контрольную выборку.

Перплексия

- Алгоритм определения качества языковой модели:
  - Разбить данные на обучающую и контрольную выборку.
  - Построить модель по обучающей выборке.

Перплексия

- Алгоритм определения качества языковой модели:
  - Разбить данные на обучающую и контрольную выборку.
  - Построить модель по обучающей выборке.
  - Посчитать перплексию на контрольной.

Перплексия

- Алгоритм определения качества языковой модели:
  - Разбить данные на обучающую и контрольную выборку.
  - Построить модель по обучающей выборке.
  - Посчитать перплексию на контрольной.
- Перплексия модели M на выборке $W = [w_1, \ldots, w_N]$:

$$PP_M(W) = p(w_1 \ldots w_N)^{-\frac{1}{N}}$$

Перплексия

- Алгоритм определения качества языковой модели:
  - Разбить данные на обучающую и контрольную выборку.
  - Построить модель по обучающей выборке.
  - Посчитать перплексию на контрольной.
- Перплексия модели M на выборке $W = [w_1, \ldots, w_N]$:

$$PP_M(W) = p(w_1 \ldots w_N)^{-\frac{1}{N}}$$

- Для униграммной модели

$$PP_M(W) = p(w_1 \ldots w_N)^{-\frac{1}{N}} = (\prod_i^N p(w_i))^{-\frac{1}{N}} = (\prod_{j=1}^{|D|} p(w_{(j)})^{n_j})^{-\frac{1}{N}}$$

$$\log_2 PP_M(W) = -\frac{1}{N} \sum_{j=1} n_j \log_2 p(w_{(j)})$$

$w_{(j)}$ — j-ое слово в словаре

Перплексия

- Алгоритм определения качества языковой модели:
  - Разбить данные на обучающую и контрольную выборку.
  - Построить модель по обучающей выборке.
  - Посчитать перплексию на контрольной.
- Перплексия модели M на выборке $W = [w_1, \ldots, w_N]$:

$$PP_M(W) = p(w_1 \ldots w_N)^{-\frac{1}{N}}$$

- Для униграммной модели

$$PP_M(W) = p(w_1 \ldots w_N)^{-\frac{1}{N}} = (\prod_i^N p(w_i))^{-\frac{1}{N}} = (\prod_{j=1}^{|D|} p(w_{(j)})^{n_j})^{-\frac{1}{N}}$$

$$\log_2 PP_M(W) = -\frac{1}{N} \sum_{j=1} n_j \log_2 p(w_{(j)})$$

$w_{(j)}$ — j-ое слово в словаре

- То есть для униграммной модели $\log_2 PP_M(W)$ — это среднее значение отрицательного логарифма вероятности слова в тексте.

Перплексия

- Алгоритм определения качества языковой модели:
  - Разбить данные на обучающую и контрольную выборку.
  - Построить модель по обучающей выборке.
  - Посчитать перплексию на контрольной.
- Перплексия модели M на выборке $W = [w_1, \ldots, w_N]$:

$$PP_M(W) = p(w_1 \ldots w_N)^{-\frac{1}{N}}$$

- Для униграммной модели

$$PP_M(W) = p(w_1 \ldots w_N)^{-\frac{1}{N}} = (\prod_i^N p(w_i))^{-\frac{1}{N}} = (\prod_{j=1}^{|D|} p(w_{(j)})^{n_j})^{-\frac{1}{N}}$$

$$\log_2 PP_M(W) = -\frac{1}{N} \sum_{j=1} n_j \log_2 p(w_{(j)})$$

$w_{(j)}$ — j-ое слово в словаре

- То есть для униграммной модели $\log_2 PP_M(W)$ — это среднее значение отрицательного логарифма вероятности слова в тексте.
- В терминах теории информации — среднее количество битов, нужное на описание одного элемента текста.
- Чем меньше перплексия, тем лучше (точнее описание).

Интерполяция и откат

- Недостатки аддитивного сглаживания:
  - непонятно, как подбирать α (зависит от размера корпуса, размера словаря, порядка энграмм и т. д.)

Интерполяция и откат

- Недостатки аддитивного сглаживания:
  - непонятно, как подбирать α (зависит от размера корпуса, размера словаря, порядка энграмм и т. д.)
  - метод негибкий, не учитывает историю $t_1 \dots t_{n-1}$.

Интерполяция и откат

- Недостатки аддитивного сглаживания:
  - непонятно, как подбирать α (зависит от размера корпуса, размера словаря, порядка энграмм и т. д.)
  - метод негибкий, не учитывает историю $t_1 \dots t_{n-1}$.
- Основная идея: будем использовать $p(t_n|t_2 \dots t_{n-1})$ для вычисления $p(t_n|t_1 \dots t_{n-1})$, если $c(t_n|t_1 \dots t_{n-1}) = 0$.

Интерполяция и откат

- Недостатки аддитивного сглаживания:
  - непонятно, как подбирать α (зависит от размера корпуса, размера словаря, порядка энграмм и т. д.)
  - метод негибкий, не учитывает историю $t_1 \dots t_{n-1}$.
- Основная идея: будем использовать $p(t_n|t_2 \dots t_{n-1})$ для вычисления $p(t_n|t_1 \dots t_{n-1})$, если $c(t_n|t_1 \dots t_{n-1}) = 0$.
- Общая интерполяционная формула:

$$p_I(t_n|t_1 \dots t_{n-1}) = \lambda p_C(t_n|\mathbf{t}_{1,n-1}) + (1 - \lambda)p_I(t_n|\mathbf{t}_{2,n-1})$$

Интерполяция и откат

- Недостатки аддитивного сглаживания:
  - непонятно, как подбирать α (зависит от размера корпуса, размера словаря, порядка энграмм и т. д.)
  - метод негибкий, не учитывает историю $t_1 \dots t_{n-1}$.
- Основная идея: будем использовать $p(t_n|t_2 \dots t_{n-1})$ для вычисления $p(t_n|t_1 \dots t_{n-1})$, если $c(t_n|t_1 \dots t_{n-1}) = 0$.
- Общая интерполяционная формула:

$$p_I(t_n|t_1 \dots t_{n-1}) = \lambda p_C(t_n|\mathbf{t}_{1,n-1}) + (1 - \lambda) p_I(t_n|\mathbf{t}_{2,n-1})$$

$p_C(t_n|t_1 \dots t_{n-1}) = \dfrac{c(t_1 \dots t_{n-1} t_n)}{c(t_1 \dots t_{n-1} \bullet)}$ — “корпусная” вероятность,
λ — коэффициент, вообще говоря, зависящий от $t_1 \dots t_{n-1}$.

Пример

$w_1\,w_2$	w_3	$c(w_1\,w_2\,w_3)$	$p(w_3 \mid w_1\,w_2)$	w_2	w_3	$c(w_2\,w_3)$	$p(w_3 \mid w_2)$
я читал		1832		читал		18149	
я читал	газету	3	0.0016	читал	газету	149	0.0082
я читал	книгу	19	0.0103	читал	книгу	138	0.0076
я читал	лекцию	11	0.0060	читал	лекцию	81	0.0045
я читал	доклад	0	0	читал	доклад	22	0.0012

Пример

$w_1\,w_2$	w_3	$c(w_1\,w_2\,w_3)$	$p(w_3 \mid w_1\,w_2)$	w_2	w_3	$c(w_2\,w_3)$	$p(w_3 \mid w_2)$
я читал		1832		читал		18149	
я читал	газету	3	0.0016	читал	газету	149	0.0082
я читал	книгу	19	0.0103	читал	книгу	138	0.0076
я читал	лекцию	11	0.0060	читал	лекцию	81	0.0045
я читал	доклад	0	0	читал	доклад	22	0.0012

При $\lambda = 0.5$ получаем

$$p(\text{газету}|\text{я читал}) = 0.5 * 0.0016 + 0.5 * 0.0082 = 0.0049$$
$$p(\text{доклад}|\text{я читал}) = 0.5 * 0.0000 + 0.5 * 0.0012 = 0.0006$$

Интерполяция и откат

- Обозначим $\mathbf{t}_{i,j} = t_i \dots t_j$.
- Общая интерполяционная формула:

$$p_I(t_n|\mathbf{t}_{1,n-1}) = \lambda p_C(t_n|\mathbf{t}_{1,n-1}) + (1-\lambda)p_I(t_n|\mathbf{t}_{2,n-1})$$

Интерполяция и откат

- Обозначим $\mathbf{t}_{i,j} = t_i \dots t_j$.
- Общая интерполяционная формула:

$$p_I(t_n|\mathbf{t}_{1,n-1}) = \lambda p_C(t_n|\mathbf{t}_{1,n-1}) + (1-\lambda)p_I(t_n|\mathbf{t}_{2,n-1})$$

- Формула отката (backoff):

$$p_I(t_n|t_1 \dots t_{n-1}) = \begin{cases} \alpha(\mathbf{t}_{1,n-1})p_C(t_n|\mathbf{t}_{1,n-1}), & c(\mathbf{t}_{1,n-1}t_n) > 0, \\ \beta(\mathbf{t}_{1,n-1})p_I(t_n|\mathbf{t}_{2,n-1}), & c(\mathbf{t}_{1,n-1}t_n) = 0 \end{cases}$$

Интерполяция и откат

- Обозначим $\mathbf{t}_{i,j} = t_i \dots t_j$.
- Общая интерполяционная формула:

$$p_I(t_n|\mathbf{t}_{1,n-1}) = \lambda p_C(t_n|\mathbf{t}_{1,n-1}) + (1-\lambda)p_I(t_n|\mathbf{t}_{2,n-1})$$

- Формула отката (backoff):

$$p_I(t_n|t_1 \dots t_{n-1}) = \begin{cases} \alpha(\mathbf{t}_{1,n-1})p_C(t_n|\mathbf{t}_{1,n-1}), & c(\mathbf{t}_{1,n-1}t_n) > 0, \\ \beta(\mathbf{t}_{1,n-1})p_I(t_n|\mathbf{t}_{2,n-1}), & c(\mathbf{t}_{1,n-1}t_n) = 0 \end{cases}$$

- Чем больше λ (α в формуле отката), тем больше мы доверяем истории $\mathbf{t}_{1,n-1}$.

Интерполяция и откат

- Обозначим $\mathbf{t}_{i,j} = t_i \dots t_j$.
- Общая интерполяционная формула:

$$p_I(t_n|\mathbf{t}_{1,n-1}) = \lambda p_C(t_n|\mathbf{t}_{1,n-1}) + (1-\lambda)p_I(t_n|\mathbf{t}_{2,n-1})$$

- Формула отката (backoff):

$$p_I(t_n|t_1 \dots t_{n-1}) = \begin{cases} \alpha(\mathbf{t}_{1,n-1})p_C(t_n|\mathbf{t}_{1,n-1}), & c(\mathbf{t}_{1,n-1}t_n) > 0, \\ \beta(\mathbf{t}_{1,n-1})p_I(t_n|\mathbf{t}_{2,n-1}), & c(\mathbf{t}_{1,n-1}t_n) = 0 \end{cases}$$

- Чем больше λ (α в формуле отката), тем больше мы доверяем истории $\mathbf{t}_{1,n-1}$.
- Много случайных продолжений у $\mathbf{t}_{1,n-1}$ — λ мало.
- Продолжений мало и они частотные — $\lambda \approx 1$

Интерполяция и откат

- Обозначим $\mathbf{t}_{i,j} = t_i \dots t_j$.
- Общая интерполяционная формула:

$$p_I(t_n|\mathbf{t}_{1,n-1}) = \lambda p_C(t_n|\mathbf{t}_{1,n-1}) + (1-\lambda)p_I(t_n|\mathbf{t}_{2,n-1})$$

- Формула отката (backoff):

$$p_I(t_n|t_1 \dots t_{n-1}) = \begin{cases} \alpha(\mathbf{t}_{1,n-1})p_C(t_n|\mathbf{t}_{1,n-1}), & c(\mathbf{t}_{1,n-1}t_n) > 0, \\ \beta(\mathbf{t}_{1,n-1})p_I(t_n|\mathbf{t}_{2,n-1}), & c(\mathbf{t}_{1,n-1}t_n) = 0 \end{cases}$$

- Чем больше λ (α в формуле отката), тем больше мы доверяем истории $\mathbf{t}_{1,n-1}$.
- Много случайных продолжений у $\mathbf{t}_{1,n-1}$ — λ мало.
- Продолжений мало и они частотные — $\lambda \approx 1$
- β подбирают, чтобы сумма вероятностей получилась 1.

Метод Уиттена-Белла

- Метод Уиттена-Белла:

$$
\begin{aligned}
p_I(t_n|\mathbf{t}_{1,n-1}) &= \lambda p_c(t_n|\mathbf{t}_{1,n-1}) + (1-\lambda) p_I(t_n|\mathbf{t}_{2,n-1}) \\
\lambda &= \frac{c(t_1 \dots t_{n-1} \odot)}{c(t_1 \dots t_{n-1} \odot) + N_{1+}(t_1 \dots t_{n-1})} \\
N_{1+}(t_1 \dots t_{n-1}) &= |\{t|c(t_1 \dots t_{n-1} t) > 0\} \\
N_{1+}(t_1 \dots t_{n-1}) &- \text{“число продолжений”}
\end{aligned}
$$

Метод Уиттена-Белла

- Метод Уиттена-Белла:

$$
\begin{aligned}
p_I(t_n|\mathbf{t}_{1,n-1}) &= \lambda p_c(t_n|\mathbf{t}_{1,n-1}) + (1-\lambda) p_I(t_n|\mathbf{t}_{2,n-1}) \\
\lambda &= \frac{c(t_1 \dots t_{n-1}\odot)}{c(t_1 \dots t_{n-1}\odot) + N_{1+}(t_1 \dots t_{n-1})} \\
N_{1+}(t_1 \dots t_{n-1}) &= |\{t|c(t_1 \dots t_{n-1} t) > 0\} \\
N_{1+}(t_1 \dots t_{n-1}) &- \text{“число продолжений”}
\end{aligned}
$$

- Пример (британский национальный корпус):

w_1	$c(w_1\odot)$	$N_{1+}(w_1)$	$N_{3+}(w_1)$	$\lambda(w_1)$	$1-\lambda(w_1)$
spite	2899	59	15	$\frac{2899}{2899+59} = 0.980$	0.02
stupid	2898	602	117	$\frac{2898}{2898+602} = 0.828$	0.172

Метод Уиттена-Белла

- Метод Уиттена-Белла:

$$
\begin{aligned}
p_I(t_n|\mathbf{t}_{1,n-1}) &= \lambda p_c(t_n|\mathbf{t}_{1,n-1}) + (1-\lambda) p_I(t_n|\mathbf{t}_{2,n-1}) \\
\lambda &= \frac{c(t_1 \dots t_{n-1}\odot)}{c(t_1 \dots t_{n-1}\odot) + N_{1+}(t_1 \dots t_{n-1})} \\
N_{1+}(t_1 \dots t_{n-1}) &= |\{t|c(t_1 \dots t_{n-1} t) > 0\} \\
N_{1+}(t_1 \dots t_{n-1}) &- \text{“число продолжений”}
\end{aligned}
$$

- Пример (британский национальный корпус):

w_1	$c(w_1\odot)$	$N_{1+}(w_1)$	$N_{3+}(w_1)$	$\lambda(w_1)$	$1-\lambda(w_1)$
spite	2899	59	15	$\frac{2899}{2899+59} = 0.980$	0.02
stupid	2898	602	117	$\frac{2898}{2898+602} = 0.828$	0.172

- Униграммная модель для *stupid* в 86 раз более значима, чем для *spite*.

Метод Кнезера-Нея

- Метод Уиттена-Белла учитывает количество возможных правых продолжений.

Метод Кнезера-Нея

- Метод Уиттена-Белла учитывает количество возможных правых продолжений.
- Можно учитывать и левые:

Метод Кнезера-Нея

- Метод Уиттена-Белла учитывает количество возможных правых продолжений.
- Можно учитывать и левые:
  - предшественником слова *York* практически всегда будет слово *New*.

Метод Кнезера-Нея

- Метод Уиттена-Белла учитывает количество возможных правых продолжений.
- Можно учитывать и левые:
  - предшественником слова *York* практически всегда будет слово *New*.
  - соответственно, $p(York|w) \approx 0$ при $w \neq new$.
  - при этом $p_{UNI}(York) = \frac{c(York)}{N}$ достаточно велика.

Метод Кнезера-Нея

- Метод Уиттена-Белла учитывает количество возможных правых продолжений.
- Можно учитывать и левые:
  - предшественником слова *York* практически всегда будет слово *New*.
  - соответственно, $p(York|w) \approx 0$ при $w \neq new$.
  - при этом $p_{UNI}(York) = \frac{c(York)}{N}$ достаточно велика.
- В методе Кнезера-Нея униграммная вероятность считается по формуле

$$p_{KN}(w) = \frac{N_{1+}(\bullet w)}{\sum_{w'} N_{1+}(\bullet w')}$$

$$N_{1+}(\bullet w) = |\{w_1 | c(w_1 w) > 0\}|$$ — число левых продолжений

Метод Кнезера-Нея

- Для перераспределения вероятностей на новые слова используется дисконтирование (из всех счётчиков вычитается δ).

$$p_0(t_n|\mathbf{t}_{1,n-1}) = \frac{c(\mathbf{t}_{1,n-1}t_n) - \delta}{c(\mathbf{t}_{1,n-1}\bullet)},\ c(\mathbf{t}_{1,n-1}t_n) > 0$$

Метод Кнезера-Нея

- Для перераспределения вероятностей на новые слова используется дисконтирование (из всех счётчиков вычитается δ).

$$p_0(t_n|\mathbf{t}_{1,n-1}) = \frac{c(\mathbf{t}_{1,n-1}t_n) - \delta}{c(\mathbf{t}_{1,n-1}\bullet)},\ c(\mathbf{t}_{1,n-1}t_n) > 0$$

- В интерполяционной формуле

$$p_{KN}(t_n|\mathbf{t}_{1,n-1}) = p_0(t_n|\mathbf{t}_{1,n-1}) + \beta(\mathbf{t}_{1,n-1})p_{KN}(t_n|\mathbf{t}_{2,n-1})$$

получаем $\beta = \dfrac{\delta N_{1+}(\mathbf{t}_{1,n-1})}{c(\mathbf{t}_{1,n-1}\bullet)}$ (выведите эту формулу).

- Для униграммных вероятностей — формула с предыдущего слайда.

Метод Кнезера-Нея

- В интерполяционной формуле

$$p_{KN}(t_n|\mathbf{t}_{1,n-1}) = p_0(t_n|\mathbf{t}_{1,n-1}) + \beta(\mathbf{t}_{1,n-1})p_{KN}(t_n|\mathbf{t}_{1,n-2})$$

Метод Кнезера-Нея

- В интерполяционной формуле

$$p_{KN}(t_n|\mathbf{t}_{1,n-1}) = p_0(t_n|\mathbf{t}_{1,n-1}) + \beta(\mathbf{t}_{1,n-1})p_{KN}(t_n|\mathbf{t}_{1,n-2})$$

- Основная проблема: поиск оптимальной δ. В стандартной реализации

$$\begin{aligned} \delta_1 &= 1 - 2Y\frac{N_2}{N_1} & \delta_2 &= 1 - 3Y\frac{N_3}{N_2} \\ \delta_{\geqslant 3} &= 1 - 4Y\frac{N_4}{N_3} & Y &= \frac{N_1}{N_1 + 2N_2} \end{aligned}$$

Метод Кнезера-Нея

- В интерполяционной формуле

$$p_{KN}(t_n|\mathbf{t}_{1,n-1}) = p_0(t_n|\mathbf{t}_{1,n-1}) + \beta(\mathbf{t}_{1,n-1})p_{KN}(t_n|\mathbf{t}_{1,n-2})$$

- Основная проблема: поиск оптимальной δ. В стандартной реализации

$$\begin{aligned} \delta_1 &= 1 - 2Y\frac{N_2}{N_1} & \delta_2 &= 1 - 3Y\frac{N_3}{N_2} \\ \delta_{\geqslant 3} &= 1 - 4Y\frac{N_4}{N_3} & Y &= \frac{N_1}{N_1 + 2N_2} \end{aligned}$$

- Здесь δ_i — дисконт для счётчиков, равных i, N_i — число энграмм частоты i.

Метод Кнезера-Нея

- В интерполяционной формуле

$$p_{KN}(t_n|\mathbf{t}_{1,n-1}) = p_0(t_n|\mathbf{t}_{1,n-1}) + \beta(\mathbf{t}_{1,n-1})p_{KN}(t_n|\mathbf{t}_{1,n-2})$$

- Основная проблема: поиск оптимальной δ. В стандартной реализации

$$\begin{aligned} \delta_1 &= 1 - 2Y\frac{N_2}{N_1} & \delta_2 &= 1 - 3Y\frac{N_3}{N_2} \\ \delta_{\geqslant 3} &= 1 - 4Y\frac{N_4}{N_3} & Y &= \frac{N_1}{N_1 + 2N_2} \end{aligned}$$

- Здесь δ_i — дисконт для счётчиков, равных i, N_i — число энграмм частоты i.
- В случае лексических энграмм метод Кнезера-Нея наиболее мощный.

Метод Кнезера-Нея

- В интерполяционной формуле

$$p_{KN}(t_n|\mathbf{t}_{1,n-1}) = p_0(t_n|\mathbf{t}_{1,n-1}) + \beta(\mathbf{t}_{1,n-1})p_{KN}(t_n|\mathbf{t}_{1,n-2})$$

- Основная проблема: поиск оптимальной δ. В стандартной реализации

$$\begin{aligned} \delta_1 &= 1 - 2Y\frac{N_2}{N_1} & \delta_2 &= 1 - 3Y\frac{N_3}{N_2} \\ \delta_{\geqslant 3} &= 1 - 4Y\frac{N_4}{N_3} & Y &= \frac{N_1}{N_1 + 2N_2} \end{aligned}$$

- Здесь δ_i — дисконт для счётчиков, равных i, N_i — число энграмм частоты i.
- В случае лексических энграмм метод Кнезера-Нея наиболее мощный.
- Недостаток: работает только в случае $N_1 \geqslant N_2 \geqslant N_3 \ldots$, поэтому плохо применим к символьным и морфологическим энграммам.

Интерполяция через удаление

- В морфологическом анализаторе TnT используется триграммная модель для морфологических меток, основанная на интерполяции через удаление:

$$p(t_3|t_1\,t_2) = \mu_3 p_C(t_3|t_1\,t_2) + \mu_2 p_C(t_3|t_2) + \mu_1 p_C(t_3)$$

Интерполяция через удаление

- В морфологическом анализаторе TnT используется триграммная модель для морфологических меток, основанная на интерполяции через удаление:

$$p(t_3|t_1\,t_2) = \mu_3 p_C(t_3|t_1\,t_2) + \mu_2 p_C(t_3|t_2) + \mu_1 p_C(t_3)$$

- Метод вычисления μ_1, μ_2, μ_3:
  - Для каждой триграммы $t_1\,t_2\,t_3$ в корпусе вычислить величины

$$f_3 = \frac{c(t_1\,t_2\,t_3) - 1}{c(t_1\,t_2\,\bullet)},\ f_2 = \frac{c(t_2\,t_3) - 1}{c(t_2\,\bullet)},\ f_1 = \frac{c(t_3) - 1}{c(\bullet)}.$$

Интерполяция через удаление

- В морфологическом анализаторе TnT используется триграммная модель для морфологических меток, основанная на интерполяции через удаление:

$$p(t_3|t_1\,t_2) = \mu_3 p_C(t_3|t_1\,t_2) + \mu_2 p_C(t_3|t_2) + \mu_1 p_C(t_3)$$

- Метод вычисления μ_1, μ_2, μ_3:
  - Для каждой триграммы $t_1\,t_2\,t_3$ в корпусе вычислить величины $f_3 = \dfrac{c(t_1\,t_2\,t_3) - 1}{c(t_1\,t_2\,\bullet)}$, $f_2 = \dfrac{c(t_2\,t_3) - 1}{c(t_2\,\bullet)}$, $f_1 = \dfrac{c(t_3) - 1}{c(\bullet)}$.
  - Увеличить μ_k, где $k = \operatorname{argmax}_j f_j$.

Интерполяция через удаление

- В морфологическом анализаторе TnT используется триграммная модель для морфологических меток, основанная на интерполяции через удаление:

$$p(t_3|t_1\,t_2) = \mu_3 p_C(t_3|t_1\,t_2) + \mu_2 p_C(t_3|t_2) + \mu_1 p_C(t_3)$$

- Метод вычисления μ_1, μ_2, μ_3:
  - Для каждой триграммы $t_1\,t_2\,t_3$ в корпусе вычислить величины $f_3 = \dfrac{c(t_1\,t_2\,t_3) - 1}{c(t_1\,t_2\,\bullet)}$, $f_2 = \dfrac{c(t_2\,t_3) - 1}{c(t_2\,\bullet)}$, $f_1 = \dfrac{c(t_3) - 1}{c(\bullet)}$.
  - Увеличить μ_k, где $k = \operatorname{argmax}_j f_j$.
- Если не вычитать 1, метод переобучается (слишком большой вес у триграмм).
- Этот метод не позволяет учитывать метки, не встречавшиеся в корпусе, можно добавить к $p(t_3|t_1\,t_2)$ слагаемое $\mu_0 \frac{1}{|D|}$.

Интерполяция через удаление

- В морфологическом анализаторе TnT используется триграммная модель для морфологических меток, основанная на интерполяции через удаление:

$$p(t_3|t_1\,t_2) = \mu_3 p_C(t_3|t_1\,t_2) + \mu_2 p_C(t_3|t_2) + \mu_1 p_C(t_3)$$

- Метод вычисления μ_1, μ_2, μ_3:
  - Для каждой триграммы $t_1\,t_2\,t_3$ в корпусе вычислить величины $f_3 = \dfrac{c(t_1\,t_2\,t_3) - 1}{c(t_1\,t_2\,\bullet)}$, $f_2 = \dfrac{c(t_2\,t_3) - 1}{c(t_2\,\bullet)}$, $f_1 = \dfrac{c(t_3) - 1}{c(\bullet)}$.
  - Увеличить μ_k, где $k = \operatorname{argmax}_j f_j$.
- Если не вычитать 1, метод переобучается (слишком большой вес у триграмм).
- Этот метод не позволяет учитывать метки, не встречавшиеся в корпусе, можно добавить к $p(t_3|t_1\,t_2)$ слагаемое $\mu_0 \frac{1}{|D|}$.
- μ_0 увеличиваем, если $\max\limits_j f_j = 0$.

Введение

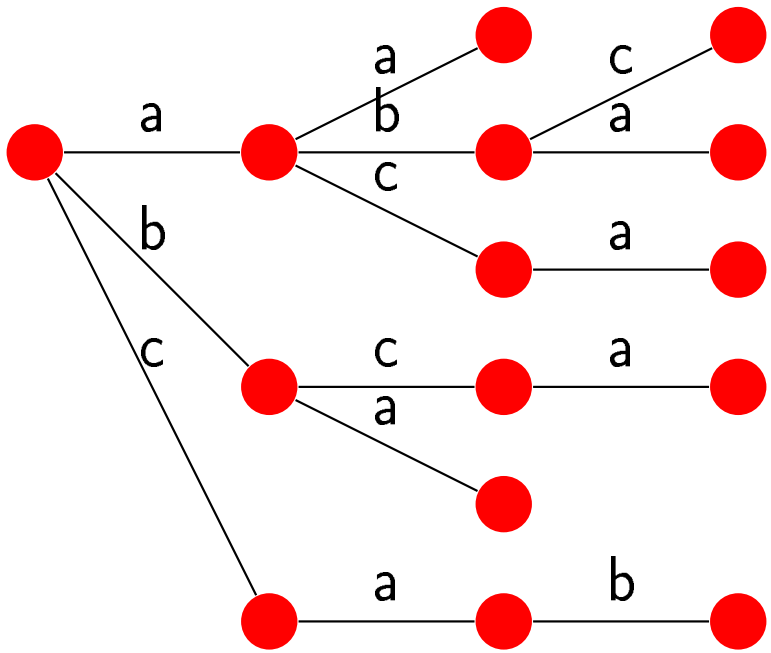
- Программы для построения энграммных моделей: KenLM (Linux, есть Python-интерфейс), SRILM (Linux, Windows, есть Python-интерфейс), IRSTLM, MITLM.

Введение

- Программы для построения энграммных моделей: KenLM (Linux, есть Python-интерфейс), SRILM (Linux, Windows, есть Python-интерфейс), IRSTLM, MITLM.
- Оптимальная структура — префиксный бор.

Введение

- Программы для построения энграммных моделей: KenLM (Linux, есть Python-интерфейс), SRILM (Linux, Windows, есть Python-интерфейс), IRSTLM, MITLM.
- Оптимальная структура — префиксный бор.
- Бор для слова *abcacaba* и $n = 3$:



Вычисление вероятностей по бору

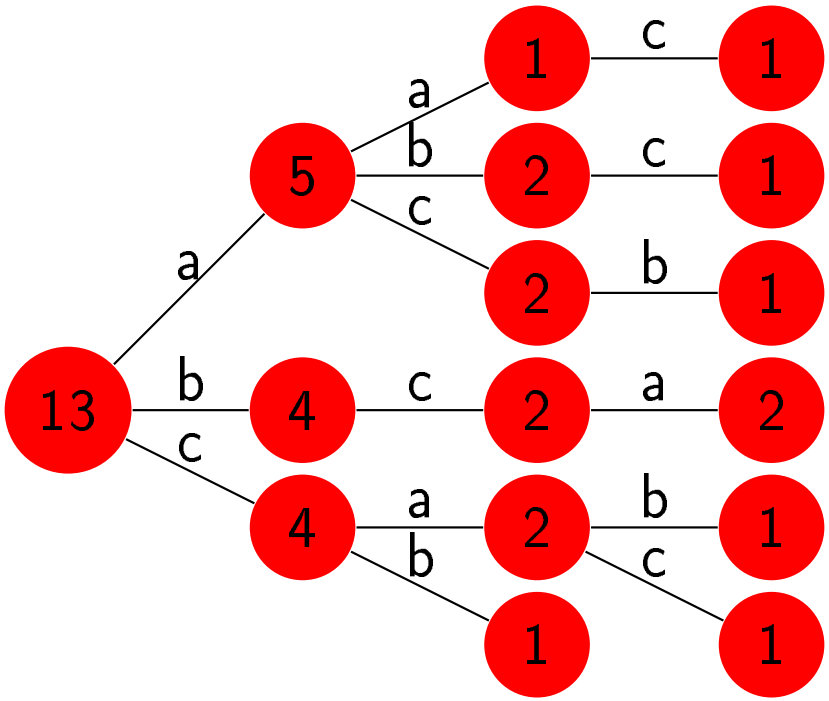
- Вероятность (интерполированную) $p(t_n|\mathbf{t}_{1,n-1})$ можно хранить в вершине $t_1 \dots t_n$.

Вычисление вероятностей по бору

- Вероятность (интерполированную) $p(t_n|\mathbf{t}_{1,n-1})$ можно хранить в вершине $t_1 \dots t_n$.
- Для начала можно аккумулировать счётчики, а потом посчитать вероятности проходом по дереву от корня к листьям.

Вычисление вероятностей по бору

- Вероятность (интерполированную) $p(t_n|\mathbf{t}_{1,n-1})$ можно хранить в вершине $t_1 \dots t_n$.
- Для начала можно аккумулировать счётчики, а потом посчитать вероятности проходом по дереву от корня к листьям.
- Бор для слов *abcac*, *bcab*, *aacb*:



Подсчёт вероятностей

- Посчитаем вероятности по методу Уиттена-Белла, начиная с вершин на глубине 1.
- Будем считать, что для униграмм сглаживающая формула имеет вид:

$$\begin{aligned} P(t) &= \lambda \frac{c(t)}{c(\bullet)} + (1-\lambda)P(\varepsilon), \\ \lambda &= \frac{\sum\limits_t c(t)}{\sum\limits_t c(t) + N_{1+}(\bullet)} = \frac{c(\bullet)}{c(\bullet) + N_{1+}(\bullet)} \\ P(\varepsilon) &= \frac{1}{|D|} = \frac{1}{N_{1+}(\bullet)}, \ |D| \text{ — размер словаря} \end{aligned}$$

Подсчёт вероятностей

- Посчитаем вероятности по методу Уиттена-Белла, начиная с вершин на глубине 1.
- Будем считать, что для униграмм сглаживающая формула имеет вид:

$$\begin{aligned} P(t) &= \lambda \frac{c(t)}{c(\bullet)} + (1-\lambda)P(\varepsilon), \\ \lambda &= \frac{\sum\limits_t c(t)}{\sum\limits_t c(t) + N_{1+}(\bullet)} = \frac{c(\bullet)}{c(\bullet) + N_{1+}(\bullet)} \\ P(\varepsilon) &= \frac{1}{|D|} = \frac{1}{N_{1+}(\bullet)}, \ |D| \text{ — размер словаря} \end{aligned}$$

- Окончательно получаем $P(t) = \dfrac{c(t)+1}{c(\bullet)+|D|}$,

  то есть $P(a) = \dfrac{5+1}{13+3} = \dfrac{3}{8}$.

Подсчёт вероятностей

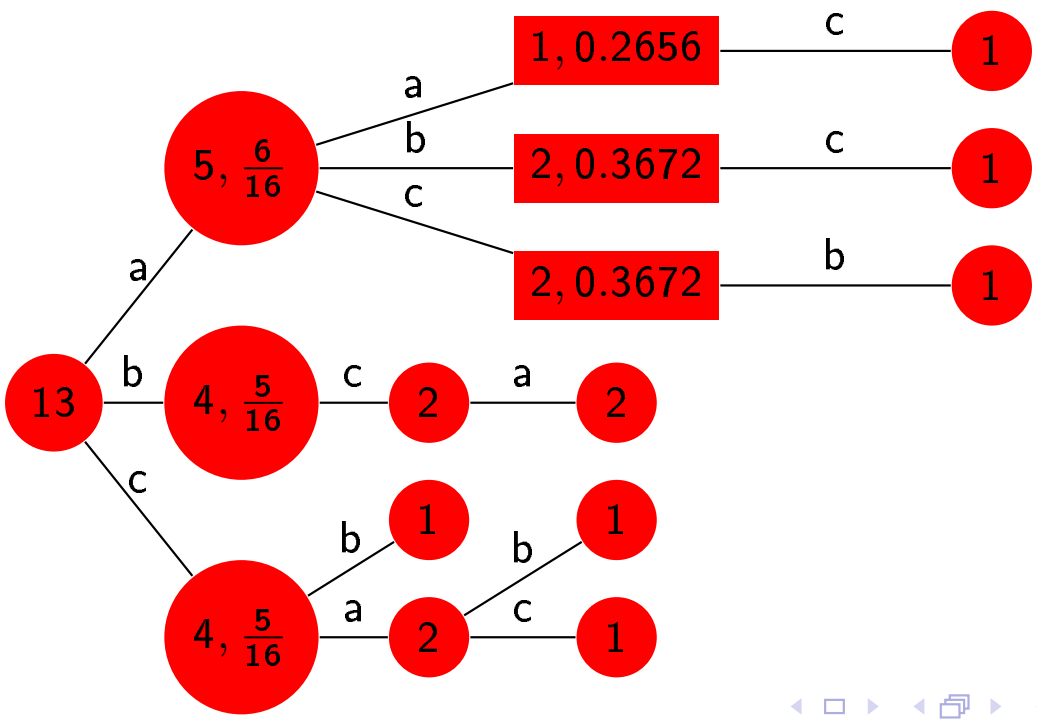
- Вычислим $\lambda(a)$: $\lambda(a) = \dfrac{c(a\bullet)}{c(a\bullet) + N_{1+}(a)} = \dfrac{5}{8}$.

Подсчёт вероятностей

- Вычислим $\lambda(a)$: $\lambda(a) = \dfrac{c(a\bullet)}{c(a\bullet) + N_{1+}(a)} = \dfrac{5}{8}$.
- Тогда $p(a|a) = \dfrac{5}{8}\dfrac{c(aa)}{c(a\bullet)} + \dfrac{3}{8}p(a) = \dfrac{5 \cdot 1}{8 \cdot 5} + \dfrac{3 \cdot 3}{8 \cdot 8} \approx 0.2656.$

Подсчёт вероятностей

- Вычислим $\lambda(a)$: $\lambda(a) = \dfrac{c(a\bullet)}{c(a\bullet) + N_{1+}(a)} = \dfrac{5}{8}$.
- Тогда $p(a|a) = \dfrac{5}{8}\dfrac{c(aa)}{c(a\bullet)} + \dfrac{3}{8}p(a) = \dfrac{5 \cdot 1}{8 \cdot 5} + \dfrac{3 \cdot 3}{8 \cdot 8} \approx 0.2656$.
- Аналогично $p(b|a) = p(c|a) = \dfrac{5 \cdot 2}{8 \cdot 5} + \dfrac{3 \cdot 5}{8 \cdot 16} \approx 0.3672$.



Подсчёт вероятностей

- Для каждой встретившейся n-граммы $t_1 \ldots t_n$ в боре хранится вероятность $p(t_n|\mathbf{t}_{1,n-1})$.

Подсчёт вероятностей

- Для каждой встретившейся n-граммы $t_1 \dots t_n$ в боре хранится вероятность $p(t_n|\mathbf{t}_{1,n-1})$.
- Что делать, если энграмма $t_1 \dots t_n$ не встречалась ?
- В этом случае $P(t_n|\mathbf{t}_{1,n-1}) = \beta(\mathbf{t}_{1,n-1})P(t_n|\mathbf{t}_{2,n-1})$.

Подсчёт вероятностей

- Для каждой встретившейся n-граммы $t_1 \dots t_n$ в боре хранится вероятность $p(t_n|\mathbf{t}_{1,n-1})$.
- Что делать, если энграмма $t_1 \dots t_n$ не встречалась ?
- В этом случае $P(t_n|\mathbf{t}_{1,n-1}) = \beta(\mathbf{t}_{1,n-1})P(t_n|\mathbf{t}_{2,n-1})$.
- То есть в каждой вершине нужно хранить и множители для отката.

Подсчёт вероятностей

- Для каждой встретившейся n-граммы $t_1 \dots t_n$ в боре хранится вероятность $p(t_n|\mathbf{t}_{1,n-1})$.
- Что делать, если энграмма $t_1 \dots t_n$ не встречалась ?
- В этом случае $P(t_n|\mathbf{t}_{1,n-1}) = \beta(\mathbf{t}_{1,n-1})P(t_n|\mathbf{t}_{2,n-1})$.
- То есть в каждой вершине нужно хранить и множители для отката.
- При подсчёте вероятности мы постепенно отрезаем начальные слова у истории $\mathbf{t}_{1,n-1}$. пока не получим историю, которая допускает t_n в качестве продолжения.

Подсчёт вероятностей

- Для каждой встретившейся n-граммы $t_1 \dots t_n$ в боре хранится вероятность $p(t_n|\mathbf{t}_{1,n-1})$.
- Что делать, если энграмма $t_1 \dots t_n$ не встречалась ?
- В этом случае $P(t_n|\mathbf{t}_{1,n-1}) = \beta(\mathbf{t}_{1,n-1})P(t_n|\mathbf{t}_{2,n-1})$.
- То есть в каждой вершине нужно хранить и множители для отката.
- При подсчёте вероятности мы постепенно отрезаем начальные слова у истории $\mathbf{t}_{1,n-1}$. пока не получим историю, которая допускает t_n в качестве продолжения.
- При этом β — мультипликативный штраф за переход к более короткой истории.

Алгоритм вычисления вероятностей

```
 1: function getProb(history, word)
 2:     ▷ находим самый длинный суффикс history, имеющийся в дереве
 3:     history ← longest_suffix(history)
 4:     prob ← 1
 5:     while history ≠ [] do
 6:         if hasChild(history, word) then
 7:             ▷ Умножаем на вероятность p(word|history)
 8:             prob *= getChild(history, word).prob
 9:             return(prob)
10:         else
11:             history ← history[1 :]          ▷ Переходим к более короткой истории
12:             prob *= history.backoff                  ▷ Умножаем на β(history)
13:         end if
14:     end while
15:     ▷ Теперь история пуста
16:     if hasChild(history, word) then
17:         prob *= getChild(history, word).prob
18:     else
19:         prob *= getUnknownWordProb()          ▷ Встретили неизвестное слово
20:     end if
21:     return(prob)
22: end function
```

Применения языковых моделей

- Применение языковых моделей:
  - Автоматический перевод
    ($p(s)$ в формуле $\widehat{t} = \text{argmax}_t\, p(s|t)p(t)$),

Применения языковых моделей

- Применение языковых моделей:
  - Автоматический перевод
    ($p(s)$ в формуле $\widehat{t} = \operatorname{argmax}_t p(s|t)p(t)$),
  - Вычислительная фонология (распознавание речи и т. д.),

Применения языковых моделей

- Применение языковых моделей:
  - Автоматический перевод ($p(s)$ в формуле $\widehat{t} = \operatorname{argmax}_t p(s|t)p(t)$),
  - Вычислительная фонология (распознавание речи и т. д.),
  - Вычислительная морфология (вычисление вероятности последовательности состояний).

Применения языковых моделей

- Применение языковых моделей:
  - Автоматический перевод ($p(s)$ в формуле $\widehat{t} = \operatorname{argmax}_t p(s|t)p(t)$),
  - Вычислительная фонология (распознавание речи и т. д.),
  - Вычислительная морфология (вычисление вероятности последовательности состояний).
- Свойства языковых моделей:
  - Большое количество параметров (для моделей глубины n — порядка $|D|^n$).

Применения языковых моделей

- Применение языковых моделей:
  - Автоматический перевод ($p(s)$ в формуле $\widehat{t} = \operatorname{argmax}_t p(s|t)p(t)$),
  - Вычислительная фонология (распознавание речи и т. д.),
  - Вычислительная морфология (вычисление вероятности последовательности состояний).
- Свойства языковых моделей:
  - Большое количество параметров (для моделей глубины n — порядка $|D|^n$).
  - Большое количество редких или неизвестных слов (для лексических энграмм).

Применения языковых моделей

- Применение языковых моделей:
  - Автоматический перевод ($p(s)$ в формуле $\widehat{t} = \operatorname{argmax}_t p(s|t)p(t)$),
  - Вычислительная фонология (распознавание речи и т. д.),
  - Вычислительная морфология (вычисление вероятности последовательности состояний).
- Свойства языковых моделей:
  - Большое количество параметров (для моделей глубины n — порядка $|D|^n$).
  - Большое количество редких или неизвестных слов (для лексических энграмм).
  - Большая глубина модели (в морфологии/фонологии, $n = 8$).

Недостатки энграммных моделей

- Трудности использования языковых моделей:
  - Большие затраты памяти, необходимы эффективные структуры данных.

Недостатки энграммных моделей

- Трудности использования языковых моделей:
  - Большие затраты памяти, необходимы эффективные структуры данных.
  - В лексических моделях много редких слов (закон Ципфа), которые не влияют на свойства модели, но занимают много памяти.

Недостатки энграммных моделей

- Трудности использования языковых моделей:
  - Большие затраты памяти, необходимы эффективные структуры данных.
  - В лексических моделях много редких слов (закон Ципфа), которые не влияют на свойства модели, но занимают много памяти.
- Недостатки энграммного метода:
  - Нельзя учесть историю большой глубины (лексические модели).

Недостатки энграммных моделей

- Трудности использования языковых моделей:
  - Большие затраты памяти, необходимы эффективные структуры данных.
  - В лексических моделях много редких слов (закон Ципфа), которые не влияют на свойства модели, но занимают много памяти.
- Недостатки энграммного метода:
  - Нельзя учесть историю большой глубины (лексические модели).
  - Не учитываются дистантные и разрывные зависимости (в триграмме $\mathrm{PREP\ ADJ\ N}$ не учитывается зависимость существительного от предлога).

Недостатки энграммных моделей

- Трудности использования языковых моделей:
  - Большие затраты памяти, необходимы эффективные структуры данных.
  - В лексических моделях много редких слов (закон Ципфа), которые не влияют на свойства модели, но занимают много памяти.
- Недостатки энграммного метода:
  - Нельзя учесть историю большой глубины (лексические модели).
  - Не учитываются дистантные и разрывные зависимости (в триграмме $\mathrm{PREP\ ADJ\ N}$ не учитывается зависимость существительного от предлога).
  - Элементы энграммы представляются как единое целое (не учитываются граммемы внутри морфологической метки).

Модификации энграммных моделей

Недостатки энграммного метода:

- В лексических моделях много редких слов.
- **Решение:** разбивать слова на классы (например, все слова частоты 1 попадут в 1 класс, Brown et al., 1992).

Модификации энграммных моделей

Недостатки энграммного метода:

- В лексических моделях много редких слов.
- **Решение:** разбивать слова на классы (например, все слова частоты 1 попадут в 1 класс, Brown et al., 1992).
- Не учитываются дистантные и разрывные зависимости.
- **Решение:** скипграммы (при $c(w_1 w_2 w) = 0$ для вычисления $p(w|w_1 w_2)$ используют $c(w_1 \bullet w)$ и $c(w_2 w)$).

Модификации энграммных моделей

Недостатки энграммного метода:

- В лексических моделях много редких слов.
- **Решение:** разбивать слова на классы (например, все слова частоты 1 попадут в 1 класс, Brown et al., 1992).
- Не учитываются дистантные и разрывные зависимости.
- **Решение:** скипграммы (при $c(w_1 w_2 w) = 0$ для вычисления $p(w|w_1 w_2)$ используют $c(w_1 \bullet w)$ и $c(w_2 w)$).
- Элементы энграммы представляются как единое целое.
- **Решение:** факторные модели (каждое слово w представляется как совокупность факторов $f_1(w), \ldots, f_m(w)$, Kirchhoff, Blimes, 2003).

Модификации энграммных моделей

Недостатки энграммного метода:

- В лексических моделях много редких слов.
- **Решение:** разбивать слова на классы (например, все слова частоты 1 попадут в 1 класс, Brown et al., 1992).
- Не учитываются дистантные и разрывные зависимости.
- **Решение:** скипграммы (при $c(w_1 w_2 w) = 0$ для вычисления $p(w|w_1 w_2)$ используют $c(w_1 \bullet w)$ и $c(w_2 w)$).
- Элементы энграммы представляются как единое целое.
- **Решение:** факторные модели (каждое слово w представляется как совокупность факторов $f_1(w), \ldots, f_m(w)$, Kirchhoff, Blimes, 2003).
- Условные вероятности оцениваются “наивно”.
- **Решение:** использовать при оценке вероятностей в факторных моделях машинное обучение.

Модификации энграммных моделей

Недостатки энграммного метода:

- В лексических моделях много редких слов.
- **Решение:** разбивать слова на классы (например, все слова частоты 1 попадут в 1 класс, Brown et al., 1992).
- Не учитываются дистантные и разрывные зависимости.
- **Решение:** скипграммы (при $c(w_1 w_2 w) = 0$ для вычисления $p(w|w_1 w_2)$ используют $c(w_1 \bullet w)$ и $c(w_2 w)$).
- Элементы энграммы представляются как единое целое.
- **Решение:** факторные модели (каждое слово w представляется как совокупность факторов $f_1(w), \ldots, f_m(w)$, Kirchhoff, Blimes, 2003).
- Условные вероятности оцениваются “наивно”.
- **Решение:** использовать при оценке вероятностей в факторных моделях машинное обучение.
- **Решение для всех проблем:** модели, основанные на нейронных сетях (Mikolov, 2013; Ling et al., 2016; Bojanowski et al., 2016, etc.)

Энграммные модели, основанные на классах

- Зачастую из одного семантического класса имеют близкую контекстную вероятность.

$$p(\text{пятницу}|\text{я уезжаю в}) \approx p(\text{четверг}|\text{я уезжаю в})$$

Энграммные модели, основанные на классах

- Зачастую из одного семантического класса имеют близкую контекстную вероятность.

$$p(\text{пятницу}|\text{я уезжаю в}) \approx p(\text{четверг}|\text{я уезжаю в})$$

- Будем считать, что каждое слово жёстко относится к своему классу.

Энграммные модели, основанные на классах

- Зачастую из одного семантического класса имеют близкую контекстную вероятность.

$$p(\text{пятницу}|\text{я уезжаю в}) \approx p(\text{четверг}|\text{я уезжаю в})$$

- Будем считать, что каждое слово жёстко относится к своему классу.
- Дополнительно будем считать, что энграммные вероятности зависят только от классов предыдущих слов.

$$p(w_n|\mathbf{w}_{1,n-1}) = p(w_n|c_n, \mathbf{c}_{1,n-1}) = p(w_n|c_n)p(c_n|\mathbf{c}_{1,n-1})$$

Энграммные модели, основанные на классах

- Зачастую из одного семантического класса имеют близкую контекстную вероятность.

$$p(\text{пятницу}|\text{я уезжаю в}) \approx p(\text{четверг}|\text{я уезжаю в})$$

- Будем считать, что каждое слово жёстко относится к своему классу.
- Дополнительно будем считать, что энграммные вероятности зависят только от классов предыдущих слов.

$$p(w_n|\mathbf{w}_{1,n-1}) = p(w_n|c_n, \mathbf{c}_{1,n-1}) = p(w_n|c_n)p(c_n|\mathbf{c}_{1,n-1})$$

- $p(c_n|\mathbf{c}_{1,n-1})$ — энграммные вероятности для классов (считаются стандартным образом).
- $p(w_n|c_n)$ — вероятность слова в своём классе.

Энграммные модели, основанные на классах

- Зачастую из одного семантического класса имеют близкую контекстную вероятность.

$$p(\text{пятницу}|\text{я уезжаю в}) \approx p(\text{четверг}|\text{я уезжаю в})$$

- Будем считать, что каждое слово жёстко относится к своему классу.
- Дополнительно будем считать, что энграммные вероятности зависят только от классов предыдущих слов.

$$p(w_n|\mathbf{w}_{1,n-1}) = p(w_n|c_n, \mathbf{c}_{1,n-1}) = p(w_n|c_n)p(c_n|\mathbf{c}_{1,n-1})$$

- $p(c_n|\mathbf{c}_{1,n-1})$ — энграммные вероятности для классов (считаются стандартным образом).
- $p(w_n|c_n)$ — вероятность слова в своём классе.
- Основная проблема: как делить на классы.

Энграммные модели, основанные на классах

- Можно использовать заранее известное разделение на семантические классы.
- Можно провести кластеризацию дистрибутивных векторов и использовать получившиеся кластеры.

Энграммные модели, основанные на классах

- Можно использовать заранее известное разделение на семантические классы.
- Можно провести кластеризацию дистрибутивных векторов и использовать получившиеся кластеры.
- Однако чаще всего классы вычисляются параллельно с обучением модели.

Энграммные модели, основанные на классах

- Можно использовать заранее известное разделение на семантические классы.
- Можно провести кластеризацию дистрибутивных векторов и использовать получившиеся кластеры.
- Однако чаще всего классы вычисляются параллельно с обучением модели.
- Алгоритм Брауна для разделения на k кластеров:
  - Вначале отнести каждое из k наиболее частотных слов к своему кластеру.

Энграммные модели, основанные на классах

- Можно использовать заранее известное разделение на семантические классы.
- Можно провести кластеризацию дистрибутивных векторов и использовать получившиеся кластеры.
- Однако чаще всего классы вычисляются параллельно с обучением модели.
- Алгоритм Брауна для разделения на k кластеров:
  - Вначале отнести каждое из k наиболее частотных слов к своему кластеру.
  - На каждом шаге рассматривать следующее слово из словаря в качестве одноэлементного кластера.

Энграммные модели, основанные на классах

- Можно использовать заранее известное разделение на семантические классы.
- Можно провести кластеризацию дистрибутивных векторов и использовать получившиеся кластеры.
- Однако чаще всего классы вычисляются параллельно с обучением модели.
- Алгоритм Брауна для разделения на k кластеров:
  - Вначале отнести каждое из k наиболее частотных слов к своему кластеру.
  - На каждом шаге рассматривать следующее слово из словаря в качестве одноэлементного кластера.
  - После этого объединить какие-то два кластера оптимальным образом.
- Оптимально значит так, чтобы вероятность обучающего корпуса была максимальной.

Факторные энграммные модели

- Общая формула отката:

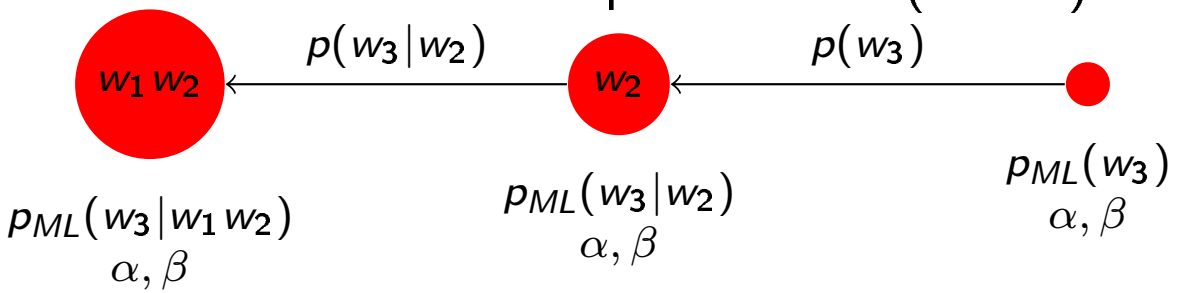
$$p_{BO}(w_n|\mathbf{w}_{1,n-1}) = \begin{cases} \alpha(\mathbf{w}_{1,n-1})\dfrac{c(\mathbf{w}_{1,n-1}w_n)}{c(\mathbf{w}_{1,n-1}\bullet)}, & c(\mathbf{w}_{1,n-1}w_n) > 0, \\ \beta(\mathbf{w}_{1,n-1})p_{BO}(w_n|\mathbf{w}_{2,n-1}), & c(\mathbf{w}_{1,n-1}w_n) = 0 \end{cases}$$

Факторные энграммные модели

- Общая формула отката:

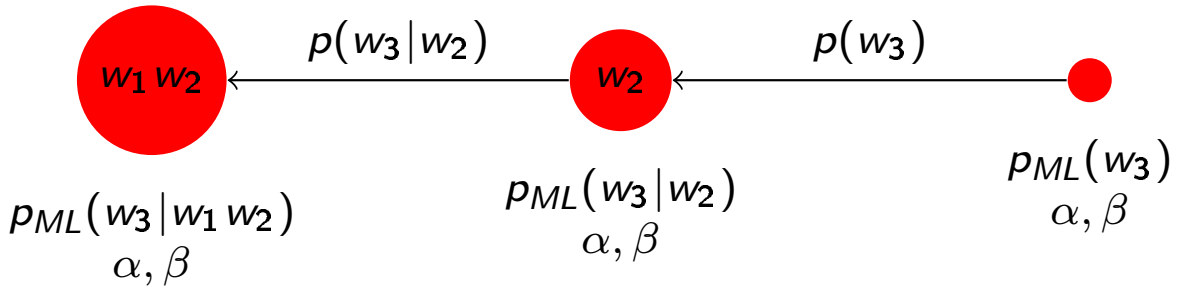
$$p_{BO}(w_n|\mathbf{w}_{1,n-1}) = \begin{cases} \alpha(\mathbf{w}_{1,n-1})\dfrac{c(\mathbf{w}_{1,n-1}w_n)}{c(\mathbf{w}_{1,n-1}\bullet)}, & c(\mathbf{w}_{1,n-1}w_n) > 0, \\ \beta(\mathbf{w}_{1,n-1})p_{BO}(w_n|\mathbf{w}_{2,n-1}), & c(\mathbf{w}_{1,n-1}w_n) = 0 \end{cases}$$

- Схематически это можно изобразить как $(n = 2)$



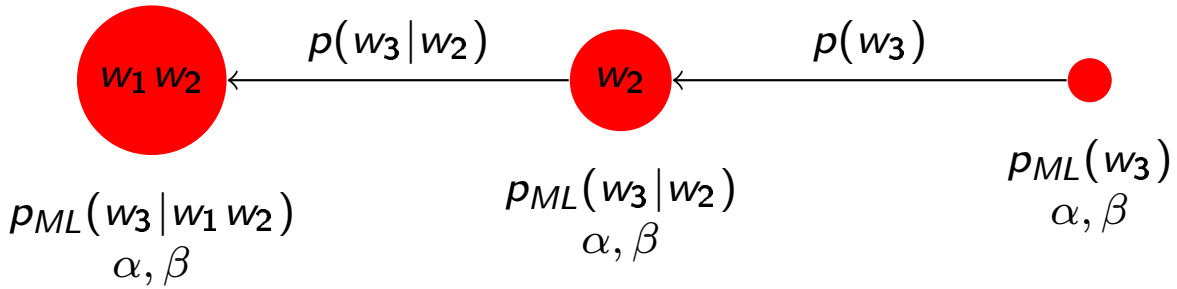
Факторные энграммные модели

Обычная энграммная модель ($n = 2$):

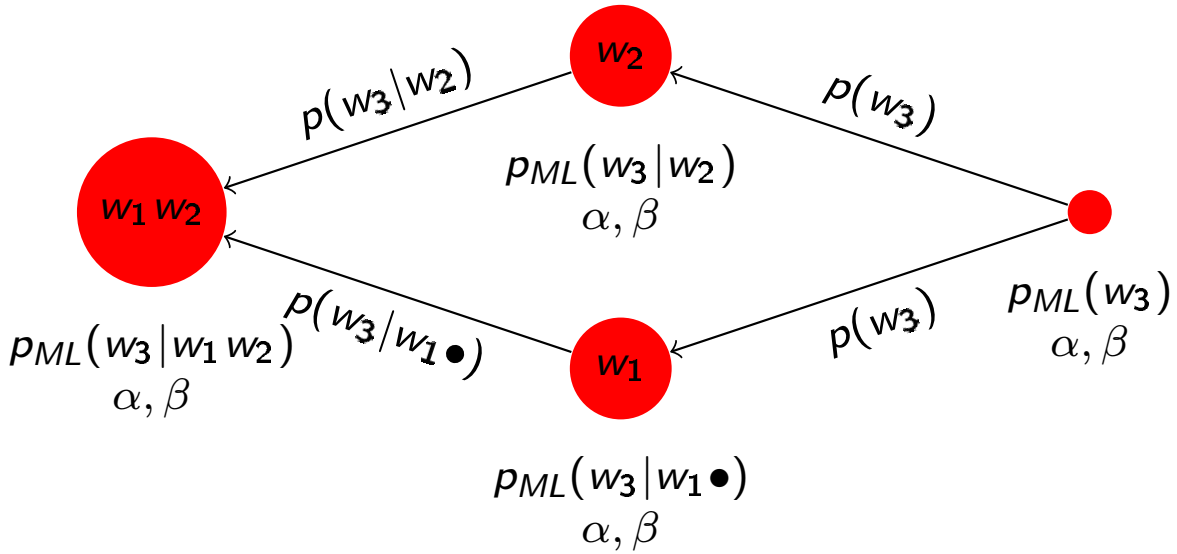


Факторные энграммные модели

Обычная энграммная модель ($n = 2$):

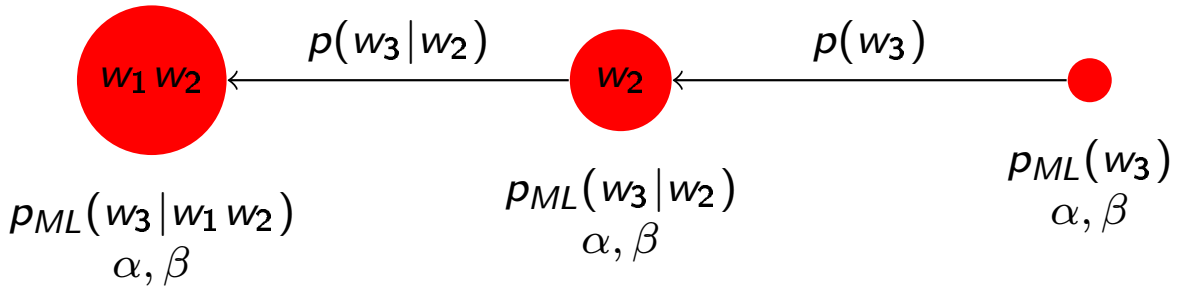


Разрешим вершине иметь несколько предков

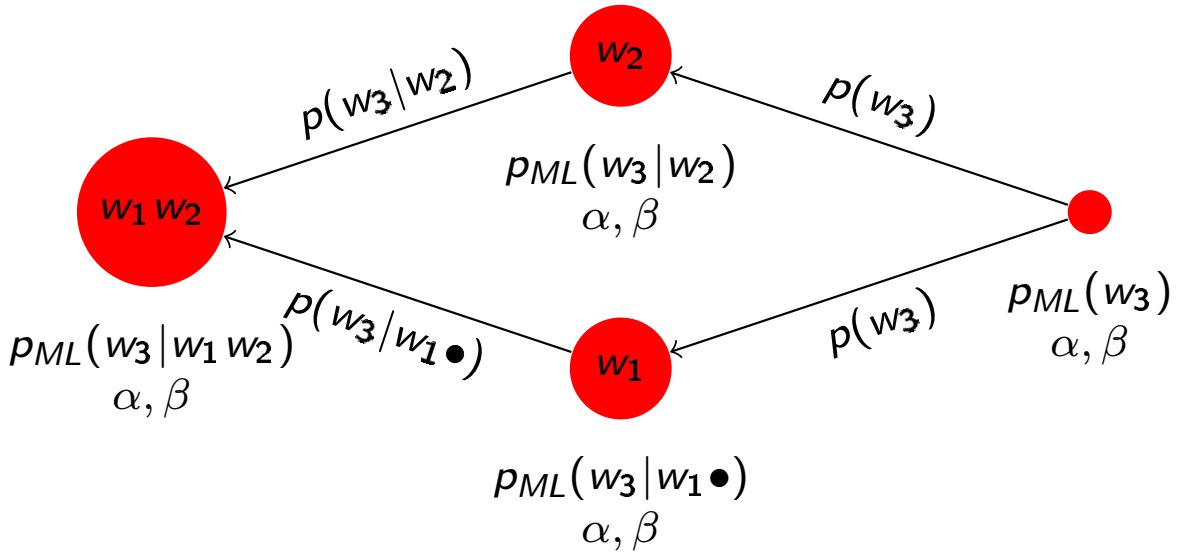


Факторные энграммные модели

Обычная энграммная модель ($n = 2$):

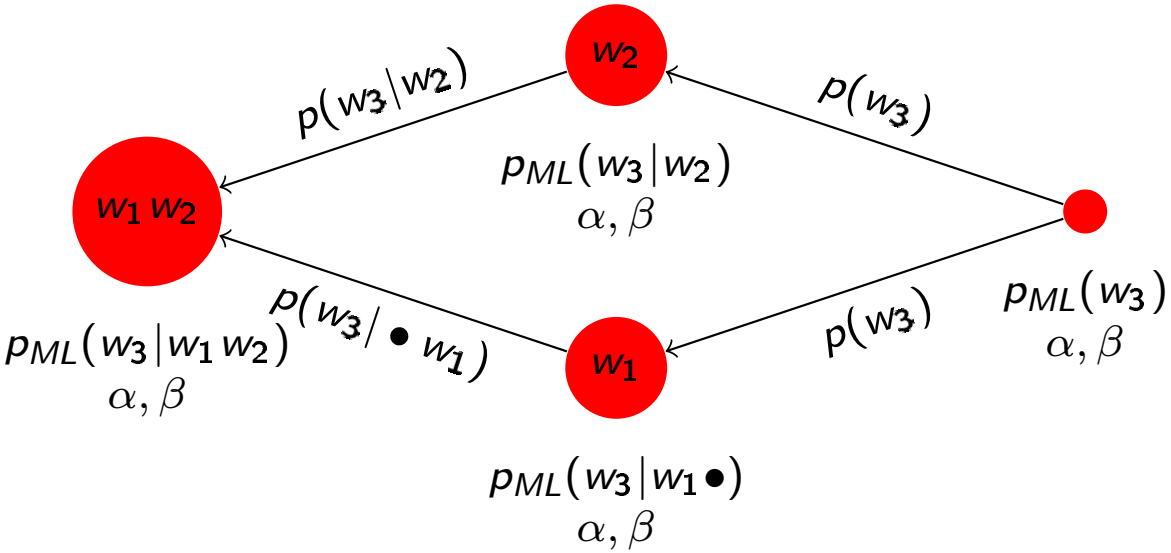


Разрешим вершине иметь несколько предков



Как описать математически?

Факторные энграммные модели



В этом случае $p(w_3|w_1 w_2)$ задаётся формулой

$$p(w_3|w_1 w_2) = \begin{cases} \alpha(\mathbf{w}_{1,2})\dfrac{c(\mathbf{w}_{1,2} w_3)}{c(\mathbf{w}_{1,2}\bullet)}, & c(\mathbf{w}_{1,2} w_n) > 0, \\ \beta(w_1, w_2) g(p_{BO}(w_3|w_2), p_{BO}(w_3|w_1\bullet)), & \text{иначе} \end{cases}$$

Факторные энграммные модели

- Общая формула:

$$p(w_3|w_1 w_2) = \begin{cases} \alpha(\mathbf{w}_{1,2})\dfrac{c(\mathbf{w}_{1,2}w_3)}{c(\mathbf{w}_{1,2}\bullet)}, & c(\mathbf{w}_{1,2}w_n) > 0, \\ \beta(w_1, w_2)g(p_{BO}(w_3|w_2), p_{BO}(w_3|w_1\bullet)), & \text{иначе} \end{cases}$$

Факторные энграммные модели

- Общая формула:

$$p(w_3|w_1 w_2) = \begin{cases} \alpha(\mathbf{w}_{1,2})\dfrac{c(\mathbf{w}_{1,2}w_3)}{c(\mathbf{w}_{1,2}\bullet)}, & c(\mathbf{w}_{1,2}w_n) > 0, \\ \beta(w_1, w_2)g(p_{BO}(w_3|w_2), p_{BO}(w_3|w_1\bullet)), & \text{иначе} \end{cases}$$

- $\alpha(\mathbf{w}_{1,2})$ — сглаживающий множитель (как в обычных методах),

Факторные энграммные модели

- Общая формула:

$$p(w_3|w_1 w_2) = \begin{cases} \alpha(\mathbf{w}_{1,2})\dfrac{c(\mathbf{w}_{1,2}w_3)}{c(\mathbf{w}_{1,2}\bullet)}, & c(\mathbf{w}_{1,2}w_n) > 0, \\ \beta(w_1, w_2)g(p_{BO}(w_3|w_2), p_{BO}(w_3|w_1\bullet)), & \text{иначе} \end{cases}$$

- $\alpha(\mathbf{w}_{1,2})$ — сглаживающий множитель (как в обычных методах),
- $\beta(w_1, w_2)$ — поправочный множитель (чтобы сумма вероятностей равнялась 1)

Факторные энграммные модели

- Общая формула:

$$p(w_3|w_1 w_2) = \begin{cases} \alpha(\mathbf{w}_{1,2})\dfrac{c(\mathbf{w}_{1,2} w_3)}{c(\mathbf{w}_{1,2}\bullet)}, & c(\mathbf{w}_{1,2} w_n) > 0, \\ \beta(w_1, w_2) g(p_{BO}(w_3|w_2), p_{BO}(w_3|w_1\bullet)), & \text{иначе} \end{cases}$$

- $\alpha(\mathbf{w}_{1,2})$ — сглаживающий множитель (как в обычных методах),
- $\beta(w_1, w_2)$ — поправочный множитель (чтобы сумма вероятностей равнялась 1)
- g — агрегирующая функция (максимум, произведение, взвешенная сумма...)

Факторные энграммные модели

- С помощью факторов можно задавать зависимость от произвольного из последних k слов.

Факторные энграммные модели

- С помощью факторов можно задавать зависимость от произвольного из последних k слов.
- Однако факторы необязательно соответствуют отдельным словам.

Факторные энграммные модели

- С помощью факторов можно задавать зависимость от произвольного из последних k слов.
- Однако факторы необязательно соответствуют отдельным словам.
- Ими можно задавать морфологические категории или основы (для случая слов), фонетические классы (для фонем), часть речи или отдельные граммемы (для морфологических меток).

Факторные энграммные модели

- С помощью факторов можно задавать зависимость от произвольного из последних k слов.
- Однако факторы необязательно соответствуют отдельным словам.
- Ими можно задавать морфологические категории или основы (для случая слов), фонетические классы (для фонем), часть речи или отдельные граммемы (для морфологических меток).
- Например, следующая биграммная модель сначала смотрит на слово w_1, потом на метку t_1, потом на часть речи c_1.