Скрытые марковские модели

Сергей Николенко

образование

Академия больших данных MADE — VK

23 марта 2022 г.

Random facts:

- 23 марта 1775 г. юрист и фермер Патрик Генри на втором вирджинском совете произнёс знаменитую речь, где сказал: «Give me liberty, or give me death!»; впрочем, он был антифедералистом и говорил конкретно о Вирджинии
- 23 марта 1801 г. (в ночь на 24) граф Пален за ужином сказал: «Напоминаю, господа, чтобы съесть яичницу — нужно сначала разбить яйца»; затем заговорщики проникли в Михайловский замок, в 0:30 ворвались в спальню императора; достоверно не известно, подписал ли Павел отречение или затеял драку, но в любом случае золотая табакерка Николая Зубова находится в собрании Государственного Эрмитажа
- 23 марта 1857 г. в E.V. Haughwout Building в Нью-Йорке был установлен первый лифт Элиши Отиса; сам Отис не дожил до эпохи небоскрёбов, умерев от дифтерии в 1861 году
- 23 марта 1909 г. Теодор Рузвельт, незадолго до того принципиально отказавшийся баллотироваться на третий срок, отправился на сафари, спонсированное Смитсоновским институтом и Национальным географическим обществом; Рузвельт с коллегами убили около 11400 животных, в том числе шесть белых носорогов
- 23 марта 2001 г. орбитальная станция «Мир» была затоплена в южной части Тихого океана, вблизи островов Фиджи

Рейтинг-системы, обучающиеся MM-алгоритмами

Частичные сравнения

- Рейтинг-система — это модель, которая ранжирует участников (игроков) в единый линейный порядок по данным сравнений небольших подмножеств этих игроков (турниров).
- Более того, результаты турниров зашумлены (отчасти случайны).
- Соответственно, и применяются они в таких ситуациях (пример: контекстная реклама в Bing).

Рейтинг Эло

- Первая известная рейтинг-система, основанная на байесовском подходе — это рейтинг Эло.
- Суть модели:
  - сила игры шахматиста в одной партии — случайная величина;
  - *рейтинг* — это ожидание этой величины; мы пытаемся оценить это ожидание;
  - исходная модель Эло — нормальное распределение силы игры вокруг рейтинга.

Рейтинг Эло

- Значит, сила игры в конкретной партии распределена как

$$p(x) = \mathcal{N}(x; s, \beta) = \frac{1}{\beta\sqrt{2\pi}} e^{-\frac{1}{2\beta^2}(x-s)^2}.$$

- Сила игры задаётся двумя параметрами: средним s (собственно рейтингом) и дисперсией β^2.
- Эло предположил, что дисперсия β^2 постоянна (и даже от игрока не зависит), а среднее — это как раз рейтинг, который мы пытаемся оценить.

Рейтинг Эло

- Значит, математически говоря, мы ищем

$$\arg\max_{s,\beta^2} p(s, \beta^2 \mid D) = \arg\max_{s,\beta^2} \frac{p(D \mid s, \beta^2)p(s, \beta^2)}{p(D)} = \\ = \arg\max_{s,\beta^2} p(D \mid s, \beta^2)p(s, \beta^2).$$

- Как мы знаем, нормальное распределение является самосопряжённым, поэтому если сила игры нормально распределена вокруг рейтинга, то логично взять нормальное распределение как априорное для рейтинга.
- Таким образом, рейтинг игрока складывается из двух чисел: его среднего значения μ и дисперсии σ^2.
- Значение μ отображается в таблице рейтингов, а σ^2 показывает, насколько достоверна имеющаяся оценка.

Рейтинг Эло

- Предположим, что встречаются два игрока с некоторыми априорными распределениями на рейтинги $\mathcal{N}(s_1; \mu_1, \sigma_1^2)$ и $\mathcal{N}(s_2; \mu_2, \sigma_2^2)$.
- Тогда сила игры каждого из них в этой конкретной партии имеет распределение

$$p(x \mid \mu, \sigma) = \int_{-\infty}^{\infty} p(x \mid s) p(s \mid \mu, \sigma) ds =$$

$$= \int_{-\infty}^{\infty} \mathcal{N}(x; s, \beta) \mathcal{N}(s; \mu, \sigma) ds =$$

$$= \int_{-\infty}^{\infty} \frac{1}{\beta\sqrt{2\pi}} e^{-\frac{1}{2\beta^2}(x-s)^2} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2\sigma^2}(s-\mu)^2} ds = \mathcal{N}(x; \mu_x, \sigma_x).$$

то есть мы снова приходим к нормальному распределению, но с другими параметрами.

Рейтинг Эло

- Задача обучения заключается в том, чтобы после новой партии принять во внимание её результат и пересчитать рейтинги.
- Эло разработал специальные аппроксимации и очень простые алгоритмы для этого случая (через «ожидаемые очки в турнире»), чтобы каждый шахматист мог сам на калькуляторе свой рейтинг посчитать, но они нас сейчас не очень интересуют.
- Сейчас есть обобщения рейтинга Эло, мы поговорим о них позже.

Модели Брэдли–Терри

- Другой подход к рейтинг–системам — модели Брэдли–Терри (Bradley–Terry).
- Модель предполагает, что для участников $1, \ldots, n$ можно подобрать такие рейтинги γ_i, $i = 1..n$, что вероятность победы участника i над участником j равна

$$p(i \text{ побеждает } j) = \frac{\gamma_i}{\gamma_i + \gamma_j}.$$

- Основная задача заключается в том, чтобы найти $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_m)$ максимального правдоподобия из имеющихся данных D.

Модели Брэдли–Терри

- Если принять априорное распределение равномерным, можно просто максимизировать правдоподобие

$$p(D|\boldsymbol{\gamma}) = \prod_{i=1}^{m} \prod_{j=1}^{m} \left(\frac{\gamma_i}{\gamma_i + \gamma_j} \right)^{w_{ij}},$$

  где w_{ij} — то, сколько раз x_i обыграл x_j при их попарном сравнении ($w_{ii} = 0$ по определению).

- Взяв логарифм, будем максимизировать

$$l(\boldsymbol{\gamma}) = \sum_{i=1}^{m} \sum_{j=1}^{m} \left(w_{ij} \log \gamma_i - w_{ij} \log(\gamma_i + \gamma_j) \right).$$

Модели Брэдли–Терри

- Максимизировать будем классическим MM-алгоритмом (minorization–maximization), фактически вариационным приближением.
- Заметим, что

$$1 + \log \frac{x}{y} - \frac{x}{y} \leq 0.$$

Упражнение. Докажите это.

Модели Брэдли–Терри

- Рассмотрим вспомогательную функцию

$$Q(\boldsymbol{\gamma}, \boldsymbol{\gamma}^{(k)}) = \sum_{i,j} w_{ij} \left[\log \gamma_i - \frac{\gamma_i + \gamma_j}{\gamma_i^{(k)} + \gamma_j^{(k)}} - \log \left(\gamma_i^{(k)} + \gamma_j^{(k)} \right) + 1 \right].$$

Упражнение. Используя предыдущее неравенство, докажите, что $Q(\boldsymbol{\gamma}, \boldsymbol{\gamma}^{(k)}) \leq l(\boldsymbol{\gamma})$.

Модели Брэдли–Терри

- Чтобы найти $\max_{\boldsymbol{\gamma}} Q\left(\boldsymbol{\gamma}, \boldsymbol{\gamma}^{(k)}\right)$, можно просто взять производные $\frac{\partial Q}{\partial \gamma_l}$:

$$\frac{\partial Q}{\partial \gamma_l} = \sum_{i,j} w_{ij} \left[\frac{\delta_{il}}{\gamma_i} - \frac{\delta_{il} + \delta_{jl}}{\gamma_i^{(k)} + \gamma_j^{(k)}} \right] =$$

$$= \frac{1}{\gamma_l} \sum_j w_{lj} - \sum_j \frac{w_{lj}}{\gamma_i^{(k)} + \gamma_j^{(k)}} - \sum_i \frac{w_{il}}{\gamma_i^{(k)} + \gamma_j^{(k)}}.$$

Модели Брэдли–Терри

- Если w_l — общее количество побед игрока l ($w_l = \sum_j w_{lj}$), и N_{ij} — количество встреч между игроками i и j ($N_{ij} = w_{ij} + w_{ji}$), получаем

$$\frac{w_l}{\gamma_l} - \sum_j \frac{N_{lj}}{\gamma_i^{(k)} + \gamma_j^{(k)}} = 0.$$

- В результате правило пересчёта на одной итерации выглядит так:

$$\gamma_l^{(k+1)} := w_l \left[\sum_j \frac{N_{lj}}{\gamma_i^{(k)} + \gamma_j^{(k)}} \right]^{-1}.$$

Модели Брэдли–Терри

- Получили алгоритм оценки рейтингов. Правда, он пока работает только для ситуации, когда игроки встречаются один на один и выигрывают или проигрывают.
- Но даже в шахматах бывают ничьи. Как их учесть?

Модели Брэдли–Терри

- Если возможны ничьи, их вероятность можно описать дополнительным параметром $\theta > 1$, и это приводит к модели, в которой

$$p(i \text{ побеждает } j) = \frac{\gamma_i}{\gamma_i + \theta\gamma_j},$$
$$p(j \text{ побеждает } i) = \frac{\gamma_j}{\theta\gamma_i + \gamma_j},$$
$$p(i \text{ и } j \text{ играют вничью}) = \frac{(\theta^2 - 1)\gamma_i\gamma_j}{(\gamma_i + \theta\gamma_j)(\theta\gamma_i + \gamma_j)}.$$

Модели Брэдли–Терри

- Аналогично можно вводить другие обобщения.
- Например, если результат может зависеть от порядка элементов в паре (скажем, команды проводят «домашние» матчи и «гостевые»), можно ввести дополнительный параметр θ, характеризующий, насколько большое преимущество дают «родные стены», и рассмотреть модель с

$$p(i \text{ побеждает } j) = \begin{cases} \frac{\theta\gamma_i}{\theta\gamma_i+\gamma_j}, & \text{если } i \text{ играет дома,} \\ \frac{\gamma_j}{\theta\gamma_i+\gamma_j}, & \text{если } j \text{ играет дома.} \end{cases}$$

Модели Брэдли–Терри

- Можно даже обобщить на случай, когда в одном турнире встречаются несколько игроков: пусть перестановка π подмножества игроков $A = \{1, \ldots, k\}$ (результат турнира) имеет вероятность

$$p_A(\pi) = \prod_{i=1}^{k} \frac{\gamma_{\pi(i)}}{\gamma_{\pi(i)} + \gamma_{\pi(i+1)} + \ldots + \gamma_{\pi(k)}}.$$

Модели Брэдли–Терри

- Можно показать, что такая модель эквивалентна весьма естественной «аксиоме Люса»: для любой модели, в которой вероятности игроков обыграть друг друга в любой паре не равны нулю, для любых подмножеств игроков $A \subset B$ и любого игрока $i \in A$

$$p_B(i \text{ побеждает}) =$$
$$= p_A(i \text{ побеждает}) p_B(\text{побеждает кто-то из множества } A).$$

- Но для крупных турниров это перестаёт работать; и совсем трудно что-то осмысленное сделать, если игроки соревнуются не поодиночке, а в командах.

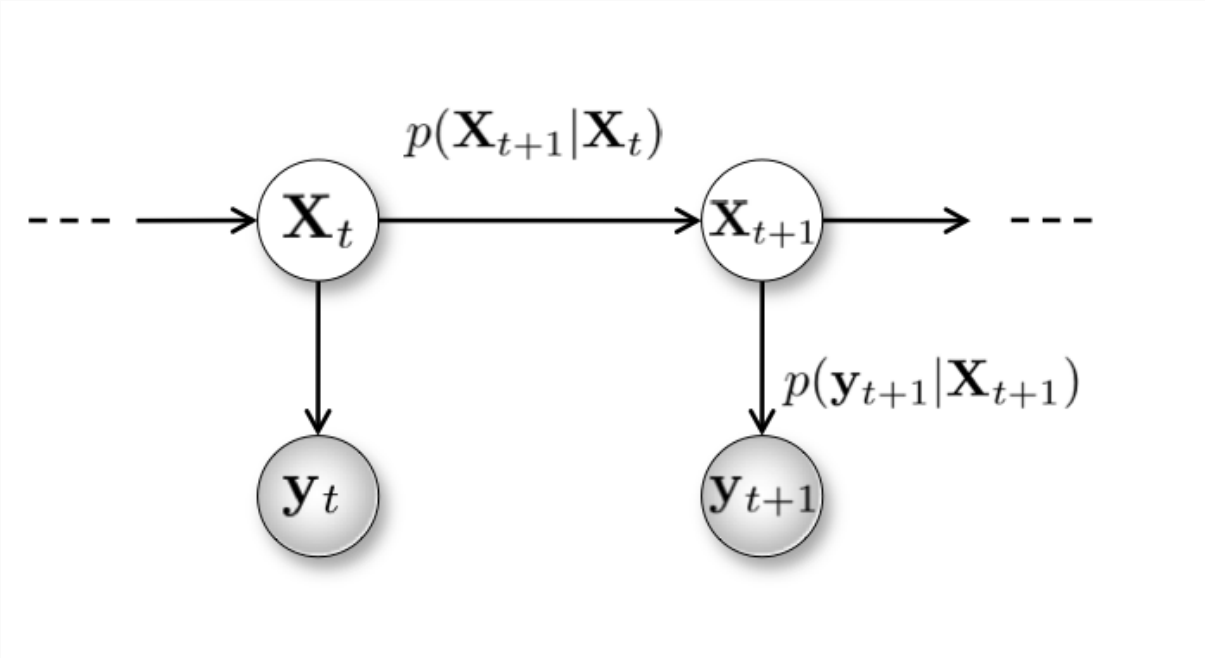
Скрытые марковские модели: основное

Марковские цепи

- Марковская цепь задаётся начальным распределением вероятностей $p^0(x)$ и вероятностями перехода $T(x'; x)$.
- $T(x'; x)$ — это распределение следующего элемента цепи в зависимости от следующего; распределение на $(t+1)$–м шаге равно
$$p^{t+1}(x') = \int T(x'; x) p^t(x) dx.$$
- В дискретном случае $T(x'; x)$ — это матрица вероятностей $p(x' = i | x = j)$.

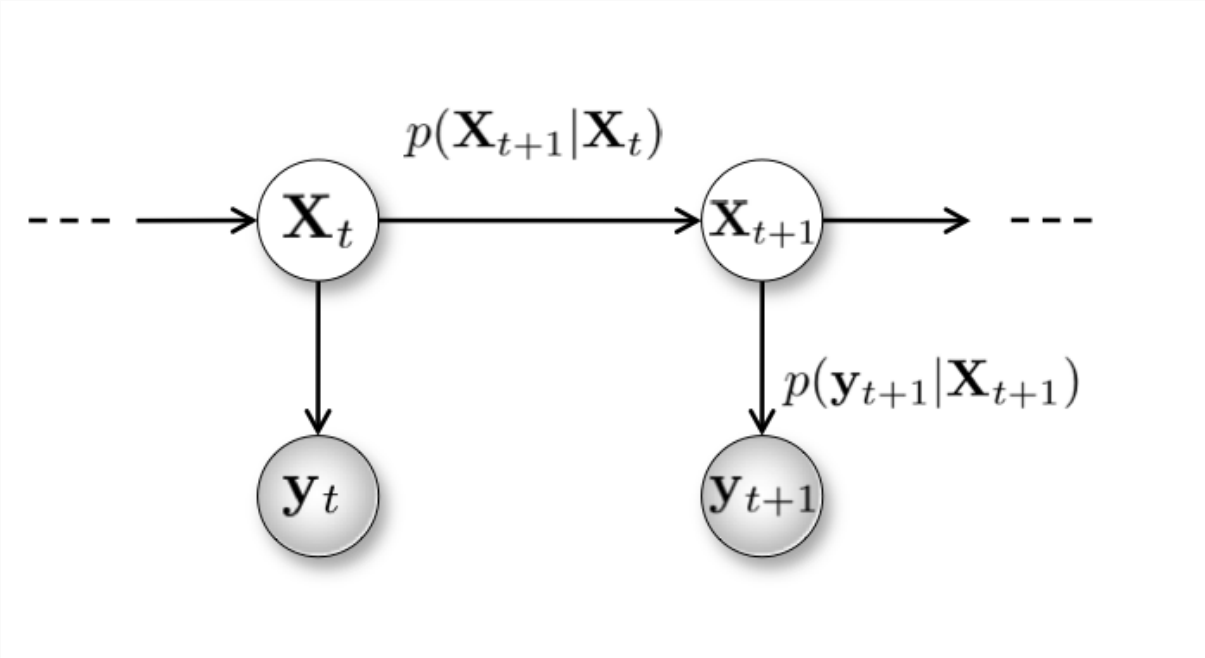
Дискретные марковские цепи

- Мы будем находиться в дискретном случае.
- Марковская модель — это когда мы можем наблюдать какие-то функции от марковского процесса.



Дискретные марковские цепи

- Здесь $x(t)$ — сам процесс (модель), а $y(t)$ — то, что мы наблюдаем.
- Задача — определить скрытые параметры процесса.



Дискретные марковские цепи

- Главное свойство — следующее состояние не зависит от истории, только от предыдущего состояния.

$$p(x(t) = x_j | x(t-1) = x_{j_{t-1}}, \ldots, x(1) = x_{j_1}) = \\ = p(x(t) = x_j | x(t-1) = x_{j_{t-1}}).$$

- Более того, эти вероятности $a_{ij} = p(x(t) = x_j | x(t-1) = x_i)$ ещё и от времени t не зависят.
- Эти вероятности и составляют матрицу перехода $A = (a_{ij})$.

Вероятности перехода

- Естественные свойства:
- $a_{ij} G_{\mathrm{enh}} 0$.
- $\sum_j a_{ij} = 1$.

Прямая задача

- Естественная задача: с какой вероятностью выпадет та или иная последовательность событий?
- Т.е. найти нужно для последовательности $Q = q_{i_1} \dots q_{i_k}$

$$p(Q|\text{модель}) = p(q_{i_1})p(q_{i_2}|q_{i_1}) \dots p(q_{i_k}|q_{i_{k-1}}).$$

- Казалось бы, это тривиально.
- Что же сложного в реальных задачах?

Скрытые марковские модели

- А сложно то, что никто нам не скажет, что модель должна быть именно такой.
- И, кроме того, мы обычно наблюдаем не $x(t)$, т.е. реальные состояния модели, а $y(t)$, т.е. некоторую функцию от них (данные).
- Пример: распознавание речи.

Задачи скрытых марковских моделей

- Первая: найти вероятность последовательности наблюдений в данной модели.
- Вторая: найти «оптимальную» последовательность состояний при условии данной модели и данной последовательности наблюдений.
- Третья: найти наиболее правдоподобную модель (параметры модели).

Состояния и наблюдаемые

- $X = \{x_1, \ldots, x_n\}$ — множество состояний.
- $V = \{v_1, \ldots, v_m\}$ — алфавит, из которого мы выбираем наблюдаемые y (множество значений y).
- q_t — состояние во время t, y_t — наблюдаемая во время t.

Распределения

- $a_{ij} = p(q_{t+1} = x_j | q_t = x_i)$ — вероятность перехода из i в j.
- $b_j(k) = p(v_k | x_j)$ — вероятность получить данные v_k в состоянии j.
- Начальное распределение $\pi = \{\pi_j\}$, $\pi_j = p(q_1 = x_j)$.
- Данные будем обозначать через $D = d_1 \dots d_T$ (последовательность наблюдаемых, d_i принимают значения из V).

Комментарий

- Проще говоря, вот как работает HMM (hidden Markov model).
- Выберем начальное состояние x_1 по распределению π.
- По t от 1 до T:
  - Выберем наблюдаемую d_t по распределению $p(v_k|x_j)$.
  - Выберем следующее состояние по распределению $p(q_{t+1} = x_j|q_t = x_i)$.
- Таким алгоритмом можно выбрать случайную последовательность наблюдаемых.

Задачи

- Теперь можно формализовать постановку задач.
- Первая задача: по данной модели $\lambda = (A, B, \pi)$ и последовательности D найти $p(D|\lambda)$. Фактически, это нужно для того, чтобы оценить, насколько хорошо модель подходит к данным.
- Вторая задача: по данной модели λ и последовательности D найти «оптимальную» последовательность состояний $Q = q_1 \dots q_T$. Как и раньше, будет два решения: «побитовое» и общее.
- Третья задача: оптимизировать параметры модели $\lambda = (A, B, \pi)$ так, чтобы максимизировать $p(D|\lambda)$ при данном D (найти модель максимального правдоподобия). Эта задача — главная, в ней и заключается обучение скрытых марковских моделей.

Постановка первой задачи

- Формально, первая задача выглядит так. Нужно найти

$$p(D|\lambda) = \sum_{Q} p(D|Q,\lambda)p(D|\lambda) =$$

$$= \sum_{q_1,\ldots,q_T} b_{q_1}(d_1)\ldots b_{q_T}(d_T)\pi_{q_1}a_{q_1q_2}\ldots a_{q_{T-1}q_T}.$$

- Ничего не напоминает?

Суть решения первой задачи

- Правильно, это такая же задача маргинализации, как мы решаем всё время.
- Мы воспользуемся так называемой forward–backward procedure, по сути — динамическим программированием на решётке.
- Будем последовательно вычислять промежуточные величины вида

$$\alpha_t(i) = p(d_1 \dots d_t, q_t = x_i | \lambda),$$

т.е. искомые вероятности, но ещё с учётом текущего состояния.

Решение первой задачи

- Инициализируем $\alpha_1(i) = \pi_i b_i(d_1)$.
- Шаг индукции:

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{n} \alpha_t(i) a_{ij}\right] b_j(d_{t+1}).$$

- После того как дойдём до шага T, подсчитаем то, что нам нужно:

$$p(D|\lambda) = \sum_{i=1}^{n} \alpha_T(i).$$

- Фактически, это только прямой проход, обратный нам здесь не понадобился.
- Что вычислял бы обратный проход?

Обратный проход

- Он вычислял бы условные вероятности $\beta_t(i) = p(d_{t+1} \dots d_T | q_t = x_i, \lambda)$.
- Их можно вычислить, проинициализировав $\beta_T(i) = 1$, а затем по индукции:

$$\beta_t(i) = \sum_{j=1}^{n} a_{ij} b_j(d_{t+1}) \beta_{t+1}(j).$$

- Это нам пригодится чуть позже, при решении второй и третьей задачи.

Два варианта второй задачи

- Как мы уже упоминали, возможны два варианта.
- Первый: решать «побитово», отвечая на вопрос «какое наиболее вероятное состояние во время j?».
- Второй: решать задачу «какая наиболее вероятная последовательность состояний?».

Побитовое решение

- Рассмотрим вспомогательные переменные

$$\gamma_t(i) = p(q_t = x_i | D, \lambda).$$

- Наша задача – найти

$$q_t = \arg\max_{1 \le i \le n} \gamma_t(i), \quad 1 \le t \le T.$$

- Как это сделать?

Побитовое решение

- Выражаем через α и β:

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{p(D|\lambda)} = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{n} \alpha_t(i)\beta_t(i)}.$$

- На знаменатель можно не обращать внимания — нам нужен $\arg\max$.

Решение относительно последовательности

- Чтобы ответить на вопрос о наиболее вероятной последовательности, мы будем использовать так называемый *алгоритм Витерби* (то есть, по сути, то же самое динамическое программирование).
- Наши вспомогательные переменные — это

$$\delta_t(i) = \max_{q_1,\ldots,q_{t-1}} p\left(q_1 q_2 \ldots q_t = x_i, d_1 d_2 \ldots d_t | \lambda\right).$$

Решение относительно последовательности

- Т.е. $\delta_t(i)$ — максимальная вероятность достичь состояния x_i на шаге t среди всех путей с заданными наблюдаемыми.
- По индукции:

$$\delta_{t+1}(j) = \left[\max_i \delta_t(i) a_{ij}\right] b_j(d_{t+1}).$$

- И надо ещё запоминать аргументы, а не только значения; для этого будет массив $\psi_t(j)$.

Решение относительно последовательности: алгоритм

- Проинициализируем $\delta_1(i) = \pi_i b_i(d_1)$, $\psi_1(i) = []$.
- Индукция:

$$\delta_t(j) = \max_{1 \le i \le n} \left[\delta_{t-1}(i) a_{ij}\right] b_j(d_t),$$

$$\psi_t(j) = \arg\max_{1 \le i \le n} \left[\delta_{t-1}(i) a_{ij}\right].$$

- Когда дойдём до шага T, финальный шаг:

$$p^* = \max_{1 \le i \le n} \delta_T(i), \qquad q_T^* = \arg\max_{1 \le i \le n} \delta_T(i).$$

- И вычислим последовательность: $q_t^* = \psi_{t+1}(q_{t+1}^*)$.

Общая суть третьей задачи

- Аналитически найти глобальный максимум $p(D|\lambda)$ у нас никак не получится.
- Зато мы рассмотрим итеративную процедуру (по сути — градиентный подъём), которая приведёт к локальному максимуму.
- Это называется алгоритм Баума–Велха (Baum–Welch algorithm). Он является на самом деле частным случаем алгоритма EM.

Вспомогательные переменные

- Теперь нашими вспомогательными переменными будут вероятности того, что мы во время t в состоянии x_i, а во время $t+1$ — в состоянии x_j:

$$\xi_t(i,j) = p(q_t = x_i, q_{t+1} = x_j | D, \lambda).$$

- Если переписать через уже знакомые переменные:

$$\xi_t(i,j) = \frac{\alpha_t(i) a_{ij} b_j(d_{t+1}) \beta_{t+1}(j)}{p(D|\lambda)} = \frac{\alpha_t(i) a_{ij} b_j(d_{t+1}) \beta_{t+1}(j)}{\sum_i \sum_j \alpha_t(i) a_{ij} b_j(d_{t+1}) \beta_{t+1}(j)}.$$

- Отметим также, что $\gamma_t(i) = \sum_j \xi_t(i,j)$.

Идея

- $\sum_t \gamma_t(i)$ — это ожидаемое количество переходов из состояния x_i, а $\sum_t \xi_t(i,j)$ — из x_i в x_j.
- Теперь на шаге M мы будем переоценивать вероятности:

$$\bar{\pi}_i = \text{ожидаемая частота в } x_i \text{ на шаге } 1 = \gamma_1(i),$$

$$\bar{a}_{ij} = \frac{\text{к-во переходов из } x_i \text{ в } x_j}{\text{к-во переходов из } x_i} = \frac{\sum_t \xi_t(i,j)}{\sum_t \gamma_t(i)}.$$

$$\bar{b}_j(k) = \frac{\text{к-во появлений в } x_i \text{ и наблюдений } v_k}{\text{к-во появлений в } x_i} = \frac{\sum_{t:d_t=v_k} \gamma_t(i)}{\sum_t \gamma_t(i)}.$$

- EM-алгоритм приведёт к цели: начать с $\lambda = (A, B, \pi)$, подсчитать $\bar{\lambda} = (\bar{A}, \bar{B}, \bar{\pi})$, снова пересчитать параметры и т.д.

Расстояние Кульбака–Лейблера

- Kullback–Leibler distance (divergence) — это информационно-теоретическая мера того, насколько далеки распределения друг от друга.

$$D_{KL}(p_1, p_2) = \sum_x p_1(x) \log \frac{p_1(x)}{p_2(x)}.$$

- Известно, что это расстояние всегда неотрицательно, равно нулю iff $p_1 \equiv p_2$.

Применительно к HMM

- Мы определим

$$p_1(Q) = \frac{p(Q, D|\lambda)}{p(D|\lambda)}, \quad p_2(Q) = \frac{p(Q, D|\lambda')}{p(D|\lambda')}.$$

- Тогда p_1 и p_2 — распределения, и расстояние Kullback–Leibler:

$$0 \leq D_{LK}(\lambda, \lambda') = \sum_Q \frac{p(Q, D|\lambda)}{p(D|\lambda)} \log \frac{p(Q, D|\lambda)p(D|\lambda')}{p(Q, D|\lambda')p(D|\lambda)} =$$

$$= \log \frac{p(D|\lambda')}{p(D|\lambda)} + \sum_Q \frac{p(Q, D|\lambda)}{p(D|\lambda)} \log \frac{p(Q, D|\lambda)}{p(Q, D|\lambda')}.$$

Вспомогательная функция

- Введём вспомогательную функцию

$$\mathrm{Q}(\lambda, \lambda') = \sum_{Q} p(Q|D, \lambda) \log p(Q|D, \lambda').$$

- Тогда из неравенства следует, что

$$\frac{\mathrm{Q}(\lambda, \lambda') - \mathrm{Q}(\lambda, \lambda)}{p(D|\lambda)} \leq \log \frac{p(D|\lambda')}{p(D|\lambda)}.$$

- Т.е., если $\mathrm{Q}(\lambda, \lambda') > \mathrm{Q}(\lambda, \lambda)$, то $p(D|\lambda') > p(D|\lambda)$.
- Т.е., если мы максимизируем $\mathrm{Q}(\lambda, \lambda')$ по λ', мы тем самым будем двигаться в нужную сторону.

Функция Q

- Нужно максимизировать $\mathbf{Q}(\lambda, \lambda')$. Перепишем:

$$\mathbf{Q}(\lambda, \lambda') = \sum_Q p(Q|D, \lambda) \log p(Q|D, \lambda') =$$

$$= \sum_Q p(Q|D, \lambda) \log \pi_{q_1} \prod_t a_{q_{t-1} q_t} b_{q_t}(d_t) =$$

$$= \sum_Q p(Q|D, \lambda) \log \pi_{q_1} + \sum_Q p(Q|D, \lambda) \sum_t \log a_{q_{t-1} q_t} b_{q_t}(d_t).$$

- Последнее выражение легко дифференцировать по a_{ij}, $b_i(k)$ и π_i, добавлять соответствующие множители Лагранжа и решать. Получится именно пересчёт по алгоритму Баума–Велха (проверьте!).

Специальные виды марковских моделей и расширения

Непрерывные плотности наблюдаемых

- У нас были дискретные наблюдаемые с вероятностями $B = (b_j(k))$.
- Но в реальной жизни всё сложнее: зачастую мы наблюдаем непрерывные сигналы, а не дискретные величины, и дискретизовать их или плохо, или неудобно.
- При этом саму цепь можно оставить дискретной, т.е. перейти к непрерывным $b_j(D)$.

Специальный вид плотности

- Не для всех плотностей найдены алгоритмы пересчёта (обобщения алгоритма Баума–Велха).
- Наиболее общий результат верен, когда $b_j(D)$ можно представить в виде

$$b_j(D) = \sum_{m=1}^{M} c_{jm} \mathcal{P}(D, \mu_{jm}, \sigma_{jm}),$$

  где c_{jm} — коэффициенты смеси ($\sum_m c_{jm} = 1$), а $\mathcal{P}$ — выпуклое распределение со средним μ и вариацией σ (гауссиан подойдёт).
- К счастью, такой конструкцией можно приблизить любое непрерывное распределение, поэтому это можно широко применять.

Вспомогательные переменные

- $\gamma_t(j, m)$ — вероятность быть в состоянии j во время t, причём за D отвечает m–й компонент смеси.
- Формально говоря,

$$\gamma_t(j, m) = \left[\frac{\alpha_t(j)\beta_t(j)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)}\right] \left[\frac{c_{jm}\mathcal{P}(d_t, \mu_{jm}, \sigma_{jm})}{\sum_{m=1}^{M} c_{jm}\mathcal{P}(d_t, \mu_{jm}, \sigma_{jm})}\right].$$

- Если $M = 1$, то это уже известные нам $\gamma_t(j)$.

Алгоритм для этого случая

- Нужно научиться пересчитывать $b_j(D)$, т.е. пересчитывать c_{jm}, μ_{jm} и σ_{jm}.
- Это делается так:

$$\bar{c}_{jm} = \frac{\sum_{t=1}^{T} \gamma_t(j,m)}{\sum_{t=1}^{T} \sum_{m=1}^{M} \gamma_t(j,m)},$$

$$\bar{\mu}_{jm} = \frac{\sum_{t=1}^{T} \gamma_t(j,m) \cdot d_t}{\sum_{t=1}^{T} \gamma_t(j,m)},$$

$$\bar{\sigma}_{jm} = \frac{\sum_{t=1}^{T} \gamma_t(j,m) \cdot (d_t - \mu_{jm})(d_t - \mu_{jm})^t}{\sum_{t=1}^{T} \gamma_t(j,m)}.$$

Продолжительность пребывания

- Как моделировать продолжительность пребывания в том или ином состоянии?
- В дискретном случае вероятность пробыть в состоянии i d шагов:

$$p_i(d) = a_{ii}^{d-1}(1 - a_{ii}).$$

- Однако для большинства физических сигналов такое экспоненциальное распределение не соответствует действительности. Мы бы хотели явно задавать плотность пребывания в данном состоянии.
- Т.е. вместо коэффициентов перехода в себя a_{ii} — явное задание распределения $p_i(d)$.

Вспомогательные переменные

- Введём переменные

$$\alpha_t(i) = p(d_1 \dots d_t, x_i \text{ заканчивается во время } t|\lambda).$$

- Всего за первые t шагов посещено r состояний $q_1 \dots q_r$, и мы там оставались $d_1, \dots, d_r$. Т.е. ограничения:

$$q_r = x_i, \qquad \sum_{s=1}^{r} d_s = t.$$

Вычисление $\alpha_t(i)$

- Тогда получается

$$\alpha_t(i) = \sum_q \sum_d \pi_{q_1} p_{q_1}(d_1) p(d_1 d_2 \dots d_{d_1} | q_1)$$
$$a_{q_1 q_2} p_{q_2}(d_2) p(d_{d_1+1} \dots d_{d_1+d_2} | q_2) \dots$$
$$\dots a_{q_{r-1} q_r} p_{q_r}(d_r) p(d_{d_1+\dots+d_{r-1}+1} \dots d_t | q_r).$$

Вычисление $\alpha_t(i)$

- По индукции

$$\alpha_t(j) = \sum_{i=1}^{n} \sum_{d=1}^{D} \alpha_{t-d}(j) a_{ij} p_j(d) \prod_{s=t-d+1}^{t} b_j(d_s),$$

  где D — максимальная остановка в любом из состояний.

- Тогда, как и раньше,

$$p(d|\lambda) = \sum_{i=1}^{n} \alpha_T(i).$$

Вспомогательные переменные

- Для пересчёта потребуются ещё три переменные:

$$\alpha_t^*(i) = p(d_1 \dots d_t, x_i \text{ начинается во время } t+1|\lambda),$$

$$\beta_t(i) = p(d_{t+1} \dots d_T|x_i \text{ заканчивается во время } t, \lambda),$$

$$\beta_t^*(i) = p(d_{t+1} \dots d_T|x_i \text{ начинается во время } t+1, \lambda).$$

Вспомогательные переменные

- Соотношения между ними:

$$\alpha_t^*(j) = \sum_{i=1}^{n} \alpha_t(i) a_{ij},$$

$$\alpha_t(i) = \sum_{d=1}^{D} \alpha_{t-d}^*(i) p_i(d) \prod_{s=t-d+1}^{t} b_i(d_s),$$

$$\beta_t(i) = \sum_{j=1}^{n} a_{ij} \beta_t^*(j),$$

$$\beta_t^*(i) = \sum_{d=1}^{D} \beta_{t+d}(i) p_i(d) \prod_{s=t+1}^{t+d} b_i(d_s).$$

Формулы пересчёта

- Приведём формулы пересчёта.
- π_i — просто вероятность того, что x_i был первым состоянием:

$$\hat{\pi}_i = \frac{\pi_i \beta_0^*(i)}{p(d|\lambda)}.$$

- a_{ij} — та же формула, что обычно, только вместе с α есть ещё и β, которая говорит, что новое состояние начинается на следующем шаге:

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T} \alpha_t(i) a_{ij} \beta_t^*(j)}{\sum_{k=1}^{n} \sum_{t=1}^{T} \alpha_t(i) a_{ik} \beta_t^*(k)}.$$

Формулы пересчёта

- $b_i(k)$ — отношение ожидания количества событий $d_t = v_k$ в состоянии x_i к ожиданию количества любого v_j в состоянии x_i:

$$\hat{b}_i(k) = \frac{\sum_{t=1, d_t=v_k}^{T} \left(\sum_{\tau<t} \alpha_\tau^*(i)\beta_\tau^*(i) - \sum_{\tau<t} \alpha_\tau(i)\beta_\tau(i) \right)}{\sum_{k=1}^{m} \sum_{t=1, d_t=v_k}^{T} \left(\sum_{\tau<t} \alpha_\tau^*(i)\beta_\tau^*(i) - \sum_{\tau<t} \alpha_\tau(i)\beta_\tau(i) \right)}.$$

- $p_i(d)$ — отношение ожидания количества раз, которые x_i случилось с продолжительностью d, к количеству раз, которые x_i вообще случалось:

$$\hat{p}_i(d) = \frac{\sum_{t=1}^{T} \alpha_t^*(i) p_i(d) \beta_{t+d}(i) \prod_{s=t+1}^{t+d} b_i(d_s)}{\sum_{d=1}^{D} \sum_{t=1}^{T} \alpha_t^*(i) p_i(d) \beta_{t+d}(i) \prod_{s=t+1}^{t+d} b_i(d_s)}.$$

За и против

- Такой подход очень полезен, когда $p_i(d)$ далеко от экспоненциального.
- Однако он сильно увеличивает вычислительную сложность (в D^2 раз).
- И, кроме того, становится гораздо больше параметров, т.е. нужно, вообще говоря, больше данных, чтобы эти параметры надёжно оценить.

Параметрическая продолжительность состояния

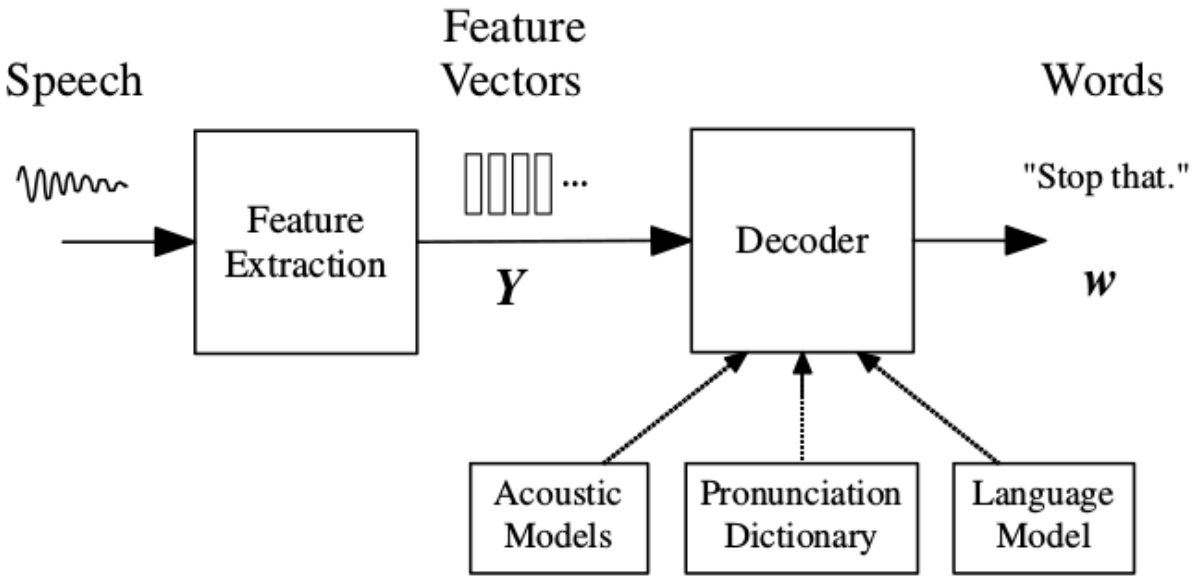
- Чтобы уменьшить количество параметров, можно иногда считать, что $p_i(d)$ — классическое распределение с не слишком большим количеством параметров.
- Например, $p_i(d)$ может быть равномерным, или нормальным ($p_i(d) = \mathcal{N}(d, \mu_i, \sigma_i^2)$), или гамма–распределением:

$$p_i(d) = \frac{\eta_i^{\nu_i} d^{\nu_i - 1} e^{-\eta_i d}}{\Gamma(\nu_i)}.$$

HMM для распознавания речи

Общая структура

- HMM – классический подход к распознаванию речи.
- Сейчас, правда, они уже в основном заменены глубокими нейронными сетями; но всё равно полезно проследить, как их можно применить.



От слов к фонемам

- Признаки как-то выделились, а потом слово w делится на фонемы $\mathbf{q} = q_1 \dots q_{|w|}$, и правдоподобие наблюдаемых признаков

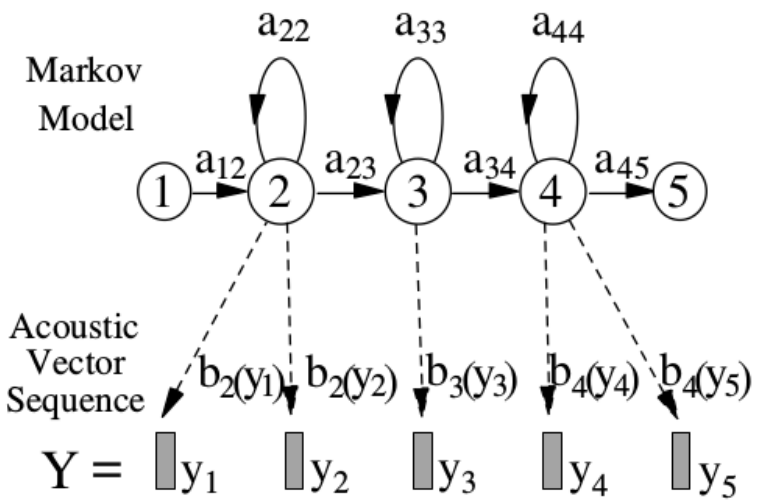
$$p(\mathbf{y} \mid \mathbf{w}) = \sum_{\mathbf{q}} p(\mathbf{y} \mid \mathbf{q}) p(\mathbf{q} \mid \mathbf{w}),$$

  сумма по возможным произношениям (маленькая сумма), а $\mathbf{w}$ – это все слова $w_1 \dots w_L$:

$$p(\mathbf{q} \mid \mathbf{w}) = \prod_{l=1}^{L} p(\mathbf{q}^{(w_l)} \mid w_l).$$

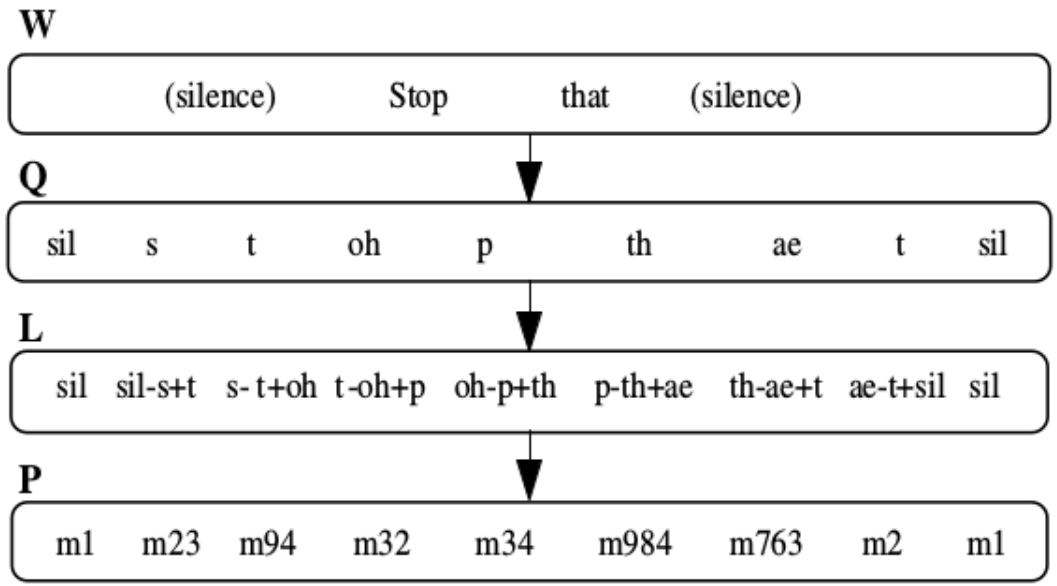
Акустическая модель

- Каждая фонема – это HMM с непрерывными наблюдаемыми $b_j(\mathbf{y}) = \mathcal{N}(\mathbf{y}; \mu^{(j)}, \Sigma^{(j)})$.
- На этом месте уже можно обучать просто всё сразу.



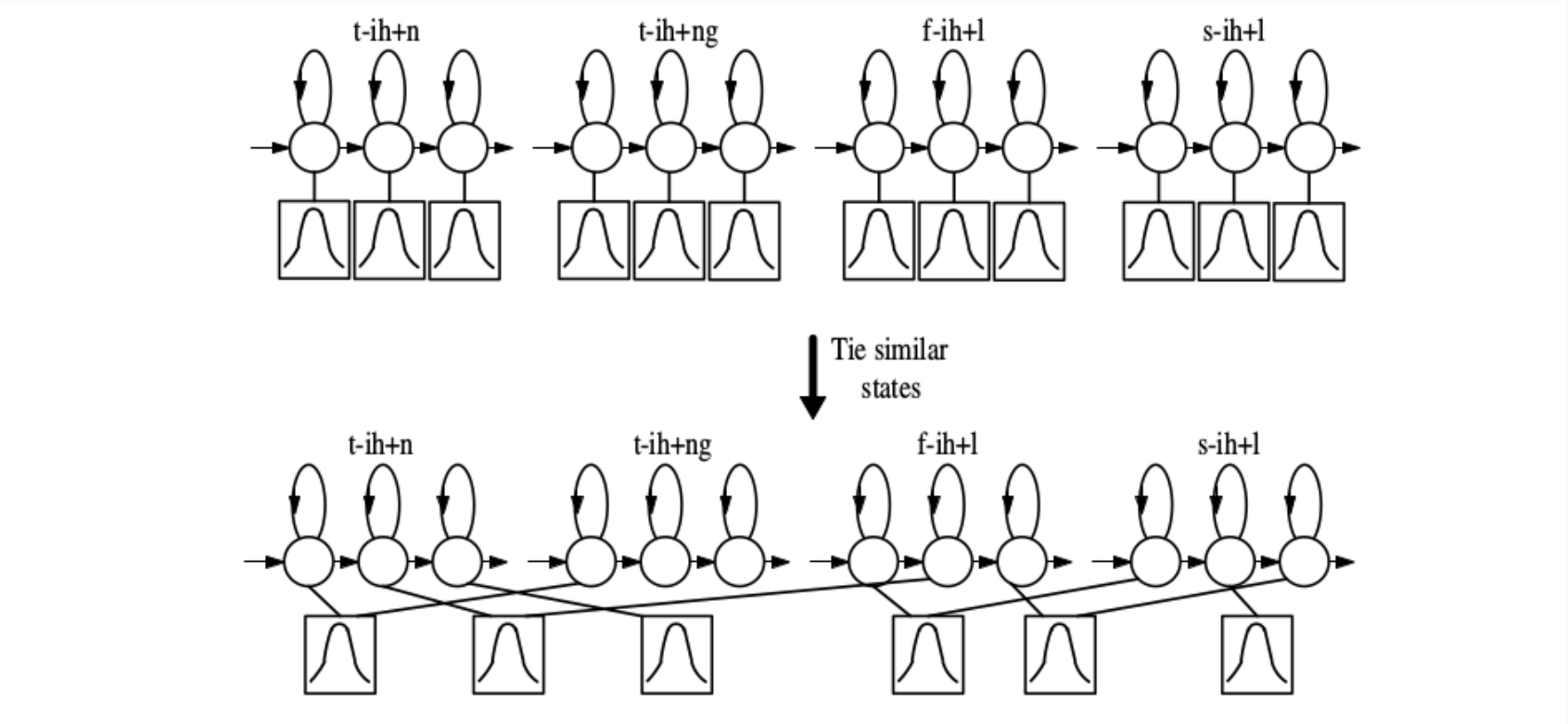
Акустическая модель

- Однако фонемы очень по-разному звучат в зависимости от контекста.
- Можно перейти к трифонам.



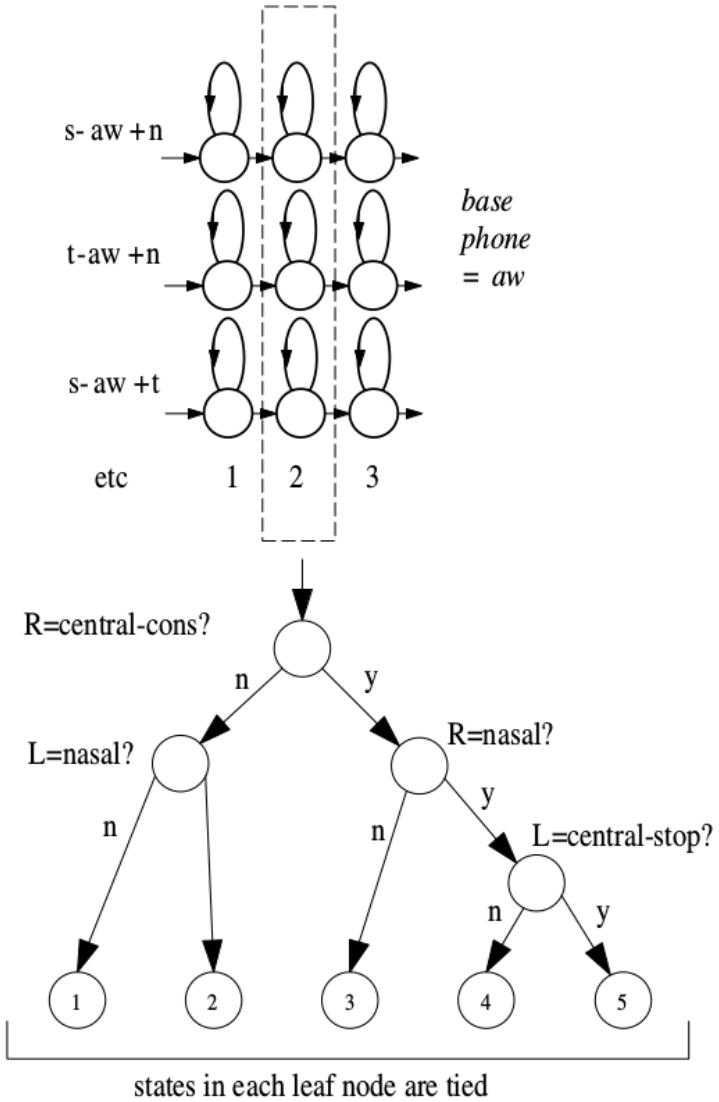
Акустическая модель

- Но их будет целых N^3, и лучше объединить похожие и связать их параметры:



Акустическая модель

- Это можно сделать просто силой мысли:



Языковая модель

- Но это только начало. Ещё нужна *языковая модель* – мы о многом просто догадываемся.
- То есть нужно априорное распределение

$$p(\mathbf{w}) = \prod_{l=1}^{L} p(w_l \mid w_1 \dots w_{l-1}).$$

- Классический подход – n-граммы для $n = 2..4$:

$$p(\mathbf{w}) = \prod_{l=1}^{L} p(w_l \mid w_{l-1} \dots w_{l-n+1}).$$

- Качество языковых моделей сравнивают в терминах их *перплексии* (perplexity)

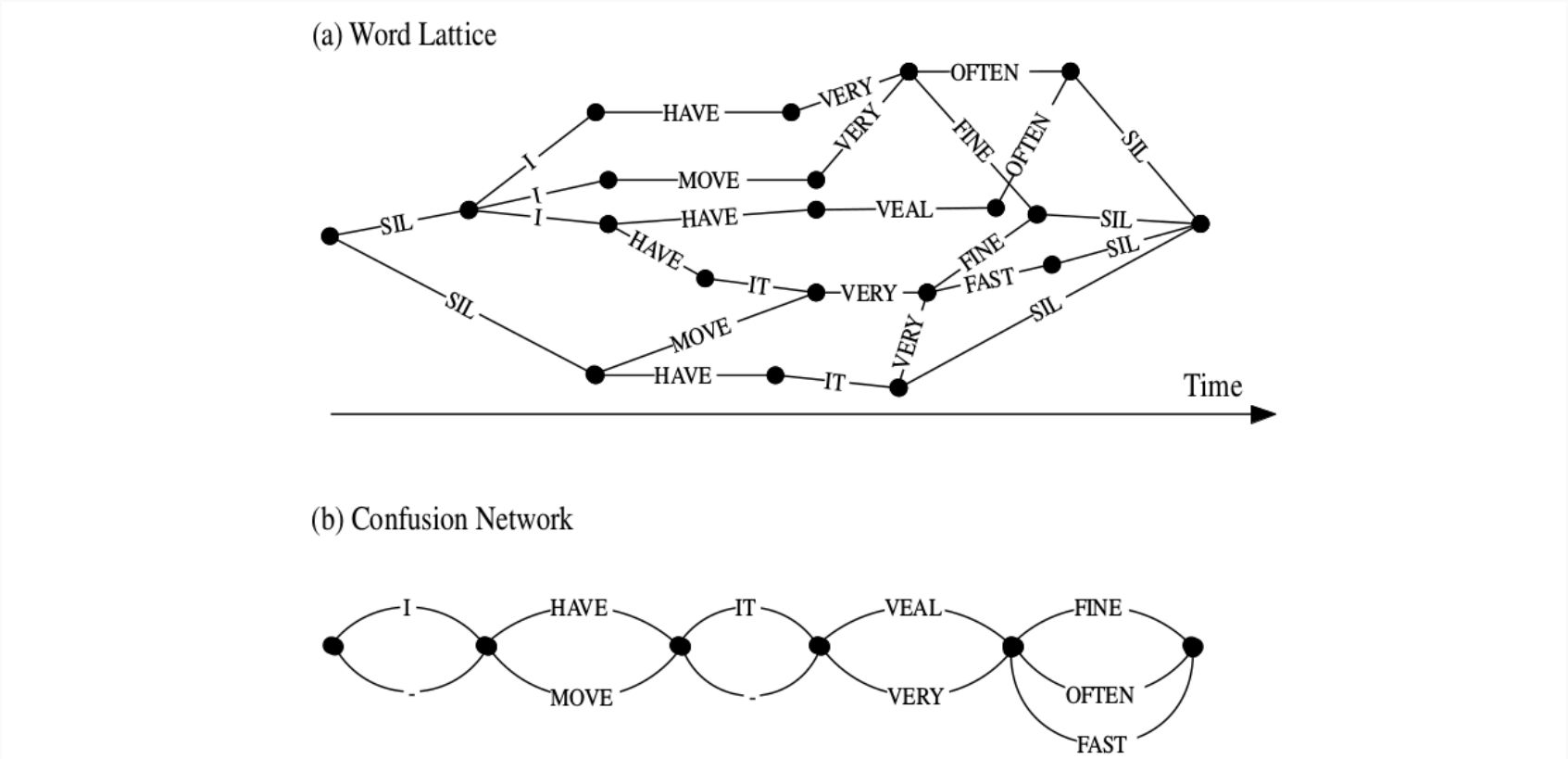
$$H = -\lim_{L\to\infty} \frac{1}{K} \log p(w_1, \dots, w_L).$$

Результаты

- О современных языковых моделях мы обязательно поговорим потом...
- А пока декодер идёт и алгоритмом Витерби всё решает.
- Но что именно решает?

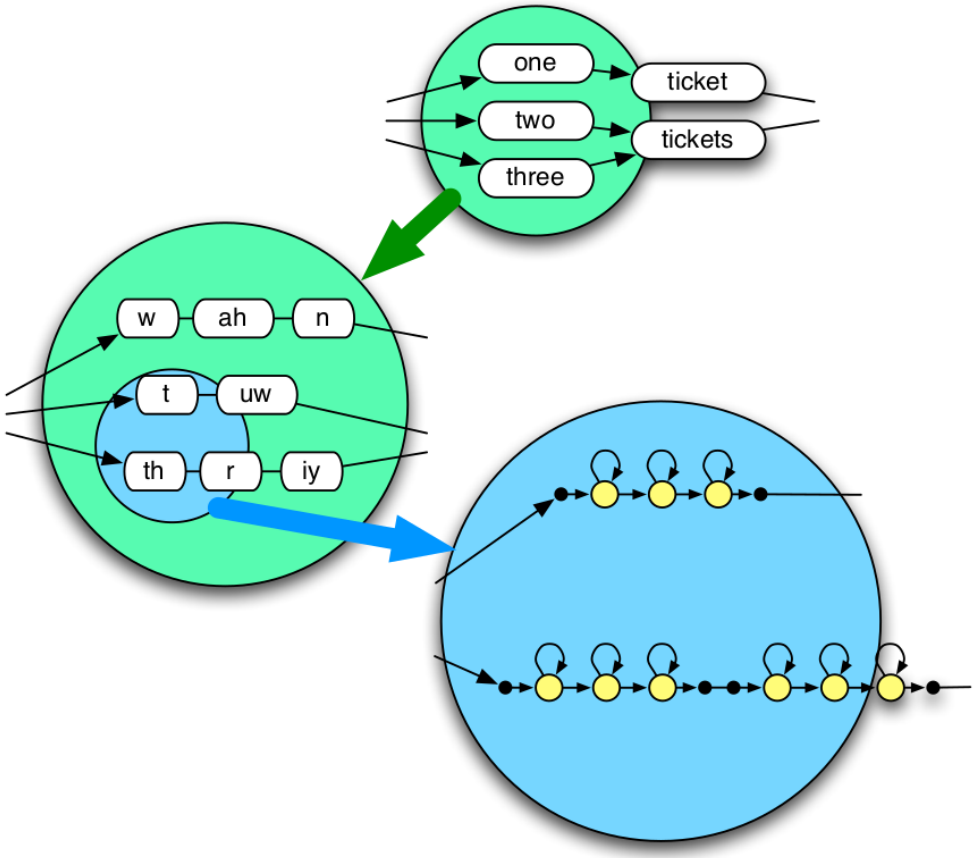
Результаты

- Возможности удобно представлять как *решётку слов* (word lattice) или *confusion network*.



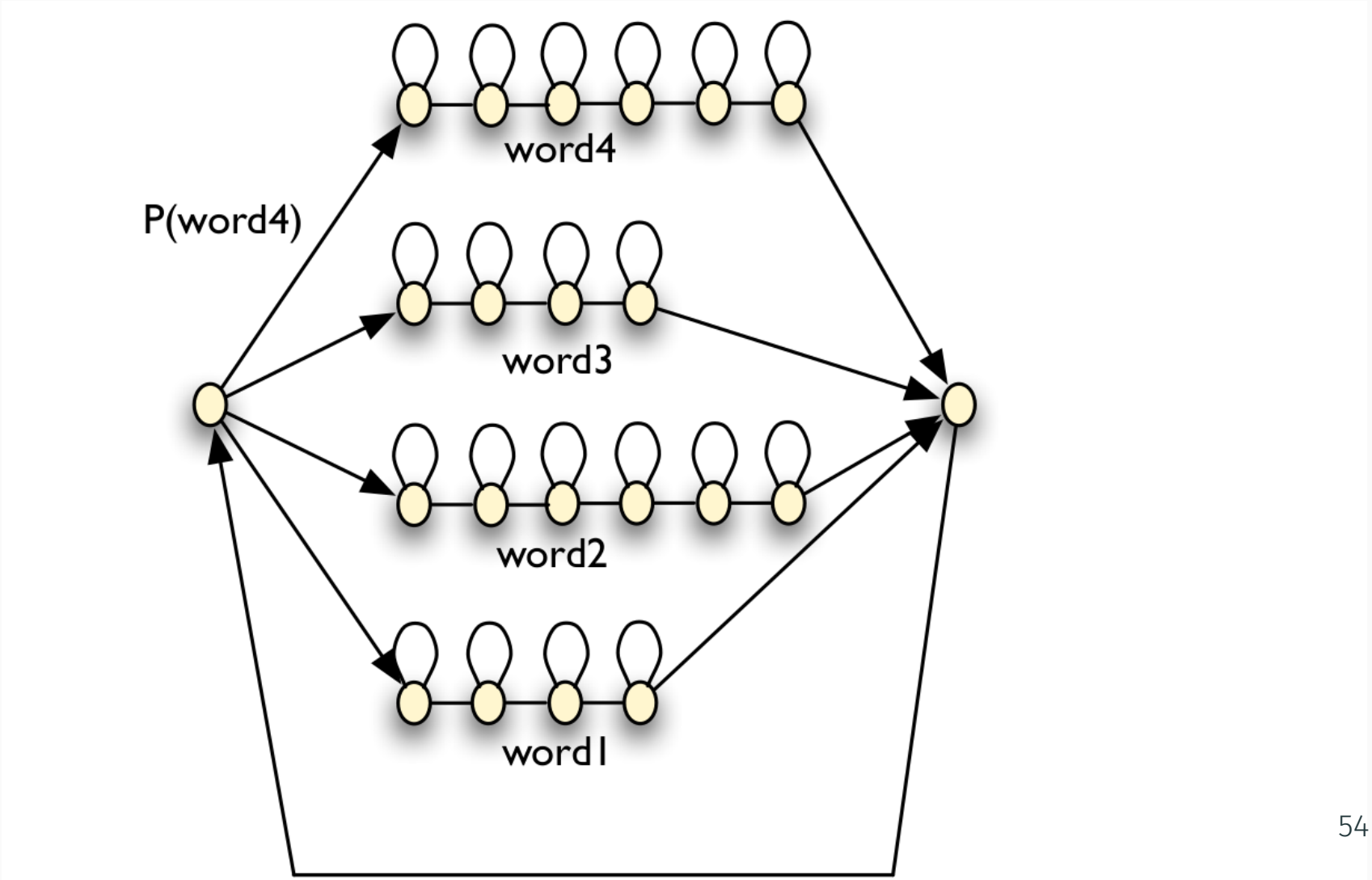
Результаты

- Сеть слов превращается в сеть фонем, потом в сеть состояний HMM.



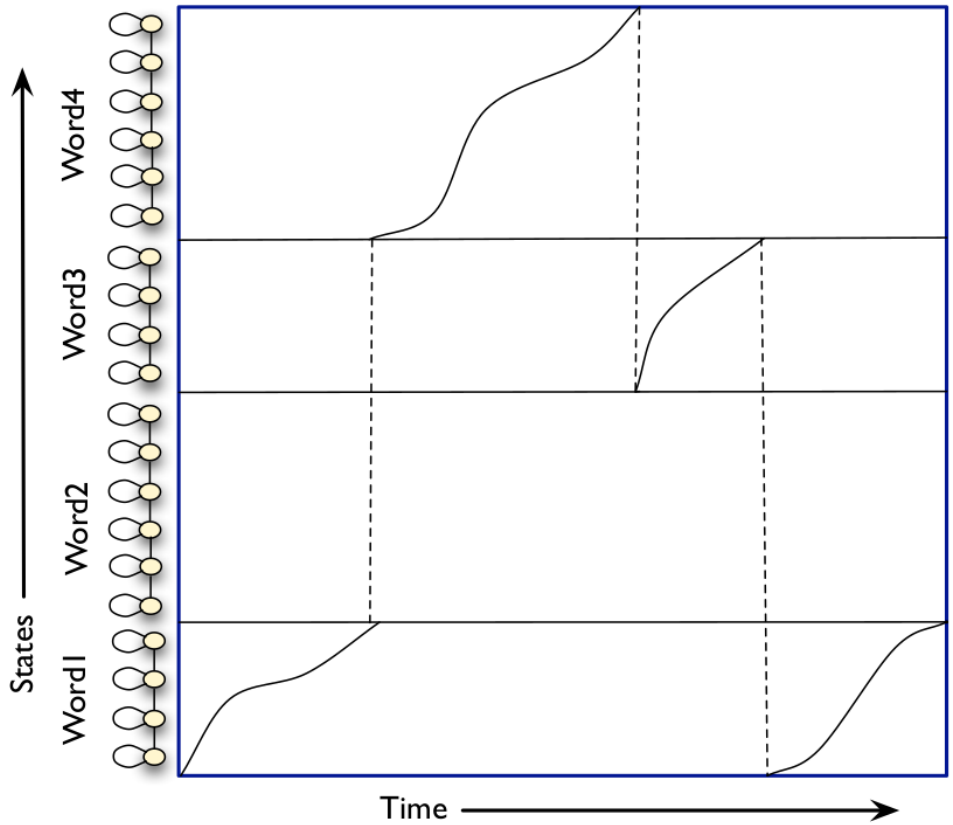
Результаты

- И можно распознавать связанные друг с другом слова тоже алгоритмом Витерби.



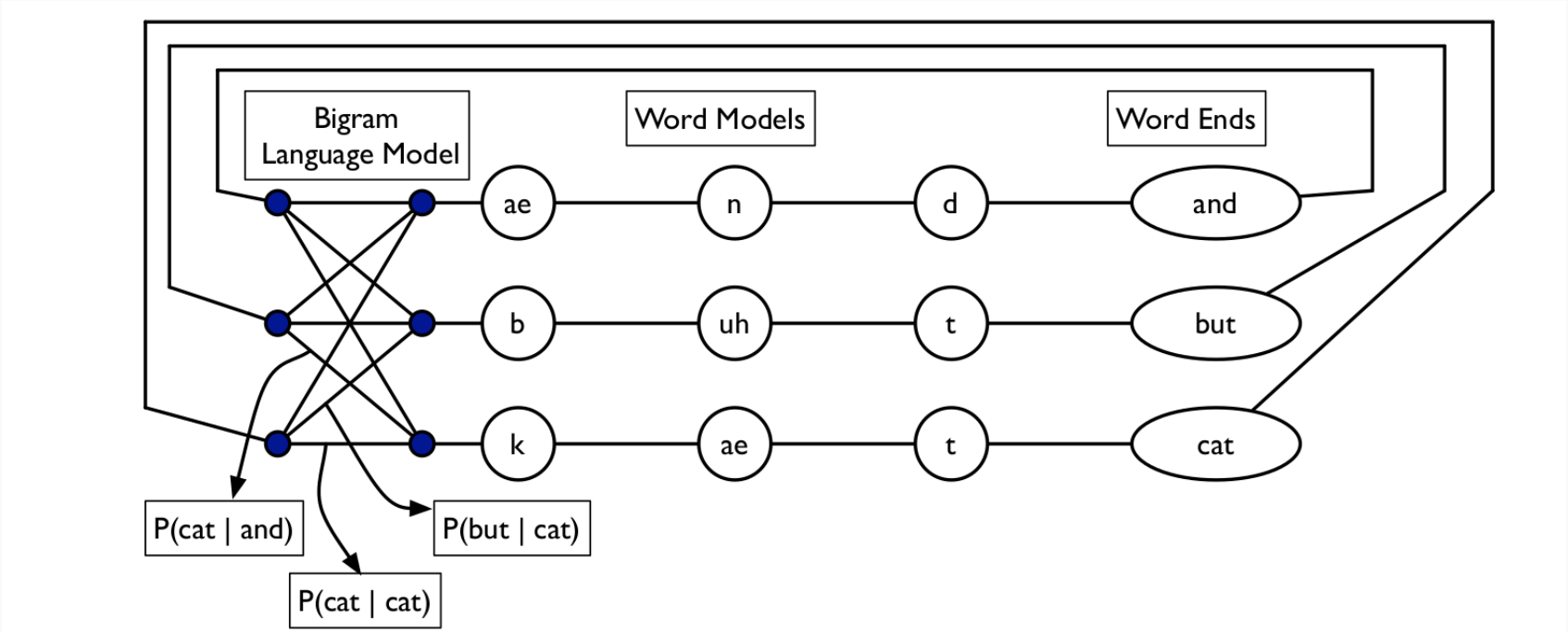
Результаты

- Всё это накладывается на собственно аудиозапись, получается последовательность во времени.



Результаты

- А языковая модель – это просто дополнительные множители (слагаемые в log), априорные вероятности.

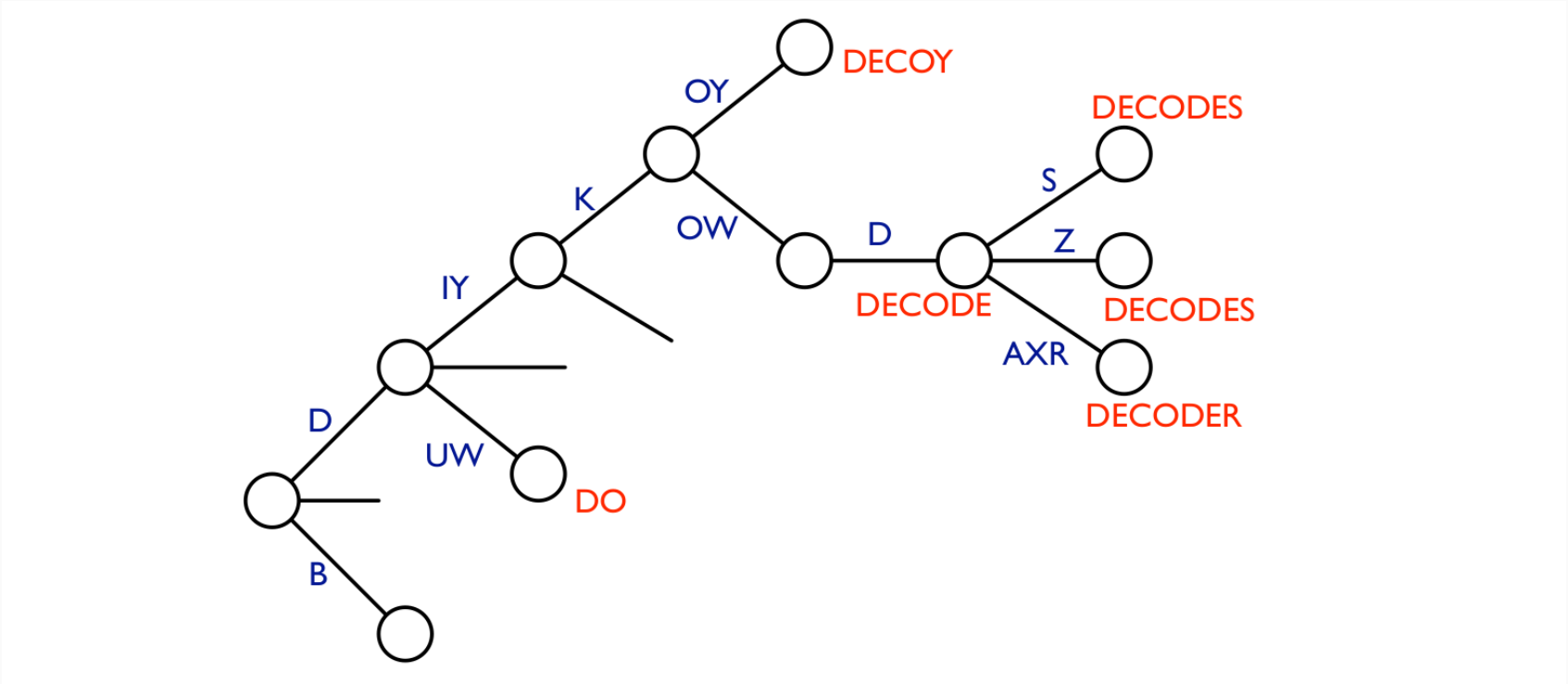


Результаты

- Декодирование по всей сети всего языка нереально.
- Обычно декодер генерирует и поддерживает только лучшие гипотезы; это называется *beam search*.
- Т.е. мы после каждого шага (обычно на границах слов) делаем pruning и либо выкидываем гипотезы, которые сильно хуже текущего лучшего варианта, либо просто поддерживаем N лучших (типа 1000).

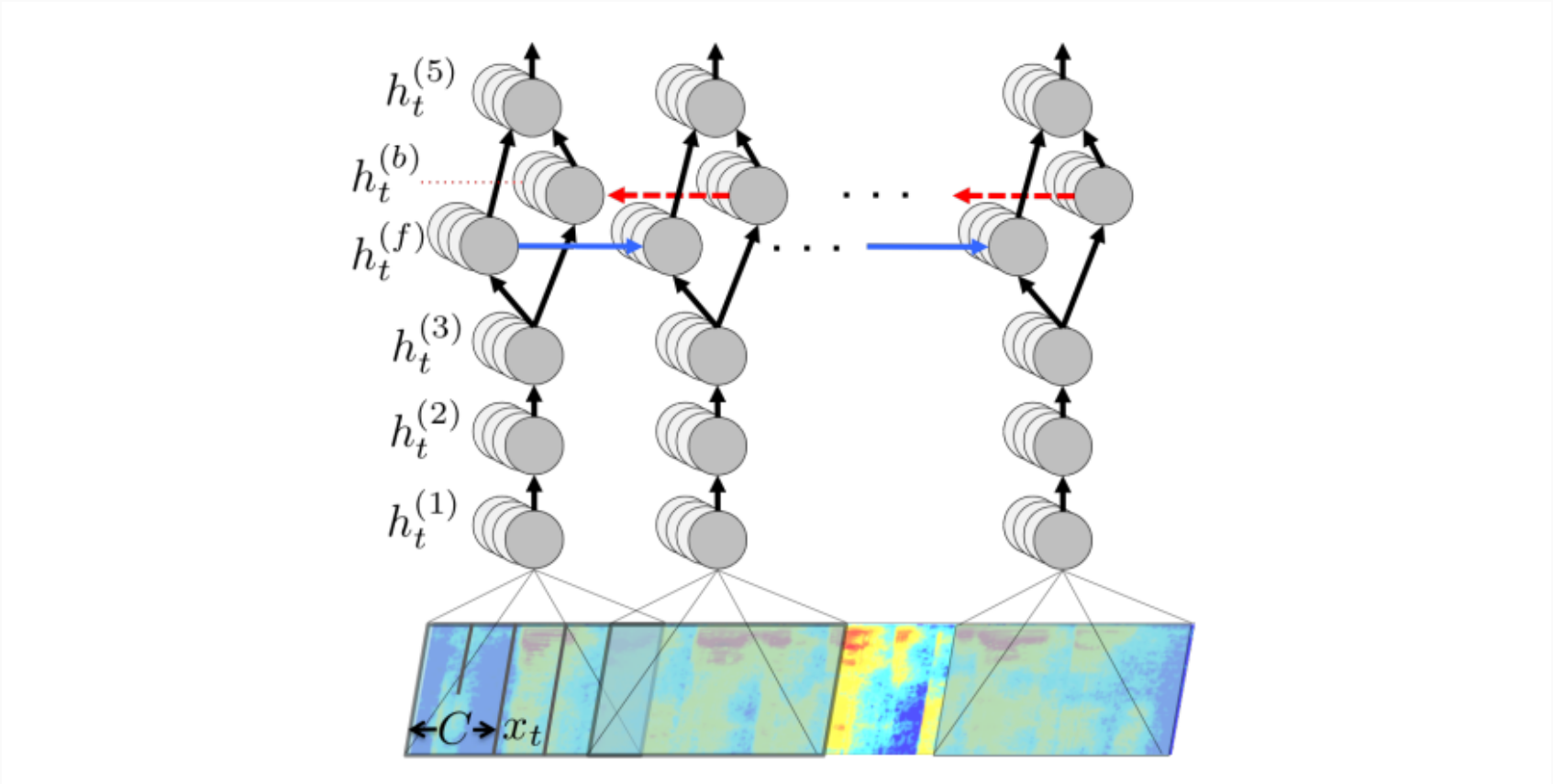
Результаты

- А HMM для отдельных слов (нам же нужно по HMM на каждое слово) тоже можно организовать в дерево (префиксное).



End-to-end

- Впрочем, сейчас уже многое делается по-другому.
- End-to-end speech recognition.



Спасибо!

Спасибо за внимание!