Федеральное государственное бюджетное образовательное учреждение
высшего образования «Санкт-Петербургский Государственный университет»

Кафедра «Статистического моделирования»

Отчет по курсовой работе на тему «Задачи оценивания значимости выравнивания при помощи скрытых марковских моделей»

Выполнил:
ст. группы 19.Б04-мм Власенко Даниил Владимирович
Научный руководитель:
к. ф.-м. н. Коробейников Антон Иванович

Санкт-Петербург
2021

1 Введение

Последовательность длины L — строка D состоящая из L символов алфавита A. Выравнивание последовательностей — размещение двух или более последовательностей друг под другом таким образом, чтобы было легче увидеть их схожие участки. Например, даны последовательности ACEAAFAE и CEAFDCE, если расположить их друг под другом, то не будет ни одного совпадения соответствующих символов, но если вставить пропуск восьмого символа в первой последовательности и пропуски первого и пятого символов во второй последовательности, то мы получим 5 совпадений.

```
A C E A A F A E
C E A F D C E

A C E A A F A — E
— C E A — F D C E
```

Оценка выравнивания — действительное число s, отражающее сходство последовательностей. Способом построения оценки выравнивания s может быть, например, увеличение оценки на 1 при совпадении символов, стоящих друг под другом, и уменьшение на $\frac{1}{2}$ при несовпадении. Тогда оценка s приведенного выше выравнивания будет равна 3. Способ оценки выравнивания выбирается исходя из целей и вида выравнивания.

Сходство последовательностей может отражать функциональные, структурные или эволюционные взаимосвязи объектов, которые описывают эти последовательности. Таким образом оценка выравнивание последовательностей может быть полезно в задаче определения степени родства биологических организмов путем сравнения их ДНК или РНК, нуклеотидных последовательностей, задаче анализа свойств белков, аминокислотных последовательностей, задаче распознавания речи человека или письменного языка и многих других приложениях.

Выше был приведен пример попарного выравнивания двух строк, но если сходство последовательностей слабое, то через такое выравнивание может не получится идентифицировать взаимосвязь описываемых последовательностями объектов. Однако сравнение сразу трех и более последовательностей может позволить выявить эту взаимосвязь, такое выравнивание называется множественным. Проводить множественное выравнивание стандартными методами динамического программирования для попарного выравнивания [Ссылка на первую книгу Алгоритмов в биоинформатики] вычислительно неэффективно, но оказывается, что аппарат скрытых марковских моделей (СММ) позволяет эффективно решать эту задачу [Ссылка на статью по HMM][Ссылка на вторую книгу Алгоритмов в биоинформатики].

СММ будут описаны далее, пока что зададимся следующим вопросом. Если есть множество последовательностей, описывающих взаимосвязанные объекты, имеется еще одна последовательность и была посчитана оценка выравнивания этой последовательности ко всему множеству каким-либо способом, то

- достаточно ли высокая эта оценка, чтобы считать объект, описываемый последовательностью, родственным к объектам, описываемым множеством, или шум, т.е. случайная последовательность, мог добиться такой оценки.
- достаточно ли низкая эта оценка, чтобы считать объект описываемый последовательностью, не родственным к объектам, описываемым множеством, или сигнал, т.е. последовательность, описывающая взаимосвязанный с множеством объект, мог получить такую оценку.

Ложноположительная вероятность оценки s — это вероятность того, что шум получит оценку равную или выше s. Истинноположительная вероятность оценки s — это вероятность того, что сигнал получит оценку равную или выше s.

Далее будет описаны метод, который позволяет эффективно вычислять введенные выше два термина.

2 Метод

Сначала опишем модели, затем алгоритмы, которые используются для манипуляции ими.

2.1 Модели

Нам потребуется Профильная СММ [Ссылка на вторую книгу Алгоритмы в Биоинформатике], с помощью которой будут вычисляться оценки последовательностей, и фоновая модель, которая будет описывать шум.

2.1.1 Скрытые марковские модели

Определение 1. *Пусть X_n и Y_n дискретные стохастические процессы, $n \geq 1$. Пара (X_n, Y_n) называется скрытой марковской моделью, если*

- *X_n — марковский процесс, поведение которого напрямую не наблюдается ("скрытый");*
- *$P(Y_n = y_n | X_1 = x_1, \ldots, X_n = x_n) = P(Y_n | X_n = x_n)$ для любого $n \geq 1$, где $x_1, \ldots, x_n$ — значения, принимаемые процессом X_n (**состояния модели**), y_n — значение, принимаемое процессом Y_n (**наблюдаемый символ модели**).*

Если для удобства реализации алгоритмов добавить специальное *начальное* и специальное *конечное* состояния, в которых СММ начинает и заканчивает работу и не испускает наблюдаемых символов, тогда *путь* π в СММ начинается в начальном состоянии, заканчивается в конечном состоянии и проходит от состояния к состоянию, испуская в каждом состоянии наблюдаемый символ. *Последовательность* D связанная с путем π — последовательность наблюдаемых символов, которая была получена в результате прохода СММ по пути π.

Оценка пути $s(\pi)$ — вероятность пути π, то есть $s(\pi) = P(X_n, \ldots, X_1)$.

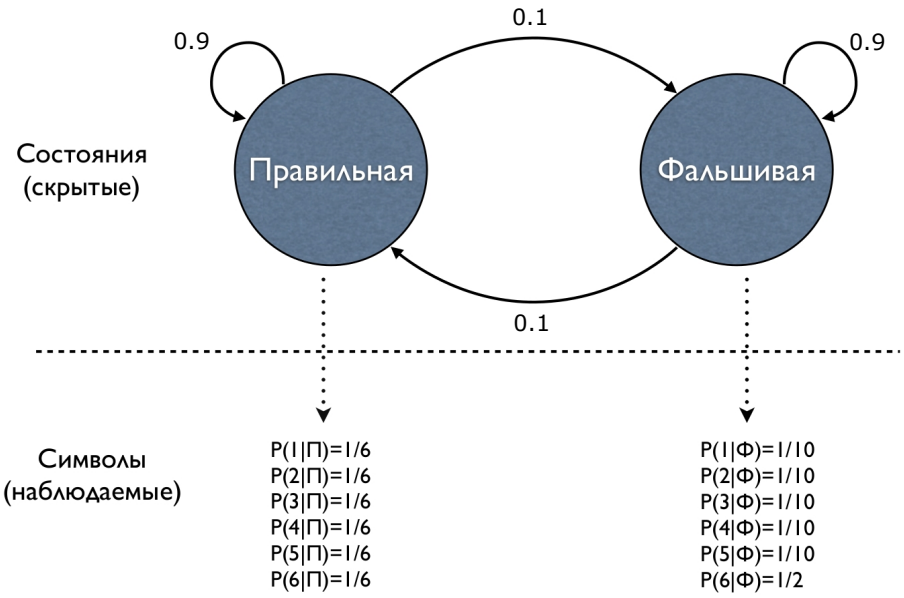

Рис. 1: Нечестное казино.

На протяжении семестра я работал с моделью «Нечестного казино» (рис. 1). Предположим, существует казино, которое вместе с обычной игральной костью использует нагруженную кость, которая с большей чем 1/6 вероятностью выдает определенное число. Кости могут меняться между собой с определенной вероятностью. Кости выглядят одинаково, поэтому мы не знаем, какая кость — честная или нагруженная — используется в определенный момент времени, но мы можем наблюдать цифры, которые выпадают раз за разом.

Целью работы на семестр было ознакомление с основами скрытых марковских моделей.

Все алгоритмы, используемые в курсовой работе, реализованы на языке C++ и доступны в публичном репозитории — https://github.com/Daniil-Vlasenko/Coursework5.git.

2.2 Обозначения

В примере выше кости являются состояниями модели, выпадающие числа — наблюдаемыми символами. Введем для дальнейшей работы обозначения:

M — число состояний модели;
K — число различных наблюдаемых символов;
N — длина наблюдаемой последовательности;
x_n — состояние модели в момент времени n;
y_n — наблюдаемый символ в момент времени n;
X — конечная последовательность состояний модели;
Y — конечная последовательность наблюдаемых символов;
$1, \dots, M$ — состояния модели;
$v_1, \dots, v_K$ — наблюдаемые символы;
$\pi = \{\pi_i\}, \pi_i = P(x_1 = i)$ — вероятность нахождения модели в состоянии $i \in 1, \dots, M$ в начале эксперимента, т.е. в момент времени $n = 1$;
$A = \{a_{ij}\}, a_{ij} = P(x_{n+1} = j | x_n = i)$ — вероятность перехода модели из состояния $i \in 1, \dots, M$ в состояние $j \in 1, \dots, M$;
$B = \{b_j(v_k)\}, b_j(v_k) = P(v_k | x_n = j)$ — вероятность получения наблюдаемого символа $v_k, k \in 1, \dots, K$ в состоянии $j \in 1, \dots, M$;
$\lambda = (A, B, \pi)$ — скрытая марковская модель с параметрами A, B, π.

3 Проделанная работа

На основе модели «Нечестного казино» и нескольких статей [1, 2, 3, 4] были изучены и реализованы алгоритмы, решающие три основные задачи, к которым сводится приложение скрытых марковских моделей:

1. Дана модель $\lambda = (A, B, \pi)$, как вычислить $P(Y|\lambda)$ — Forward–Backward процедуры.
2. Дана модель $\lambda = (A, B, \pi)$, нужно найти последовательность состояний X, которая максимизирует вероятность $P(Y, X|\lambda)$ — алгоритм Витерби.
3. Как изменить параметры модели $\lambda = (A, B, \pi)$, чтобы максимизировать вероятность $P(Y|\lambda)$ — алгоритм Баума—Велша.

Алгоритмы подробно описаны в статье [1], здесь приведу краткое изложение шагов алгоритмов и результаты их работы.

3.1 Forward–Backward процедуры

Предположим, что нужно узнать вероятность наблюдаемой последовательности $P(Y|\lambda)$. Для этого нужно посчитать $P(Y|X, \lambda)P(X|\lambda)$ для всех возможных последовательностей состояний X, затем сложить результаты.

$$P(Y|\lambda) = \sum_X P(Y|X, \lambda)P(X|\lambda) = \sum_X \pi_{x_1} b_{x_1}(y_1) a_{x_1 x_2} b_{x_2}(y_2) \dots a_{x_{N-1} x_N} b_{x_N}(y_N)$$

В этом уравнении происходит $2NM^N$ умножений, что при $M = 5$ и $N = 100$ будет означать примерно 10^{72} умножений, что делает такое вычисление вероятности крайне неэффективным.

Forward–Backward алгоритмы полезны тем, что позволяют вычислять вероятность наблюдаемой последовательности Y за M^2N умножений вместо $2NM^N$, если считать вероятность обычным методом через сумму произведений.

3.1.1 Forward процедура

Определим forward значение $\alpha_n(i)$ следующим образом:

$$\alpha_n(i) = P(y_1, y_2, \dots, y_n, x_n = i|\lambda)$$

т.е. это вероятность последовательности наблюдаемых символов Y до момента времени n, при этом в момент времени n модель λ находится в состоянии i. $\alpha_n(i)$ может быть посчитана последовательно:

1. $$\alpha_1(i) = \pi_i b_i(y_1), 1 \leq i \leq M$$

2. для $n = 1, 2, \ldots, N-1, 1 \leq j \leq M$

$$\alpha_{n+1}(j) = \left(\sum_{i=1}^{M} \alpha_n(i) a_{ij} \right) b_j(y_{n+1})$$

3. тогда

$$P(Y|\lambda) = \sum_{i=1}^{M} \alpha_N(i)$$

3.1.2 Backward процедура

Определим backward значение $\beta_n(i)$ следующим образом:

$$\beta_n(i) = P(y_{n+1}, y_{n+2}, \ldots, y_N | x_n = i, \lambda)$$

т.е. это вероятность наблюдаемой последовательности Y с момента времени $n+1$ до N, при этом в момент времени n модель находится в состоянии i. $\beta_n(i)$ считается так:

1. $$\beta_N(i) = 1, 1 \leq i \leq M$$

2. для $n = N-1, N-2, \ldots, 1, 1 \leq i \leq M$

$$\beta_n(i) = \sum_{j=1}^{M} a_{ij} b_j(y_{n+1}) \beta_{n+1}(j)$$

3. тогда

$$P(Y|\lambda) = \sum_{i=1}^{M} \pi_i b_i(y_1) \beta_1(i)$$

Оба алгоритма были протестированы и использованы далее в курсовой работе.

3.2 Алгоритм Витерби

Алгоритм Витерби позволяет найти последовательность состояний X^*, которая максимизирует вероятность появления данной наблюдаемой последовательности $P(Y, X|\lambda)$, т.е $X^* = \arg\max_X P(Y, X|\lambda)$.

1. Инициализация. Для $1 \leq i \leq M$

$$\delta_1(i) = \pi_i b_i(y_1)$$

$$\psi_1(i) = 0$$

2. Рекурсивное вычисление. Для $2 \leq n \leq N$ для $1 \leq j \leq M$

$$\delta_n(j) = \max_{1 \leq i \leq M} (\delta_{n-1}(i) a_{ij}) b_j(y_n)$$

$$\psi_n(j) = \arg \max_{1 \leq i \leq M} (\delta_{n-1}(i) a_{ij})$$

3. Прерывание алгоритма

$$P^* = \max_{1 \le i \le M} (\delta_N(i))$$

$$q_N^* = \arg \max_{1 \le i \le M} (\delta_N(i))$$

4. Восстановление оптимальной последовательности состояний. Для $n = N-1, N-2, \ldots, 1$

$$q_n^* = \psi_{n+1}(q_{n+1}^*)$$

Продемонстрирую работу алгоритма. Предположим, дана наблюдаемая последовательность $Y =$ 131545531254541235531113431315455312543412315515
534331445131224541235555155343. Тогда последовательность состояний X^*, максимизирующая $P(Y, X|\lambda)$ для модели указанной на рисунке 1, будет 0001111111
11111111111111100001111111111110001111111111111111000011100111111111.

3.3 Алгоритм Баума—Велша

Данный алгоритм позволяет изменять параметры λ так, чтобы максимизировалась вероятность появления заданной наблюдаемой последовательности $P(Y|\lambda)$. Предполагается, что известны некоторые начальные параметры модели λ. Для его реализации потребуется объявить две вспомогательные функции. Определим $\gamma_n(i)$:

$$\gamma_n(i) = P(x_n = i, Y|\lambda) = \frac{P(x_n = i, Y|\lambda)}{P(Y|\lambda)} = \frac{\alpha_n(i)\beta_n(i)}{P(Y|\lambda)}$$

т.е. это вероятность того, что в момент времени n модель будет находиться в состоянии i. Следовательно, сумма этих значений для конкретного i от $n = 0$ до $n = N$ будет представлять собой ожидаемое число посещений моделью состояния i.

Определим $\xi_n(i, j)$:

$$\xi_n(i,j) = P(x_n = i, x_{n+1} = j|Y, \lambda) = \frac{P(x_n = i, x_{n+1} = j, Y|\lambda)}{P(Y|\lambda)} = \\ = \frac{\alpha_n(i)a_{ij}b_j(y_{n+1})\beta_{n+1}(j)}{P(Y|\lambda)}$$

т.е. это вероятность того, что модель в момент времени n будет находиться в состоянии i, а в момент времени $n+1$ в состоянии j. Тогда сумма от $n = 0$ до $n = N-1$ для конкретных i и j будет означать ожидаемое число переходов модели из состояния i в состояние j.

Теперь можно определить формулы переоценки параметров Баума—Велша, которые будут изменять параметры модели так, чтобы вероятность появления заданной наблюдаемой последовательности $P(Y|\lambda)$ увеличивалась:

$$\hat{\pi}_i = \gamma_n(i), 1 \le i \le M$$

$$\hat{a}_{ij} = \sum_{n=1}^{N-1} \xi_n(i,j) \Big/ \sum_{n=1}^{N-1} \gamma_n(i)$$

$$\hat{b}_j(v_k) = \sum_{n=1, y_n = v_k}^{N} \gamma_n(i) \Big/ \sum_{n=1}^{N} \gamma_n(i)$$

Если повторять описанные выше вычисления для моделей с новыми параметрами раз за разом, то функция вероятности $P(Y|\lambda)$ будет сходиться к своему экстремальному значению.

Оказывается, что начальные параметры модели очень важны, так как алгоритм может сходиться к локальным экстремумам, что приводит к неверным результатам.

3.3.1 Начальные параметры алгоритма Баума—Велша

Предположим, что дана модель «Нечестного казино» λ, которая приводилась на рисунке 1. Мы хотим выяснить вероятностные параметры π, A, B модели λ. Для этого понадобится задать априорные параметры для модели λ, которые очень важны для результата.

Пусть честная кость — это состояние 1, нагруженная кость — состояние 2. Ясно, что честная кость будет выдавать результаты от 1 до 6 с вероятностью 1/6, т.е. $b_1(v_k) = 1/6, v_k = 1, \ldots, 6$. Вероятность 6 для нагруженной кости $b_2(6)$ может быть посчитана как частота, с которой 6 встречается в наблюдаемой последовательности, умноженная на 2. Тогда $b_2(v_k) = 1 - b_2(6)/5, v_k = 1, \ldots, 5$. Начальное распределение состояний не сильно влияет на результаты, поэтому можно задать его так $\pi = (0.5, 0.5)$. Осталось разобраться с параметрами переходов.

Нарисуем график функции вероятности $P(Y|\lambda)$ от значения параметра перехода состояния в себя (пусть эти параметры будут одинаковые для обоих состояний) (рис. 2). Такая функция называется функцией правдоподобия. Априорное значение 0.928 для перехода в себя наиболее подходящее, вычисления демонстрирует график, значит начнем с него.

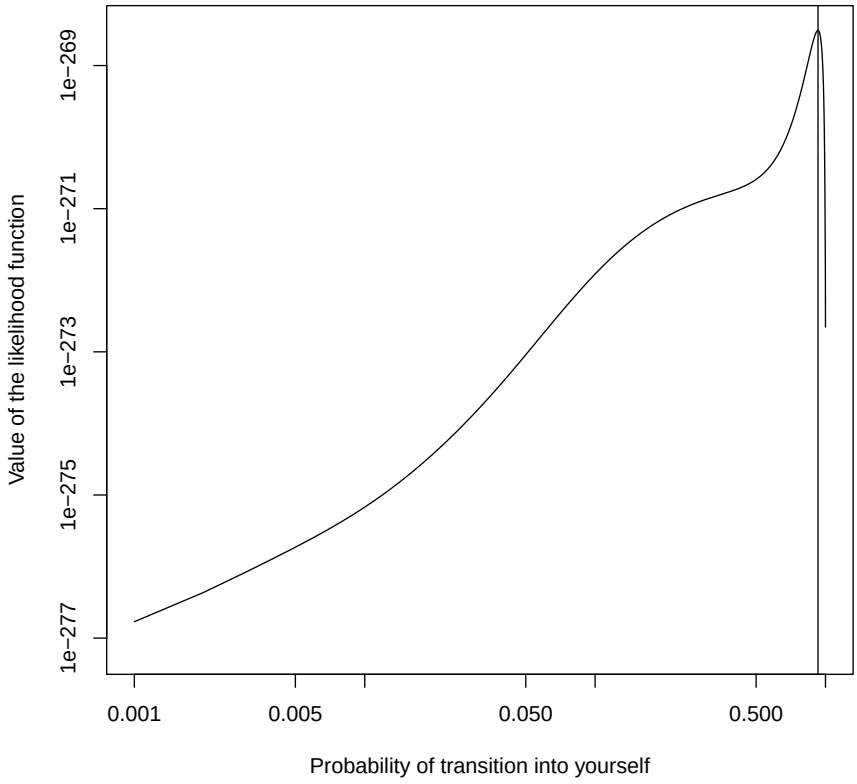

Рис. 2: Зависимость функции правдоподобия от вероятности перехода состояния в себя

Алгоритм сходится к определенным параметрам, повторим вычисления 100 раз и найдем среднее для параметров. Получаем, что

$$\bar{\pi} = (0.71041, 0.28959)$$

$$\bar{A} = \begin{pmatrix} 0.915482 & 0.084518 \\ 0.141648 & 0.858352 \end{pmatrix}$$

$$\bar{B} = \begin{pmatrix} 0.166206 & 0.166292 & 0.166868 & 0.162564 & 0.164274 & 0.173797 \\ 0.084606 & 0.084335 & 0.084714 & 0.082356 & 0.083339 & 0.580647 \end{pmatrix}$$

Дисперсия результатов алгоритма:

$$D(\pi) = (0.038035, 0.038035)$$

$$D(A) = \begin{pmatrix} 0.000111 & 0.000111 \\ 0.000058 & 0.000058 \end{pmatrix}$$

$$D(B) = \begin{pmatrix} 0.000506 & 0.000503 & 0.000538 & 0.000604 & 0.000568 & 0.000003 \\ 0.000387 & 0.00047 & 0.000418 & 0.000411 & 0.000374 & 0.007149 \end{pmatrix}$$

4 Дальнейшая работа

Дальше предстоит изучить методы выравнивания последовательностей, методы оценивания выравнивания последовательностей и обобщить имеющиеся знания об оценивании выравнивания для классических профильных скрытых марковских моделей на альтернативные классы скрытых марковских моделей, часто встречающихся в задачах биоинформатики.

5 Литература

[1] A tutorial on Hidden Markov Models : Rep. / Signal Processing and Artifcial Neural Networks Laboratory Department of Electrical Engineering Indian Institute of Technology ; Executor: Rakesh Dugad, U. B. Desai : 1996.

[2] Stamp Mark. Introduction to Machine Learning with Applications in Information Security // A Revealing Introduction to Hidden Markov Models. — Chapman and Hall, 2017.

[3] Jurafsky Daniel, Martin James H. Speech and Language Processing. // Hidden Markov Models. — Prentice Hall, 2021.

[4] Rabiner L.R. A tutorial on hidden Markov models and selected applications in speech recognition // Proceedings of the IEEE. — 1989. — P. 257 – 286.