Федеральное Государственное Автономное
Образовательное Учреждение Высшего Образования
Национальный Исследовательский Институт
«Высшая Школа Экономики»

Факультет Математики

Максим Раменский

О скорости расшифровки текста с помощью алгоритма Метрополиса-Гастингса

Курсовая работа за 2-ой год обучения

Образовательная программа: бакалавриат «Математика»

Научный руководитель:
Доцент факультета математики
Дымов Андрей Викторович

Москва 2024

1 Введение

Перед тем как приступить к последовательному изложению математики, стоит пояснить смысл происходящего: я попытаюсь рассказать о методе расшифровки текстов с использованием марковских цепей и связанной с ней математикой. Стоит заметить, что это далеко не самый эффективный алгоритм для расшифровки. Этот алгоритм основан на прямом продолжении идей А.А. Маркова – отца-основателя цепей.

Ко всему написанному *курсивом* стоит относиться, как к художественной литературе.

2 Алгоритм

Допустим, у вас есть строка S конечной длины, состоящая из символов из некоторого алфавита $\mathcal{A}$. Например, $S =$ `Alyona was walking in the park in the center`, а $\mathcal{A}$ - латинский алфавит с пробелом. Вы не знаете саму строку S, но у вас есть строка S_*, полученная применением к каждому символу S некоторой неизвестной вам перестановки $\sigma_* : \mathcal{A} \mapsto \mathcal{A}$:

$$S_* = \sigma_*(S) = \sigma_*(a_1) \dots \sigma_*(a_n).$$

Вам нужно найти исходную строку S или эквивалентно перестановку σ_*.

Один из способов решения этой задачи - использовать статистические вероятности переходов между символами в тексте на языке $\mathcal{A}$. Затем можно рассматривать любую строку как траекторию марковской цепи с этими вероятностями. Таким образом, вероятность строки может быть вычислена как произведение вероятностей переходов между символами. А вероятности переходов между символами мы возьмем в большом куске хорошего английского текста, например «Война и мир», и вычислим вероятность перехода между элементами алфавита + пробел.

Для поиска исходной строки S или перестановки σ_* можно использовать "квазивероятность определенную как произведение вероятностей переходов между символами.

$$\widetilde{\mathbb{P}}(S) = p_{a_1 a_2} \cdot p_{a_2 a_3} \dots p_{a_{n-1} a_n}. \tag{1}$$

Задача сводится к поиску аргумента с максимальной "квазивероятностью".

Подробнее можете прочитать в записке лекций [4].

После введения основных определений расскажу как устроен сам алгоритм:

1. Стартуем с произвольной перестановки σ_0. То есть у нас закодировано сообщение посредством $\sigma_0 : \mathcal{A} \mapsto \mathcal{A}$. Смотрим на ее «квазивероятность»

$$\pi_{\sigma_0} = \widetilde{\mathbb{P}}(\sigma_0^{-1}(S_*))$$

2. Делаем случайную транспозицию τ и обозначим $\widetilde{\sigma_1} = \tau \circ \sigma_0$. Найдем ее «квазивероятность» $\pi_{\widetilde{\sigma_1}}$.

3. Мы принимаем перестановку $\widetilde{\sigma_1}$, то есть $\sigma_1 = \widetilde{\sigma_1}$, с вероятностью

$$\min\left(1, \frac{\pi_{\widetilde{\sigma_1}}}{\pi_{\sigma_0}}\right)$$

   То есть, если транспозиция, увеличила квазивероятность, то мы ее принимаем, а если уменьшила, то берем с некой вероятностью. Это сделано для того, чтобы мы могли выходить из локальных максимумов в поиске глобального.

4. Продолжаем итерировать действия выше.

Для обоснования работоспособности алгоритма нужно воспользоваться алгоритмом Метрополиса – Гастнинга. Смотрите в [4].

3 Результаты

Весь код, который я написал в процессе курсовой работы содержится здесь [5]

3.1 Матрица переходных вероятностей

Первоначальное применение цепей Маркова состояло в анализе русской поэмы Евгений Онегин. Марков проанализировал чередование согласных и гласных, которую он смоделировал как простую цепь Маркова с двумя состояниями. Нам же в эпоху компьютеров можно не ограничиваться на 2 состояниях, и взять 27 состояний. Его гипотеза состояла в том, что по матрице переходных вероятностей можно отличать авторов друг от друга: у каждого автора свой словарный запас, и следовательно примерно постоянная матрица переходных вероятностей.

Сегодня существует много различных способов формального определения авторства, один из которых основан на Марковских цепях.(см. подробнее [2] стр. 3) Процедура Маркова очень похожа на сравнение двух аналитических функций f и g в точке x_0: насколько они похожи в ее окрестности. Взять два текста и провести их частотный анализ: посмотреть с какой частой выпадает та или иная буква – это будет эквивалентно $f(x_0) = g(x_0)$. Далее смотрим на матрицы переходных вероятностей – это эквивалентно совпадению производных функций в точке. Далее смотрим на N-граммы (смотри в пункте 1.2 [1]) – эквивалент совпадению значения n-той производной в точке.

Давайте же взглянем на пример матрицы, для рассказов Говарда Лавкрафта.

Из рисунка видны некоторые естественные характеристики английского языка: практически всегда после буквы 'Q' будет идти 'U', буква 'Y' обычно идет в конце слова, как и буква 'D'. Вообще можно сделать много лингвистических наблюдений.

В процессе курсовой работы я построил матрицы для текстов различной направленности эпохи и языка, а так же провел их частотный анализ. Как и в статье [1] пункта 4, частотный анализ (сравнение значений функций в точке) смог различить лишь языки написания текста. Но уже матрицы переходных вероятностей смогли различать художественную литературу от технической.

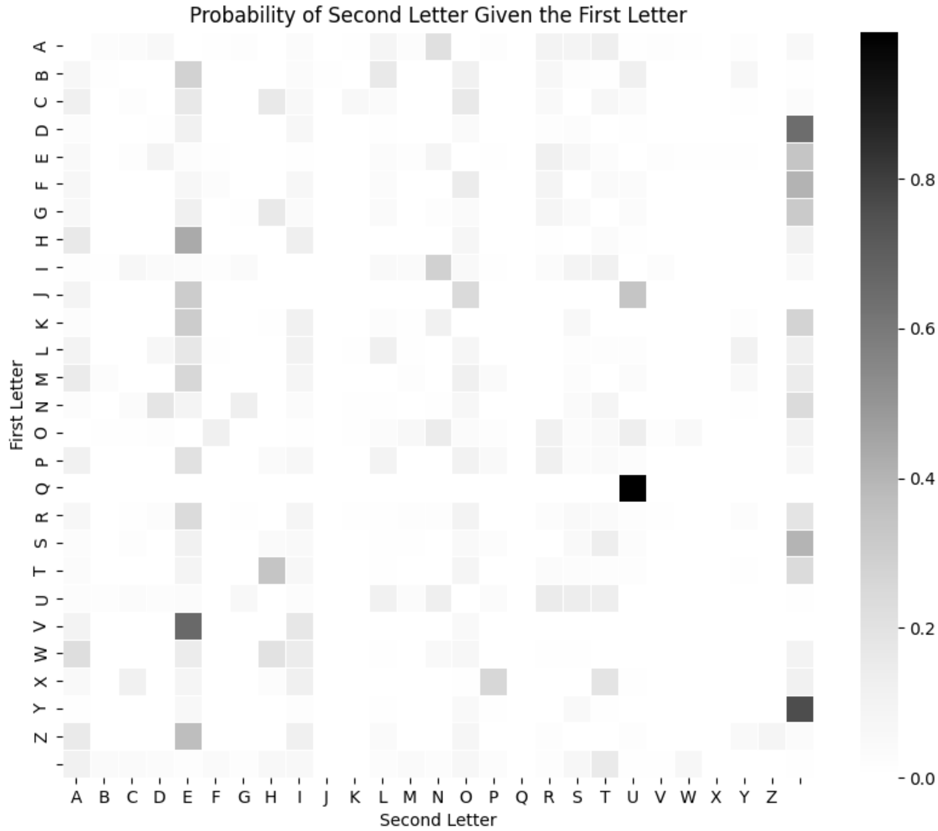

Рис. 1: Матрица по рассказам Говарда Лавкрафта

3.2 Декодирование

Алгоритм работает: он действительно максимизирует функцию вероятности в смысле 1. Понятное дело, что текст для расшифровки нужно брать достаточно большим (в моем случае алгоритм начинал «хорошо» работать начиная от ~ 30 различных слов. Подробнее об этой проблеме см. пункт 5.1 [1].

Рассмотрим пример(первым элементом идет номер итерации, вторым текущий «наивероятнейший» текст, третьим натуральный логарифм вероятности):

0 XACIMJIUSMZILZYIMKUJMGUJMCURMX ARNMKLCURMYATIMF AYGMKLSYGMDF LLYMDXUXIMTUCAYJMDXQGIRXMNSLQZMFLQRXSJMZSLPYICM URGMZUSXMZ YUFIMFUDIMKIIVMFLCZURJ MDJDXICMZSLNSUCMEQIDXALRMKLSVMNLHISRCIRX -1003.9

1000 TIDE YEOL REARCE VOY HOY DON TPINK VADON CIZE MPICH VALCH SMPAAC STOTE ZODICY STWHENT KLAWR MAWNTLY RLAUCED PONH ROLT RCOME MOSE VEEF MADRONY SYSTED RLAKLOD GWESTIAN VALF KABELNDENT -523.6

10000 TIME YEAR PEOPLE BAY DAY MAN THINF BOMAN LIVE CHILD BORLD

SCHOOL STATE VAMILY STUDENT FROUP COUNTRY PROKLEM HAND PART PLACE CASE BEEG COMPANY SYSTEM PROFRAM JUESTION BORG FOWERNMENT -470.7

Как мы видим, что осталось немного «доделать» руками и мы расшифровали.

Но есть нюанс. *И очень странно, почему об этой проблеме не пишут в статьях, где рассматривается имплантация алгоритма расшифровки с помощью марковских цепей, я попробую это исправить.*

Вера в нашу эвристику, что самая большая вероятность вымысле 1 будет у σ_*^{-1} оказывается не верна. И доказательством этого является наш пример выше: в результате работы кода максимальная вероятность получилась -470.7, тогда как натуральный логарифм вероятности для исходного текста составляет -482.

4 Попытка оценить скорость сходимости

Определение 1: Говорят, что однородная марковская цепь *перемешивает* (или обладает свойством перемешивания), если она имеет единственную стационарную меру π и для любого начального распределения $p^{(0)}$ соответствующее распределение $p^{(n)}$ в момент времени n удовлетворяет

$$p^{(n)} \to \pi \quad, n \to \infty.$$

Под $p^{(n)}$ я понимаю вероятностное распределение на n итерации алгоритма, а сходимость понимается покомпонентно.
Определение 2: Говорят, что однородная марковская цепь экспоненциально перемешивает, если она перемешивает и существуют постоянные $C > 0$ и $0 < \lambda < 1$, такие что для любого начального распределения $p^{(0)}$ имеем

$$|p^{(n)} - \pi|_{\ell_1} \leq C\lambda^n \quad \text{где } \pi \text{ - стационарное состояние.}$$

Наш код для расшифровки текстов, ходит по марковской цепи состоящей из всевозможных 27! перестановок, где из одного состояние в другое можно попасть посредством транспозиции. Будем понимать под скоростью сходимости количество итераций алгоритма для для декодирования сообщения. Наша цепь – однородная, с s-положительной матрицей переходов M. Где s-положительное число, такое что $(M^s)_{ij} > 0$, $\forall i, j$. Стационарное состояние нашей цепи – когда наш текст расшифрован. Поэтому применима теорема:

Теорема: Однородная марковская цепь с конечным числом состояний и s-положительной матрицей переходных вероятностей экспоненциально перемешивает.

При этом известны оценки на константы C и λ:

$$C \leq 2(1-\alpha)^{-1},\ \lambda \leq (1-\alpha)^{1/s},\ \text{где } \alpha = \min_{i,j} p_{ij}^{(s)} > 0.$$

Для оценки сходимости мне также понадобится следствие из закона больших чисел для марковских цепей, формулировку которого можно прочесть в [3]:

$$\mathbb{P}(|\nu - \pi_{\sigma_*}| > \varepsilon) \leq (1 + \frac{2C}{1-\lambda})\frac{1}{\varepsilon^2 n},$$

где ν – частота попадания на нужную расшифровывающую перестановку, π_{σ_*} – стационарный вес расшифровывающей перестановки.

Для нашей марковской цепи наилучшей оценкой для $\frac{1}{1-\lambda}$ является 10^{58}, что совершенно не имеет отношения к сути происходящего, ведь на практике алгоритм расшифровывает за ~ 25000 итераций, подробнее об этих результатах см. в заключительной главе [1].

Список литературы

[1] Stephen Connor. *Simulation and Solving Substitution Codes*. URL: https://www-users.york.ac.uk/~sbc502/decode.pdf.

[2] Т. В. Батура. *Формальные методы определения авторства текстов*. URL: https://cyberleninka.ru/article/n/formalnye-metody-opredeleniya-avtorstva-tekstov/viewer.

[3] Андрей Дымов. *Лекции по марковским цепям: закон больших чисел*. URL: https://math.hse.ru/data/2023/11/27/2107838147/%D0%9B%D0%B5%D0%BA%D1%86%D0%B8%D1%8F%208%20(%D0%97%D0%B0%D0%BA%D0%BE%D0%BD%20%D0%B1%D0%BE%D0%BB%D1%8C%D1%88%D0%B8%D1%85%20%D1%87%D0%B8%D1%81%D0%B5%D0%BB).pdf.

[4] Андрей Дымов. *Лекции по марковским цепям: расшифрофка текстов*. URL: https://math.hse.ru/data/2023/12/14/2112246358/%D0%9B%D0%B5%D0%BA%D1%86%D0%B8%D1%8F%2010%20(%D0%B0%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC%20%D0%9C%D0%B5%D1%82%D1%80%D0%BE%D0%BF%D0%BE%D0%BB%D0%B8%D1%81%D0%B0-%D0%93%D0%B0%D1%81%D1%82%D0%B8%D0%BD%D0%B3%D1%81%D0%B0).pdf.

[5] Максим Раменский. *Код для расшифрофки текста на Python*. URL: https://colab.research.google.com/drive/1B5LnmaZZlAgPRuaW8yv3gLIZ-6k2zzQe?usp=sharing.