Распознавание букв на основе прокрустова преобразования

Муха В.С.
Кафедра ИТАС, факультет информационных технологий и управления
Белорусский государственный университет информатики и радиоэлектроники
Минск, Беларусь
e-mail: mukha@bsuir.by

Сформулирована и решена задача линейного прокрустова преобразования обычных (двухмерных) матриц. Предложен подход и разработаны алгоритмы использования линейного прокрустова преобразования для распознавания букв. Приведены результаты распознавания печатных латинских букв, подверженных вращениям и отражениям.

I. Введение

Прокрустово преобразование (Procrustes transformation) находит широкое применение для сравнения различных родственных множеств данных во многих предметных областях: химии, биологии, психологии, географии, антропологии, геодезии и др. [1–3].

Термин «прокрустово преобразование» непосредственно связан с термином «уложить в прокрустово ложе». Прокрустово преобразование относится к математической проблеме сравнения матриц [1]. Идея прокрустова преобразования состоит в том, чтобы преобразованием P сдвига, вращения, масштабирования подогнать $(k \times n)$ -матрицу данных X к другой $(k \times n)$ -матрице Y таким образом, чтобы результат преобразования $D = PX$ как можно меньше отличался от Y. Здесь X – гость-неудачник, Y – прокрустово ложе, P – «преобразователь»-Прокруст. Поскольку столбцы матриц X и Y представляют собой упорядоченные точки в многомерном пространстве, т.е. некоторые фигуры, то прокрустово преобразование состоит в том, чтобы подогнать одну фигуру как можно ближе к другой и по остающемуся различию ответить на вопрос, представляют ли эти наборы точек одну и ту же фигуру, но в различных формах, или же они представляют две различные фигуры.

Классической считается ортогональная прокрустова задача

$$tr((Y - GJ' - cBX)'(Y - GJ' - cBX)) \to \min_{B,G,c}, \quad (1)$$

где B – ортогональная $(k \times k)$-матрица вращения-отражения, $G' = (g_1, g_2, ..., g_k)$, $J' = (1_1, 1_2, ..., 1_n)$ – векторы, c – скаляр. Вектор G называется вектором смещения или сдвига, а скаляр c – коэффициентом растяжения-сжатия или масштабирования. Преобразование (1) осуществляет смещение, поворот-отражение и масштабирование с сохранением пропорций пространственной фигуры, определяемой матрицей X. Результат такого преобразования хорошо узнается человеком как исходная фигура.

Ортогональное прокрустово преобразование связано с необходимостью выполнения сингулярного разложения (singular value decomposition, SVD) матрицы YX'. Алгоритм сингулярного разложения матрицы нельзя считать тривиальным. Достаточно сказать, что разработчики открытой системы программирования Matlab не предоставили доступа к коду программы svd.m, выполняющей сингулярное разложение.

Другим возможным типом преобразования является линейное преобразование

$$D = A + BX, \quad (2)$$

где A – $(k \times n)$ -матрица, B – $(k \times k)$ -матрица. Преобразование (2) осуществляет более широкий спектр преобразований, а именно, смещение, поворот-отражение и масштабирование как с сохранением, так и без сохранения пропорций пространственной фигуры, определяемой матрицей X. Тем не менее результат такого преобразования в большинстве случаев будет узнаваем человеком как исходная фигура.

С учетом трудностей, связанных с программной реализацией ортогонального прокрустова преобразования (1), и возможностей линейного прокрустова преобразования (2) в ряде практических задач можно отдать предпочтение линейному прокрустову преобразованию (2) без предположения ортогональности.

Задача линейного прокрустова преобразования впервые была сформулирована и решена в [4, 5]. В данной статье приводится теорема о линейном прокрустовом преобразовании и рассматриваются вопросы применения прокрустова преобразования в задаче распознавания букв.

II. Линейное прокрустово преобразование

Теорема 1. Если $X = (x_{i,\mu})$ *и* $Y = (y_{i,\mu})$, $i = \overline{1,k}$, $\mu = \overline{1,n}$, *– две* $(k \times n)$*-матрицы данных,* $n \ge k + 1$, *то линейное преобразование* $D = A + BX$ *с матрицей* $A = (A^*, A^*, ..., A^*)$, *состоящей из* n *одинаковых столбцов* $A^* = (a_i)$, $i = \overline{1,k}$, *и произвольной* $(k \times k)$*-матрицей* B, *обеспечивающее минимум критерия*

$$r = \|Z\|_E^2 = \|Y - D\|_E^2 = \sum_{i=1}^{k} \sum_{\mu=1}^{n} z_{i,\mu}^2,$$

определяется матричным выражением

$$D = \overline{\overline{Y}} + \mathring{Y}(\mathring{X})'(\mathring{X}(\mathring{X})')^{-1}\mathring{X},$$

в котором $\overset{\circ}{X} = X - \overline{\overline{X}}$, $\overset{\circ}{Y} = Y - \overline{\overline{Y}}$, $\overline{\overline{X}} = (\overline{X}, \overline{X}, ..., \overline{X})$, $\overline{\overline{Y}} = (\overline{Y}, \overline{Y}, ..., \overline{Y})$,

$$\overline{X} = \frac{1}{n}\sum_{\mu=1}^{n} X_\mu, \ \overline{Y} = \frac{1}{n}\sum_{\mu=1}^{n} Y_\mu,$$

X_μ , Y_μ , $i = \overline{1,n}$, – столбцы матриц X , Y , а нормированное прокрустово расстояние – выражением

$$\rho_{\min} = \frac{r_{\min}}{\| \overset{\circ}{Y} \|_E^2} = 1 - \frac{tr\left(\overset{\circ}{Y}(\overset{\circ}{X})'(\overset{\circ}{X}(\overset{\circ}{X})')^{-1} \overset{\circ}{X}(\overset{\circ}{Y})' \right)}{tr\left(\overset{\circ}{Y}(\overset{\circ}{Y})' \right)}.$$

III. Распознавание букв на основе прокрустова преобразования

Линейное прокрустово преобразование может быть использовано для распознавания букв. С этой целью распознаваемые буквы и буквы-эталоны необходимо представить в виде упорядоченных наборов точек (матриц). Решение при распознавании выносится в пользу той буквы-эталона, прокрустово расстояние от которой до распознаваемой буквы наименьшее по сравнению с другими буквами-эталонами. Алгоритм представления буквы в виде упорядоченного набора точек будем называть алгоритмом векторизации буквы, так как он определяет векторное описание растрового изображения буквы. Наиболее естественным представляется выбор точек на внешнем контуре черно-белого изображения буквы. Алгоритм векторизации буквы заключается в выборе последовательных черных точек контура буквы и последующем равномерном их прореживании для оставления определенного и одинакового для каждой буквы числа точек. Алгоритм должен работать автоматически и обеспечивать единообразный выбор начальной точки для всех букв. Обоснование способа выбора первой точки буквы лежит в предположении, что буква представляет собой некоторую плоскую кривую. Это обоснование содержится в следующей теореме.

Теорема 2. Пусть $\Gamma = \{(x(t), y(t)); a \le t \le b\}$ – непрерывная кривая в плоскости xOy , $t \in [a,b]$ – параметр, координатные функции $x(t), y(t)$ кривой ограничены снизу и сверху, т.е.

$$m_x \le x(t) \le M_x, \ m_y \le y(t) \le M_y,$$

$m_x \le M_x$, $m_y \le M_y$ – некоторые числа, и x_0, y_0 – координаты центра массы этой кривой,

$$x_0 = \frac{1}{b-a}\int_a^b x(t)dt, \ y_0 = \frac{1}{b-a}\int_a^b y(t)dt.$$

Тогда существует точка (x, y_0), лежащая на кривой.

На основании этой теоремы мы можем рассчитывать, что начальную точку для векторизации буквы мы обязательно обнаружим справа или слева от центра массы буквы. В укрупненной форме алгоритм векторизации растрового черно-белого изображения буквы состоит в следующем.

1. Находим центр массы буквы (x_0, y_0) по формулам

$$x_0 = \frac{1}{n}\sum_{i=1}^{n} x_i, \ y_0 = \frac{1}{n}\sum_{i=1}^{n} y_i,$$

где x_i , y_i , $i = \overline{1,n}$, – координаты черных пикселов в пределах растрового пространства, содержащего букву, n – количество черных пикселов.

2. Организуем движение вправо от центра массы до достижения черного пикселя на внешнем контуре буквы. Если такой пиксель не обнаружен, то организуем движение влево от центра массы. Полученная в результате выполнения этого пункта алгоритма точка является первой точкой векторного представления буквы.

3. Организуем движение от первой точки против часовой стрелки, фиксируя координаты каждого последующего черного пикселя внешнего контура буквы до достижения первой точки.

4. Выполняем равномерное прореживание полученных точек для оставления требуемого их количества.

Данный алгоритм был запрограммирован и использован для векторизации букв латинского алфавита. Растровые двоичные файлы букв-эталонов и распознаваемых букв формировались (нарезались) вручную и сохранялись как файлы *.bmp. Каждый файл подвергался векторизации и сохранялся в виде текстового файла из двух строк: в первой строке абсциссы точек буквы, во второй – ординаты. В каждой букве удерживалось 40 точек. Распознавание осуществлялось следующим образом: считывались текстовые файлы букв-эталонов и распознаваемых букв и вычислялись прокрустовы расстояния между распознаваемыми и эталонными буквами. Решение выносилось по минимальному прокрустову расстоянию между буквами. Определенные модификации алгоритма распознавания позволили уверенно распознавать печатные латинские буквы и цифры, подверженные вращениям и отражениям, что недоступно мировому лидеру в области распознавания текста программе FineReader [6].

[1] Schneider W. Jesper, Pia Borlund. Matrix comparison, part 2: Measuring the resemblance between proximity measures or ordination results by use of the mantel and procrustes statistics // Journal of the American Society for Information Science and Technology, Volume 58, Issue 11, September 2007, Pages: 1596–1609.

[2] Andrade J.M. et al. Procrustes rotation in analytical chemistry, a tutorial // Chemometrics and Intelligent Laboratory Systems. – 2004. – № 72. – P. 123–132.

[3] Crosilla Fabio. Procrustes Analysis and Geodetic Sciences. Technical report. Part 1. – Stuttgart, Germamy: Univ. of Stuttgart, Dept. of Geodesy and Geoinformatics. – 1999. – P. 69–78.

[4] Муха В.С. Линейное прокрустово преобразование двухмерных матриц // Информатика. – № 3. – 2010. – С. 97–102.

[5] Муха В.С. Линейное прокрустово преобразование в распознавании букв // Автоматика и вычислительная техника. – 2012. – № 3. – С. 36 – 48.

[6] Корнеев А.П., Иванова А.А., Прокди Р.Г. Программа FineReader. Руководство. – Санкт-Петербург: Наука и техника, 2010. – 83 с.