Алгоритмы кластеризации образов символов

Н. В. Котович

В статье описаны алгоритмы кластеризации образов распознанных печатных символов, используемые в механизме адаптивного распознавания документа. Перечислены известные методы кластеризации объектов. Описан алгоритм цепной развертки, приведены доказательства его работоспособности, состоящие в ряде положений о произвольности выбора начального объекта кластеризации. Обсуждается сложность алгоритма цепной развертки и приемы оптимизации сложности. Рассматривается вопрос выбора функции расстояния. Обсуждаются приемы оптимизации вычисления функции расстояния, являющейся псевдосимметрикой Хаусдорфа.

1. Кластеризация

Кластеризация символов, распознанных шрифтонезависимым алгоритмом, занимает важное место в схеме адаптивного распознавания, описанной в [1].

Известно множество разнообразных методов кластеризации [2–8]. Наибольшее распространение получили две группы методов кластерного анализа: иерархические агломеративные методы и итеративные методы группировки.

В агломеративно-иерархических методах первоначально все объекты рассматриваются как отдельные, самостоятельные кластеры, состоящие всего лишь из одного элемента. Процедура кластеризации состоит в постепенном объединении объектов в достаточно большие кластеры, используя некоторую меру сходства или расстояние между объектами. Типичным результатом такой кластеризации является иерархическое дерево. На первом

шаге, когда каждый объект представляет собой отдельный кластер, расстояния между этими объектами определяются выбранной мерой. Однако когда связываются вместе несколько объектов, возникает вопрос, как следует определить расстояния между кластерами?

Существует множество методов и алгоритмов объединения кластеров, перечислим некоторые из них.

Одиночная связь (*метод ближайшего соседа*). Расстояние между двумя кластерами определяется расстоянием между двумя наиболее близкими объектами (ближайшими соседями) в различных кластерах.

Полная связь (*метод наиболее удаленных соседей*). В этом случае расстояние между кластерами определяется наибольшим расстоянием между любыми двумя объектами в различных кластерах (т. е. «наиболее удаленными соседями»).

Невзвешенное попарное среднее. Здесь расстояние между двумя различными кластерами вычисляется как среднее расстояние между всеми парами объектов в них.

Взвешенное попарное среднее. Метод идентичен методу невзвешенного попарного среднего, за исключением того, что при вычислениях размер соответствующих кластеров (т. е. число объектов, содержащихся в них) используется в качестве весового коэффициента.

Невзвешенный центроидный метод. В этом методе расстояние между двумя кластерами определяется как расстояние между их центрами тяжести.

Взвешенный центроидный метод (*медиана*). Этот вариант идентичен предыдущему, за исключением того, что при вычислениях используются веса для учета разницы между размерами кластеров (т. е. числами объектов в них).

Метод Варда. Этот алгоритм использует методы дисперсионного анализа для оценки расстояний между кластерами. Метод минимизирует сумму квадратов расстояний до центров кластеров. Сначала во всех уже имеющихся кластерах производится расчет средних значений переменных. Затем вычисляются квадраты евклидовых расстояний от отдельных представителей каждого кластера до этого кластерного среднего значения. Эти расстояния суммируются. В один новый кластер объединяются те кластеры, при объединении которых получается наименьший прирост общей суммы расстояний.

Вопрос, который естественно возникает в агломеративных методах — когда прекратить объединение кластеров. Для этого используют число полученных кластеров, межкластерное расстояние, максимальный скачок в изменении межкластерного расстояния. Также используются основные

статистические характеристики кластеров, как количество объектов в кластере, средние значения признаков в каждом кластере, дисперсии и т. д.

Кроме объединяющих методов иерархической кластеризации существуют и противоположные методы — дивизимные, в которых на начальном этапе вся выборка рассматривается как единый кластер, а затем уже начинается процесс его деления на составляющие части. Процесс деления продолжается до тех пор, пока каждое наблюдение не превратится в отдельный кластер. В свою очередь дивизимные алгоритмы делятся на монотетические и политетические. В монотетической классификации деление производится на основании единственного признака, имеющего максимальную информативность. В политетических же алгоритмах учитываются все признаки. Поскольку данные алгоритмы оперируют расстояниями между наблюдениями, то в некоторых программах предусмотрена возможность работы не с исходной матрицей «объект—признак», а с симметричной матрицей расстояний между наблюдениями.

Кроме иерархических методов классификации большое распространение получили также различные итерационные процедуры, которые пытаются найти наилучшее разбиение, ориентируясь на заданный критерий оптимизации, не строя при этом полного дерева. В начале последовательных итераций в качестве центра выбирается один из элементов и формируется кластер из элементов, удаленных от него не далее чем на **r**. Далее процедура повторяется для остальных элементов, причем в качестве очередного центра выбирается, например, «типическая» лежащая на минимальном расстоянии от центра оставшегося множества — точка объектов. После выполнения очередного шага выясняется, достигнуто ли желательное разбиение. Существуют различные методы определения критерия остановки процедуры:

— получено определенное заранее количество кластеров;
— все кластеры содержат более определенного числа элементов;
— кластеры обладают требуемым соотношением внутренней однородности и разнородности между собой.

Среди итерационных методов наиболее популярным методом является метод k-средних Мак-Кина. В отличие от иерархических методов сам пользователь должен задать искомое число конечных кластеров, которое обычно обозначается как k. Как и в иерархических методах кластеризации, при этом можно выбрать тот или иной тип метрики. Разные алгоритмы метода k-средних отличаются и способом выбора начальных центров задаваемых кластеров.

В некоторых вариантах метода сам пользователь может (или должен) задать такие начальные точки, либо выбрав их из реальных наблюдений,

либо задав координаты этих точек по каждой из переменных. В других реализациях этого метода выбор заданного числа k начальных точек производится случайным образом, причем эти начальные точки (зерна кластеров) могут в последующем уточняться в несколько этапов. Можно выделить 4 основных этапа таких методов:

— выбираются или назначаются k наблюдений, которые будут первичными центрами кластеров;
— при необходимости формируются промежуточные кластеры приписыванием каждого наблюдения к ближайшим заданным кластерным центрам;
— после назначения всех наблюдений отдельным кластерам производится замена первичных кластерных центров на кластерные средние;
— предыдущая итерация повторяется до тех пор, пока изменения координат кластерных центров не станут минимальными.

В некоторых вариантах этого метода пользователь может задать числовое значение критерия, трактуемого как минимальное расстояние для отбора новых центров кластеров. Наблюдение не будет рассматриваться как претендент на новый центр кластера, если его расстояние до заменяемого центра кластера превышает заданное число. Такой параметр в ряде программ называется радиусом. Кроме этого параметра возможно задание и максимального числа итераций либо достижения определенного, обычно достаточно малого, числа, с которым сравнивается изменение расстояния для всех кластерных центров. Этот параметр обычно называется *конвергенцией*, так как отражает сходимость итерационного процесса кластеризации.

Далеко не все из приведенных алгоритмов кластеризации пригодны для использования в автоматическом адаптивном распознавании. Конечно, непригодны методы, в которых требуется вмешательство пользователя, поскольку в этом случае невозможно получить полную автоматизацию. Итерационные методы кластеризации в нашем случае имеют свои недостатки — при их использовании невозможно оценить время кластеризации, а в случае автоматического адаптивного распознавания текста требуется кластеризация в реальном времени. В качестве метода кластеризации, удобного для использования в указанной схеме адаптивного распознавания, представляется метод цепной развертки.

2. Алгоритм цепной развертки

Алгоритм цепной развертки, рекомендованный Щепиным Е. В., относится к группе методов одиночной связи. Алгоритм кластеризации,

приведенный в [10] качественно отличается от широко известных методов, прежде всего, из-за структуры информации, подлежащей классификации во время обучения и целей кластеризации. Минимизация числа кластеров, высокое «качество» кластеров и другие вопросы классической кластеризации имеют для нас важное, но не первостепенное значение. Цепная развертка может выполняться с большой скоростью, что очень актуально для адаптивного распознавания, кроме того, для проведения развертки достаточно вычислять расстояния только между отдельными объектами, не требуется дополнительный анализ кластеров, межкластерного расстояния и т. п.

Напомним определение цепного расстояния. Пусть в пространстве объектов задано некоторое расстояние $d(x, y)$. Это расстояние не обязано быть метрикой или псевдометрикой, а может быть произвольной неотрицательной симметричной функцией с условием $d(x, x) = 0$ для $\forall\, x$. Таким образом, в пространстве объектов задана некоторая псевдосимметрика. Возьмем две произвольные точки (обозначим их p_1, p_2) в этом пространстве. Рассмотрим всевозможные конечные цепочки точек $q_1, q_2, .., q_n$, $n \geqslant 2$. Обозначим d_i — расстояние от q_i до q_{i+1}, т. е. $d_i = d(q_i, q_{i+1})$. Обозначим $d_{q_1 \ldots q_n}$ — максимальное число из $d_1, d_2, .., d_{n-1}$. Тогда цепное расстояние между точками p_1 и p_2 $Ch(p_1, p_2)$ — это нижняя грань $d_{q_1 \ldots q_n}$ по всевозможным цепочкам $q_1, q_2, .., q_n$, таким, что $q_1 = p_1$, $q_n = p_2$.

Опишем метод цепной развертки. В этом методе в качестве исходного объекта берется любой объект из предъявленной совокупности, ему приписывается номер 1 и расстояние 0. Затем просматриваются все оставшиеся объекты. Выбирается объект, расстояние от которого до исходного элемента минимально. Ему присваивается номер 2 и расстояние, равное расстоянию до исходного объекта. Затем среди оставшихся ищется объект, расстояние от которого до уже отмеченного множества объектов из двух элементов минимально, и т. д. — всегда на очередном шаге выбирается объект, расстояние от которого до уже пронумерованных объектов (как расстояние до множества) минимально, ему приписывается очередной номер и это расстояние. Процедура повторяется, пока все объекты не будут пронумерованы.

В результате все объекты будут выстроены в некотором порядке, и каждому объекту приписано некоторое число — расстояние до предшествующего множества.

Наша цель — разделить исходное множество на несколько кластеров таким образом, чтобы цепное расстояние между любыми объектами, входящими в разные кластеры, было больше заданного расстояния d_0, а для любых объектов из одного кластера цепное расстояние было не больше

d_0. Как будет показано ниже, для этого достаточно просто просмотреть все приписанные объектам расстояния и пометить те из них, которые больше d_0. Пусть это будут номера $N_1, N_2, ..., N_k$. Тогда к первому кластеру отнесем все объекты с номерами меньше N_1, ко второму все объекты с номерами от N_1 до N_2 и т. д. Обозначим такое разбиение $K(d_0)$, а сами полученные таким образом кластеры обозначим $K_1(d_0)$, $K_2(d_0)$ и т. д.

Докажем следующие утверждения.

Предложение 1. *Пусть построена цепная развертка и имеется разбиение $K(d_0)$ на N кластеров $K_1(d_0), \ldots, K_N(d_0)$. Тогда для $\forall\, i, i \in \{1, .., N\}$ для $\forall\, p_1, p_2$ таких, что $p_1 \in K_i(d_0)$, $p_2 \in K_i(d_0)$ цепное расстояние $Ch(p_1, p_2) \leqslant d_0$.*

Доказательство. Пусть кластеру $K_i(d_0)$ принадлежат объекты с номерами от N_i до N_{i+1}.

Проведем доказательство по индукции по максимальному из номеров p_1, p_2. Не нарушая общности можно считать, что $p_1 \leqslant p_2$. Если $p_1 = p_2 = N_i$ — утверждение очевидно. Пусть утверждение верно для всех $p_1, p_2 < M$. Докажем утверждение для $p_1 \leqslant M, p_2 = M$, где $M > N_i$. Если $p_1 = p_2$ — утверждение очевидно. Поэтому будем считать $p_1 < p_2$.

Поскольку $p_2 > N_i$ и $p_2 \in K_i(d_0)$, то по построению цепной развертки существует некоторый объект с номером $p_3, p_3 \in K_i(d_0), p_3 < p_2$ такой, что $d(p_3, p_2) \leqslant d_0$. Поскольку $p_3 < M, p_1 < M$, то по предположению индукции цепное расстояние $Ch(p_1, p_3) \leqslant d_0$. Значит существует цепочка объектов $q_1, q_2, .., q_n$ такая, что $q_1 = p_1, q_n = p_3$ и $d(q_i, q_{i+1}) \leqslant d_0$ для $\forall\, i < n$. Добавим к цепочке объектов точку $q_{n+1} = p_2$. Тогда $d(q_i, q_{i+1}) \leqslant d_0$ для $\forall\, i < n+1$, так как $d(p_3, p_2) \leqslant d_0$, при этом $q_1 = p_1$, $q_{n+1} = p_2$, и значит $Ch(p_1, p_2) \leqslant d_0$, что и требовалось доказать. □

Предложение 2. *Пусть построена цепная развертка и имеется разбиение $K(d_0)$ на N кластеров $K_1(d_0), \ldots, K_N(d_0)$. Тогда если для некоторых p_1, p_2 цепное расстояние $Ch(p_1, p_2) \leqslant d_0$, то $\exists\, i, i \in \{1, .., N\}$ такое, что $p_1 \in K_i(d_0)$, $p_2 \in K_i(d_0)$.*

Доказательство. Проведем доказательство по индукции по длине n цепочки $q_1, q_2, .., q_n$ такой, что $q_1 = p_1, q_n = p_2$ и $d(q_i, q_{i+1}) \leqslant d_0$ для $\forall\, i < n$. Такая цепочка существует по определению цепного расстояния. Если $p_1 = p_2$, т. е. длина цепочки равна 1 — утверждение очевидно. Рассмотрим случай цепочки длины 2. В этом случае $d(q_1, q_2) = d(p_1, p_2) \leqslant d_0$, то есть обычное расстояние между объектами не больше d_0, и значит $\exists\, i, i \in \{1, .., N\}$ такое, что $p_1 \in K_i(d_0), p_2 \in K_i(d_0)$ по построению цепной развертки.

Перейдем к общему случаю. Пусть утверждение верно для всех цепочек длины меньше M. Докажем утверждение для цепочки длины M, $M > 2$. Имеется цепочка $q_1, q_2, .., q_M., q_1 = p_1, q_M = p_2, d(q_i, q_{i+1}) \leqslant d_0$ для $\forall\, i < M$.

$Ch(p_1, p_2) \leqslant d_0$, значит $Ch(q_1, q_2) \leqslant d_0$ и $Ch(q_2, q_M) \leqslant d_0$. Длина цепочки q_1, q_2 равна 2, значит $\exists\, k, k \in \{1, .., N\}$ такое, что $q_1 \in K_k(d_0)$, $q_2 \in K_k(d_0)$ по доказанному выше. Длина цепочки q_2, q_M равна $M-1$, значит $\exists\, j, j \in \{1, .., N\}$ такое, что $q_2 \in K_j(d_0), q_M \in K_j(d_0)$ по предположению индукции. А значит $\exists\, i, i \in \{1, .., N\}$ такое, что $q_1 \in K_i(d_0), q_2 \in K_i(d_0)$, $q_M \in K_i(d_0)$.

Поскольку $q_1 = p_1, q_M = p_2$, то утверждение доказано. □

Как следствие из предложений 1, 2 вытекает следующее

Предложение 3. *При заданном d_0 результат кластеризации $K(d_0)$ не зависит от выбора начального объекта при построении цепной развертки.*

Доказательство. Из предложений 1, 2 следует, что кластеры, полученные в результате цепной развертки с произвольным начальным объектом с разбиением по порогу d_0, совпадают с множествами объектов, цепное расстояние между которыми не превышает d_0.

Следовательно, результат кластеризации $K(d_0)$ не зависит от выбора начального объекта при построении цепной развертки, что и требовалось доказать. □

Таким образом, мы видим, что обозначение $K(d_0)$ без учета выбора начального объекта имеет свое основание.

После процедуры цепной развертки также легко проводить анализ, при каких значениях порога d_0 возникают разные варианты кластеризации, как эти варианты соотносятся между собой и многое другое. Но как легко видеть, данная процедура требует $N \cdot (N-1)/2$ процедур вычисления расстояния между объектами, если всего имеется N объектов, поэтому бывает необходимо в связи с повышением быстродействия использовать иные приемы. В качестве примера (как правило) более быстрого варианта кластеризации можно привести модификацию описанного выше метода — кластеризацию с фиксированным порогом. В качестве исходного объекта берется любой символ, ему присваивается принадлежность к первому кластеру. К данному первому кластеру присоединяются все символы, принадлежность которых к какому-либо кластеру еще не установлена, и расстояние от которых до исходного символа меньше порога d_0. Затем для каждого из вновь присоединенных символов данная процедура повторяется. После того как к первому кластеру никто не может

быть больше отнесен, среди символов, которые остались, берется произвольный символ в качестве затравочного для второго кластера и т. д. пока не будут исчерпаны все символы. В худшем случае и здесь при наличии d_0 символов надо использовать $N \cdot (N-1)/2$ процедур вычисления расстояния, но в лучшем случае всего N процедур.

3. Расстояние между символами

Конечно, важнейшую роль в кластеризации играет выбранная функция расстояния, т. е. что понимается под расстоянием между объектами. При разном выборе расстояния естественно возникают разные варианты кластеризации. Какую функцию расстояния использовать применительно к символам? Учитывая то, что к одному и тому же кластеру желательно отнести как хорошо пропечатанные и отсканированные, так и дефектные символы одинакового размера и начертания (как это сделал бы человек), кажется естественным прибегнуть к побитовому (поточечному) сравнению растров. Как видно из описания процедуры кластеризации, она может потребовать много операций вычисления расстояний между различными изображениями символов. Поэтому, кроме того, что выбранное расстояние должно по возможности быть устойчиво к различным дефектам изображения (как уже бывшим в исходном изображении, так и появившимся после сканирования), функция расстояния должна допускать быстрое вычисление.

Если просто подсчитывать число несовпадающих точек в двух разных растрах, то возникает проблема центрирования — смещение даже на одну точку может вызвать большой скачок при вычислении расстояния.

Так, в примере на рис. 1 расстояние между символами (первая и вторая картинка слева) при простом наложении сильно изменяется при сдвиге всего на 1 пиксел. При наложении с использованием правой и верхней границ (вторая картинка справа) расстояние между символами (число несовпадающих пикселов) равно 17, в то время как при наложении с использованием левой и верхней границ (первая картинка справа)

Рис. 1. Скачок расстояния при сдвиге (простое наложение)

расстояние (число несовпадающих пикселов) равно 47. Изменение положения при этом составляет всего один пиксел по горизонтали.

Кроме того, случайные изменения положения ряда черных точек, которые постоянно возникают при сканировании даже текстов хорошего качества, не говоря о текстах плохого качества, также могут вызывать большие изменения при вычислении расстояния простым сравнением.

В качестве примера можно обратиться к рис. 2, где при вычислении расстояния простым наложением полученное значение велико, хотя символы очень похожи.

Поэтому представляется разумным вычислять расстояние между символами, основываясь на метрике Хаусдорфа. Так, на рис. 1 расстояние Хаусдорфа между символами равно единице при обоих способах наложения картинок, то есть существенно более стабильно при небольшом изменении взаимного расположения.

Кратко напомним определение метрики Хаусдорфа. Пусть в некотором пространстве определено расстояние между точками, обозначим его $d(x, y)$. Тогда расстояние $d(x, Y)$ от точки x до множества Y определяется как нижняя грань расстояний $d(x, y)$ для y из Y. Расстояние $d_1(X, Y)$ от множества X до множества Y определяется как верхняя грань расстояний $d(x, Y)$ для всех x из X. Расстояние Хаусдорфа $d(X, Y)$ между множествами X, Y определяется как максимум расстояний от X до Y и от Y до X, т. е.

$$d(X, Y) = \max\left(d_1(X, Y), d_1(Y, X)\right).$$

Таким образом, разные множества в метрике Хаусдорфа близки, если и только если для любой точки из одного множества в ее малой окрестности содержится хотя бы одна точка другого множества (и то же самое для другого множества).

Рис. 2. Пример простого наложения символов (число несовпадающих точек у двух символов хорошего качества одного шрифта при наложении друг на друга велико — равно 19, т. е. порядка высоты символа)

В случае символов можно рассматривать расстояние между символами как расстояние Хаусдорфа между множествами точек, составляющих разные символы.

Однако, хорошо известно, что собственно метрика Хаусдорфа неустойчива к помехам: появление даже одной «лишней» точки на большом расстоянии от растра приводит к большому скачку при вычислении расстояния. Но в то же время данная метрика прекрасно справляется с особенностями, возникающими вблизи растра или непосредственно в самом растре — добавление точек («бахрома») или исчезновение некоторых точек. Поэтому для вычисления расстояния между растрами можно использовать псевдо-*симметрику Хаусдорфа. В качестве расстояния между двумя растрами берется сумма числа точек первого растра, выходящих за единичную окрестность второго растра, и числа точек второго растра, выходящих за единичную окрестность первого растра. Если размер символов очень велик, то величину окрестности можно увеличить. Но при стандартных разрешениях 200–300–400 точек на дюйм и обычных шрифтах единичная окрестность представляется наиболее подходящей. Таким образом, мы объединяем в один кластер не только символы, расстояние Хаусдорфа между которыми не больше 1, но и символы, у которых расстояние Хаусдорфа больше 1, но расстояние меньше или равно единицы для почти всех точек, составляющих символы.

Необходимо заметить, что непосредственное вычисление расстояния Хаусдорфа требует слишком много вычислений даже для проверки того, что это расстояние между символами больше или не больше единицы. Для того чтобы проверить все точки растра, надо проверить каждую точку: черная она или белая, (т. е. принадлежит символу или нет). Если точка принадлежит символу, надо проверить до восьми ее соседей. То есть при средней заполненности растров на четверть и при размерах растров m на n может потребоваться до

$$2 \cdot \left(m \cdot n \cdot \frac{3}{4} + m \cdot n \cdot \frac{9}{4} \right) = 6 \cdot m \cdot n \quad \text{элементарных операций.}$$

Поэтому полезно использовать некоторые приемы для оценки расстояния, основанного на метрике Хаусдорфа. Для эффективного вычисления расстояния между символами используется следующий прием. Для каждого символа помимо собственно изображения создается растр, в котором кроме исходных точек содержатся также точки окрестности (*расширенный образ*). После чего вычисление расстояния между символами сводится к простому выявлению пикселей одного растра, которые не вошли в «расширенный» образ другого символа. Таким образом, требуется одна

операция для одного байта (для восьми точек) растра — «исключающее или», или две операции — если надо узнать и количество пикселей, выходящих за рамки единичной (или иной) окрестности. Следовательно, надо не больше $2 \cdot (m \cdot n \cdot 2/8) = m \cdot n/2$ элементарных операций.

Заключение

Описанные алгоритмы адаптивного распознавания реализованы OCR Cuneifrom [10] и системе массового ввода документов Cognitive Forms [11], учитывающих особенности используемых в документе шрифтов и их модификаций. Результатом адаптации является уверенное и надежное распознавание образов, невозможное шрифтонезависимыми методами.

Литература

1. *Арлазаров В. Л.*, *Астахов А. Д.*, *Трояпкер В. В.*, *Котович Н. В.* Адаптивное распознавание символов // Интеллектуальные технологии ввода и обработки информации: Труды ИСА РАН. М.: URSS, 1998. С. 39–56.
2. *Duda R.*, *Stock D.*, *Hart P.* Pattern Classification. Wiley: John & sons, 1999. 680 p.
3. *Горелик А. Л.*, *Скрипкин В. А.* Методы распознавания. 2-е изд. М.: Высшая школа, 1984. 219 с.
4. *Theodoridis S.*, *Koutroumbas K.* Pattern Recognition. New York: Academic Press, 1998. 591 p.
5. *Nabney I. T.* Netlab: Algorithms for Pattern Recognition. New York: Springer-Verlag, 2001. 440 p.
6. *Y-H Pao*. Adaptive pattern recognition and neural network. Addison-Wesley, 1989.
7. *Tryon R. C.* Cluster Analysis. New York: McGraw-Hill, 1939.
8. *Everett B.*, *Landau S.*, *Leese M.* Cluster Analysis. Oxford University Press. 2001. 224 p.
9. *Арлазаров В. Л.*, *Котович Н. В.*, *Славин О. А.* Адаптивное распознавание // Информационные технологии и вычислительные системы. 2002. № 4. С. 11–22.
10. *Арлазаров В. Л.*, *Славин О. А.* Алгоритмы распознавания и технологии ввода текстов в ЭВМ // Информационные технологии и вычислительные системы. 1996. № 1. С. 48–54.
11. *Арлазаров В. Л.*, *Постников В. В.*, *Шоломов Д. Л.* Cognitive Forms — система массового ввода структурированных документов // Управление информационными потоками: Труды ИСА РАН. М.: URSS, 2002. С. 35–46.