Алгоритм адаптивного формирования эталонных образов печатных символов

О. А. Славин

Институт системного анализа Российской академии наук,
Россия, 117312 Москва, пр. 60-летия Октября, 9

В работе рассмотрен алгоритм сравнения с эталонами с обучением без учителя на основе кластеризации частично распознанных образов символов печатного документа. Описан способ извлечения эталонов, позволяющий учесть как случайные искажения образов символов, так и эффекты оцифровки. Исследованы вопросы стабильности извлечения эталонов. Приведены алгоритмы распознавания образов отдельных символов, в том числе комбинированные алгоритмы. Уделено внимание инструментарию исследования описанных в статье задач.

Введение

Механизм адаптивного распознавания отсканированных образов текстовых документов, описанный в [1, 2, 7], состоит из следующих этапов.

На первом этапе (*первом проходе распознавания*) происходит распознавание текста одним из шрифтонезависимых методов. В результате работы первого этапа для каждого распознаваемого образа указывается класс, к которому принадлежит образ, и дается некоторая оценка качества распознавания, т. е. надежность принадлежности к выбранному классу.

На втором этапе (*этапе кластеризации*) происходит анализ результатов первого прохода распознавания. Среди надежно распознанных образов каждого класса проводится кластеризация.

На третьем этапе (*этапе самообучения*) полученные кластеры анализируются. Проводится поиск использованных шрифтов, отбор кластеров, обладающих наиболее благоприятными характеристиками — большей мощностью, лучшей надежностью распознавания составляющих объектов.

Для каждого отобранного кластера необходимо построить *эталон*, наилучшим образом характеризующий элементы кластера. В дальнейшем эталоны будут использованы в алгоритме распознавания, который является обучается без учителя с помощью кластеризации.

В статье рассматриваются вопросы суммирования бинарных образов элементов, составляющих кластер, формирование эталонов и их использование в последующем распознавании.

1. Суммирование элементов кластера

Кластеризации подвергаются предварительно распознанные бинарные образы, каждый из которых обладает следующими характеристиками:

- имя (код) символов;
- размеры образа;
- образ (бинарный растр) $R(m,n) = \left\| r_{ij} \right\|$, где $i = \overline{1,m}$, $j = \overline{1,n}$, $r_{ij} \in \{0, 1\}$, в котором 1 соответствует черной точке, а 0 — белой;
- коллекция альтернатив распознавания, упорядоченных по убыванию оценки надежности распознавания;
- признак подтвержденного лингвистическим механизмом слова, в которое входит данных символ (берутся слова с длиной не менее четырех букв).

Кластер состоит из одного или нескольких элементов $S_1, \ldots, S_n$, каждый из которых был предварительно распознан шрифтозависимым методом, и обладает следующими характеристиками:

- имя (код символов) кластера;
- кегль кластера;
- признаки графем (признаки жирности, серифности, наклона);
- код шрифта.

Мощностью кластера $\mu(Cl)$ будем называть количество элементов, составивших кластер.

Определим для сетки с общими для всех кластеров размерами M и N (мы использовали сетку с размерами 256 × 128) *центрированный* образ $\tilde{R}(M,N) = \left\| \tilde{r}_{ij} \right\|$ следующим образом:

$$\tilde{r}_{(i-s_1),(j-s_2)} = r_{ij}, \text{ если } s_1 \le i < s_1 + m\text{–}1 \text{ и } s_2 \le j < s_2 + n\text{–}1,$$
$$\tilde{r}_{ij} = 0, \text{ если } i < s_1, \text{ или } i \ge s_1 + m, \text{ или } j < s_2, \text{ или } j \ge s_2 + n,$$

где $\|r_{ij}\|$ — образ исходного элемента с размерами m и n,

$$s_1 = [(M-m)/2], \; s_2 = [(N-n)/2],$$

причем размеры M и N являются общими для всех кластеров.

Пару (i, j) будем называть *точкой* образа, а p_{ij} — значением точки (i, j). Если $p_{ij} > 0$, такие точки будем называть *ненулевыми*.

Определим функцию расстояния d между двумя элементами кластера, удовлетворяющую трем свойствам:

$$d(A, B) \geq 0 \quad \forall\, A, B,$$
$$d(A, A) = 0 \quad \forall\, A,$$
$$d(A, B) = d(B, A) \quad \forall\, A, B.$$

Опишем способ построения симметрики d, приведенный в [2]. Рассмотрим единичную окрестность образа $R(m, n) = \left\| r_{ij} \right\|$, где $i = \overline{1,m}$, $j = \overline{1,n}$, $r_{ij} \in \{0,1\}$, т. е. $N^{(1)}(R)(m, n) = \| N_{ij}^{(1)}(R) \|$, где

$$N_{ij}^{(1)}(R) = \max(r_{(i-1)(j-1)}, r_{(i-1)j}, r_{(i-1)(j+1)}, r_{i(j-1)}, r_{i(j+1)}, r_{(i+1)(j-1)}, r_{(i+1)j}, r_{(i+1)(j+1)}),$$

если $0 < i < m-1$ и $0 < j < n-1$,

$$N_{ij}^{(1)}(R) = r_{ij}, \text{ если } i = 0, \text{ или } i = m-1, \text{ или } j = 0, \text{ или } j = n-1.$$

В качестве расстояния между двумя образами A и B берется сумма числа точек первого образа A, выходящих за единичную окрестность второго образа $N^{(1)}(B)$, и числа точек второго образа B, выходящих за единичную окрестность первого образа $N^{(1)}(A)$:

$$d(A, B) = \sum_{i=1}^{m} \sum_{j=1}^{n} (a_{ij} \wedge \overline{b}_{ij}^{(1)} + b_{ij} \wedge \overline{a}_{ij}^{(1)}) .$$

Перед началом суммирования бинарных образов выбирается *опорный элемент* S_0 кластера как лучший по оценке распознавания среди подтвержденных словарем. Если таких элементов несколько, то выбирается произвольный из них. Обоснование возможности произвольного выбора опорного элемента из нескольких возможных будет приведено ниже. Опорный элемент S_0 сохраняется в центре сетки $M \times N$.

При суммировании положение любого из остальных элементов S_X выбирается таким образом, чтобы расстояние $d(S_X, S_0)$ между этими элементами было наименьшим, где d — расстояние между образами растров. То есть помимо центрированного положения вычисление $d(S_X, S_0)$ проводится для нескольких взаимных положений элементов S_X и S_0 при сдвигах образа элемента $S_X = \|r_{ij}\|$ на единицу в разных направлениях:

$$S_h^{(\pm 1)} = \|r_{i\pm 1, j}\|, \; S_v^{(\pm 1)} = \|r_{i, j\pm 1}\|, \; S_{hv}^{(\pm 1)} = \|r_{i\pm 1, j\pm 1}\|.$$

В качестве расстояния берется минимальная величина из полученных значений $d_{min}(S_X, S_0) = \min(d(S_X, S_0), d(S_h^{(\pm 1)}, S_0), d(S_v^{(\pm 1)}, S_0), d(S_{hv}^{(\pm 1)}, S_0))$.

После центрирования становится возможным определить *образ кластера* $P(Cl) = \|p_{ij}\|$ как сумму бинарных центрированных образов элементов $S_1, \ldots, S_n$, составивших кластер Cl. Значение p_{ij} точки образа кластера является количеством элементов, в центрированных образах которых точка (i, j) ненулевая.

Будем также использовать понятие *порогового образа* кластера, определяемого следующим образом:

$$Tr(T) = \begin{cases} p_{ij}, & \text{если } p_{ij} \geq T, \\ 0, & \text{если } p_{ij} < T, \end{cases}$$

где T — порог, не превосходящий *мощности кластера*, т. е. числа составивших его образов. Пороговый образ $Tr(\mu(Cl))$ будем называть *общей областью* кластера.

После суммирования достаточно большого количества образов мы можем наблюдать структуру кластера, а именно, в кластере можно выделить точки, общие для всех растров, и точки границы кластера.

На рис. 1 приведен образ кластера, который будет рассматриваться как пример для иллюстрации утверждений настоящей статьи. Кластер содержит 72 элемента символов «я» бессерифного шрифта. Очевидно, что в образе этого кластера присутствуют общие точки и граница.

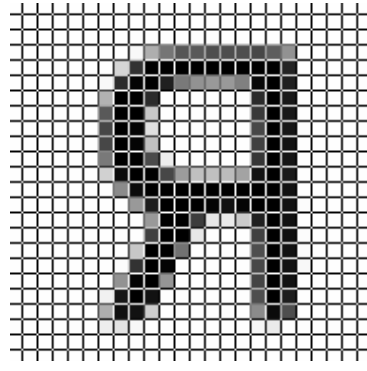
Прежде всего отметим, что площадь общей области A_C, т. е. количество ненулевых точек порогового образа $Tr(\mu(Cl))$, состоящего из точек, общих для всех 72 элементов, составляет $S_\wedge = 57$, в то время как общая площадь множества порогового образа $Tr(1)$ составляет $S_\vee = 170$, то есть площадь $S_\wedge$ составляет около 35 % от площади $S_\vee$, что иллюстрируется рис. 2. Второй пороговый образ соответствует уровню

$$T_1 = \frac{\sum\limits_{i=0, j=0}^{i<128, j<256} \tilde{r}_{ij}}{\sum\limits_{i=0, j=0}^{i<128, j<256} (\tilde{r}_{ij} > 0)} = 54,$$

его площадь равняется 108.

Таким образом, площадь границы является значительной по отношению к общей площади кластера $S_\vee$.

Функция распределения значений точек рассматриваемого кластера приведена на рис. 3. Ось n соответствует диапазону, в котором находятся значения точек образа кластера p_{ij}, а на оси $v(n)$ откладываются частоты появления значений точек.



		2	23	39	48	47	50	50	50	52	46	31	
	12	58	72	72	72	72	72	72	72	72	72	62	
2	60	71	72	63	38	31	29	29	36	70	72	63	
22	72	72	60							60	72	65	
47	72	72	16							56	72	65	
52	72	71	9							52	72	66	1
52	72	71	18							54	72	67	
38	72	72	51	5						57	72	66	
13	69	72	72	52	28	18	18	19	26	67	72	65	
	33	69	72	72	72	72	72	72	72	72	72	67	
		29	64	72	72	72	71	67	71	72	72	67	1
			29	72	72	55	7	6	15	63	72	67	
			54	72	72	7				54	72	68	1
		16	72	72	40					50	72	67	1
	1	58	72	71						52	72	67	2
	31	71	72	30						53	72	67	
4	66	72	67							50	72	64	
23	68	67	14							33	69	50	
5	8	2								3	6		

Рис. 1. Пример суммы образов

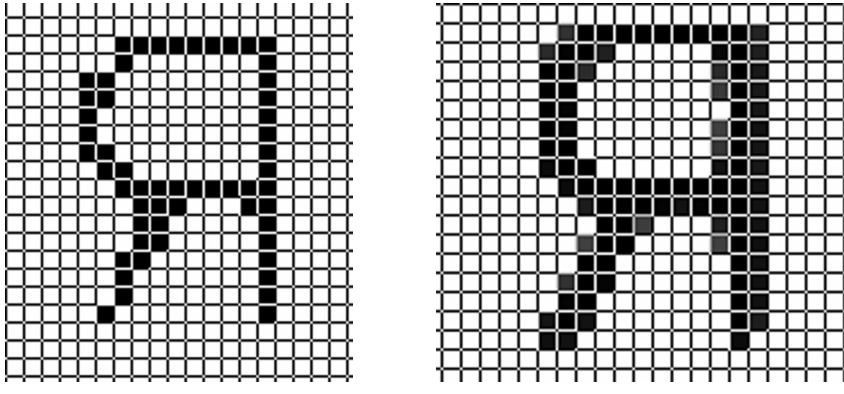


Рис. 2. Пороговые образы *Tr*(72) и *Tr*(54)

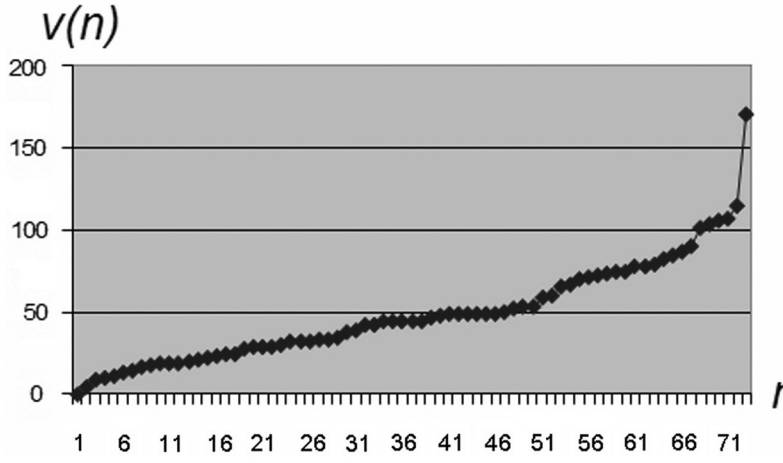


Рис. 3. Функция распределение значений точек кластера

Причинами образования границы образа кластера с указанным распределением значений точек являются:

- искажения образов букв при печати документов (загрязнение стекла сканера) и дефокусировка (на сенсор сканера попадает уже искаженное отражение);
- аппаратные искажения при сканировании (эффекты оцифровки).

Искажения подробно рассмотрены в работе [3] и сводятся к случайным искажениям границы единичных символов и к случайному зашумлению границы всех символов, хорошо моделируемому с помощью розового шума.

Рассмотрим эффект оцифровки, вносящий основной вклад в появление границы кластера. Например, пусть мы хотим построить сумму отсканированных образов вертикального отрезка прямой шириной $w = 0{,}2$ мм, ориентируясь на сканирование с разрешением 300 DPI (т. е. 300 точек на дюйм). При этом каждая точка отсканированного изображения соответствует на бумаге квадрату размером 85×85 микрон. Очевидно, что никакая фигура на сетке сканера не соответствует точно идеальному отрезку.

При совпадении левой границы отрезка с границей сетки сканера (см. рис. 4) мы получим вертикальный отрезок шириной в две точки. Однако напечатанный символ при сканировании даст другой результат, если сетка наложится на него как на рис. 5, и в результате получается вертикальный отрезок шириной в три точки.

Таким образом, необходимо будет суммировать отсканированные образы шириной как 2, так и 3 точки, что приведет к образованию границы кластера $P(Cl)$.

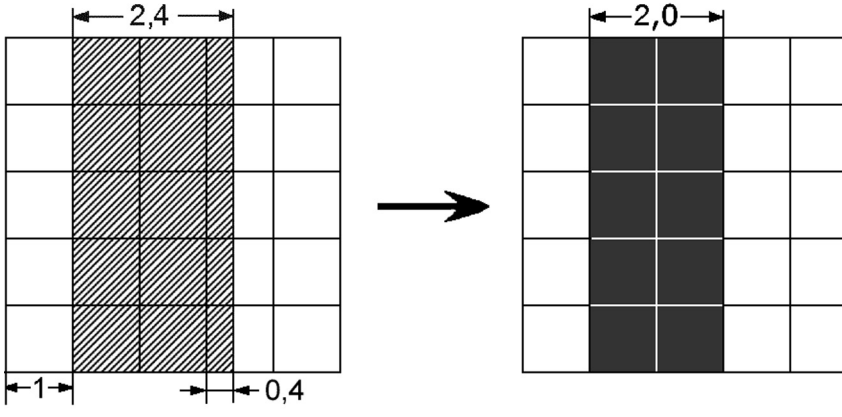


Рис. 4. Пример оцифровки образа

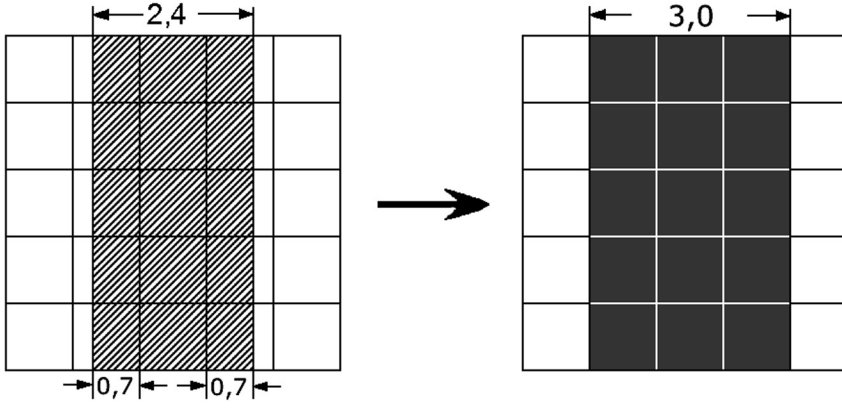


Рис. 5. Другой вариант оцифровки образа при сдвиге сетки

Опишем модель оцифровки графического образа. Пусть $X(L, Q) = ||X_{lq}||$, $l = \overline{1,L}$, $q = \overline{1,Q}$, $X_{lq} \in \{0,1\}$ — оригинальный бинарный образ. Определим процедуру «сжатие образа» в образ с меньшим масштабом $m{\cdot}n$ ($m < L$, $n < Q$). Сначала преобразуем образ $X(L, Q)$ в бинарный образ $X^{(1)}(m{\cdot}L, n{\cdot}Q)$ по следующему правилу:

$$X_{kv}^{(1)} = X_{lq}, \text{ где } m(l-1) \le k < m{\cdot}l \text{ и } n(q-1) \le v < n{\cdot}q.$$

Вычислим образ $X^{(2)}(m, n)$ следующим методом:

$$X_{ij}^{(2)} = \sum_{k=m(i-1)+1}^{mi} \sum_{v=n(j-1)+1}^{nj} X_{kv}^{(1)}$$

Определим *свертку* как бинарный образ $x(m, n) = ||x_{ij}||$, извлекаемый из образа $X^{(2)}$:

$$x_{ij} = \begin{cases} 1, & \text{если} \quad X^{(2)}{}_{ij} \geq L \cdot Q / 2, \\ 0, & \text{если} \quad X^{(2)}{}_{ij} < L \cdot Q / 2. \end{cases}$$

В случае кратных значений размеров, то есть $\exists\ m_1, n_1$: $L = m \cdot m_1$, $Q = n \cdot n_1$, вычисление промежуточного образа $X^{(1)}$ становится ненужным.

Оцифровку в процессе сканирования будем моделировать процедурой сжатия образов с уменьшением их размеров.

Проведем численный эксперимент, задавшись бинарным образом с размерами 256 × 256 (см. рис. 6, на котором приведен образ символа «я» бессерифного шрифта), который будем сжимать до размеров 32 × 32, то есть размеры образов кратны и $m_1 = n_1 = 8$.

Произведем 64 сдвига исходного образа (8 сдвигов по горизонтали и 8 — по вертикали), каждый из сдвиговых образов сожмем. Получим последовательность из 64 элементов, соответствующих исходному образу символа «я», которую будем рассматривать как один кластер. Сложим 64 элемента

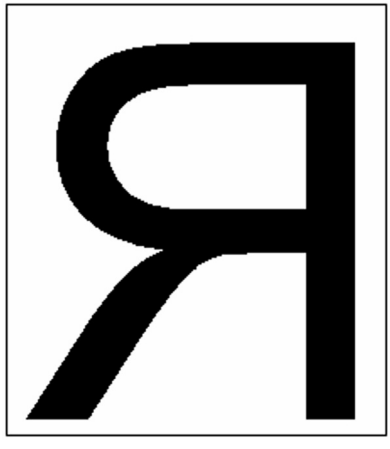


Рис. 6. Бинарный образ для сжатия

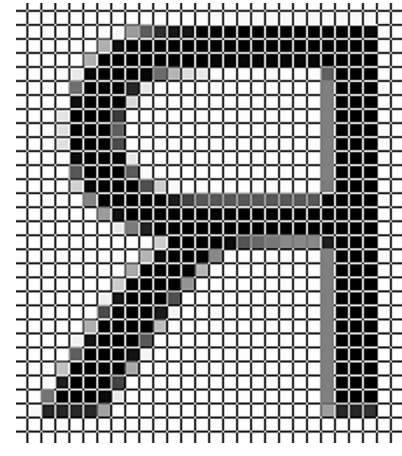


Рис. 7. Сумма образов сверток бинарного образа

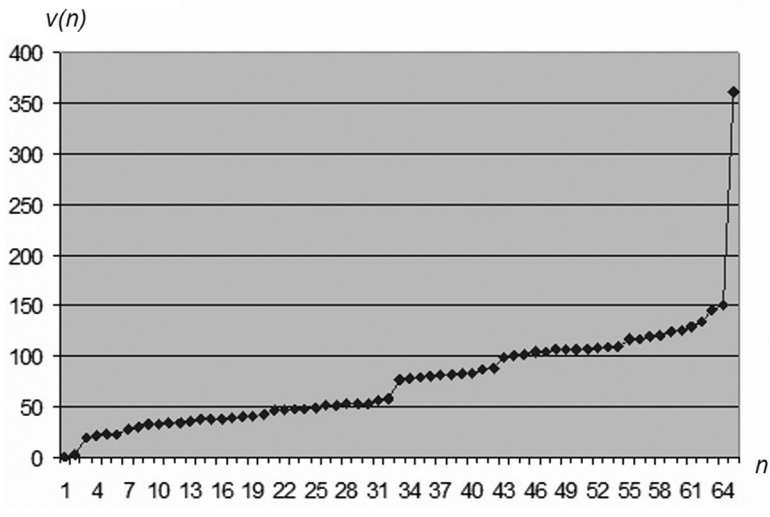


Рис. 8. Функция распределения значений точек кластера сверток

по описанной выше процедуре, взяв в качестве опорного элемента первую свертку. Рассмотрим получившуюся сумму и соответствующую ей график функции распределения значений точек образа кластера (см. рис. 7 и 8).

Проведенный эксперимент иллюстрирует объяснение появления границы суммы образов как следствие оцифровки.

2. Модель образа кластера

Опишем границу кластера, состоящего из n элементов, как совокупность слоев L_1, L_2…, каждый из которых содержит точки образа кластера $P(Cl)$ с одинаковым расстоянием Хаусдорфа d_H до общей области:

$$L_q = \{r_{ij} \mid r_{ij} \in P(Cl),\ d_H(r_{ij}, A_C) = q\}$$

где $d_H(x, Y) = \min d_2(x, y)$, $y \in Y$; d_2 — некоторая функция расстояния в $\mathbf{R}^2$ (использовалась метрика Евклида).

Образ кластера можно представить в форме

$$P(Cl) = A_C \cup L_1 \cup L_2 \cup \ldots .$$

Для рассматриваемого примера существует ровно два слоя L_1 и L_2, содержащих 99 и 15 точек соответственно.

Также будем рассматривать *кортеж укладки*:

$$p(Cl) = \{p_0(\mathrm{Cl}), p_1(\mathrm{Cl}), p_2(\mathrm{Cl}), \ldots\} = \{S_\wedge/S_\vee, L_1/S_\vee, L_2/S_\vee \ldots\}.$$

Таблица 1

Распределение точек в кластера по слоям

Символ	n	S_v	$\lvert A_C \rvert$	$\lvert L_1 \rvert$	$\lvert L_2 \rvert$	$\lvert L_3 \rvert$	$\lvert L_4 \rvert$	$\lvert L_5 \rvert$
р	13	305	131	134	40	0	0	0
а	192	276	46	124	86	18	2	0
е	35	259	119	115	25	0	0	0
д	15	306	138	130	38	0	0	0
и	45	214	87	113	14	0	0	0
я	72	170	56	99	15	0	0	0
п	107	178	39	93	46	0	0	0
о	92	250	77	113	59	1	0	0
ш	24	234	95	120	19	0	0	0
м	97	278	73	143	62	0	0	0
й	18	283	128	130	25	0	0	0
А	15	220	108	107	5	0	0	0
М	13	322	211	110	1	0	0	0

Два варианта $P_1(Cl)$ и $P_2(Cl)$ укладки элементов кластера Cl с кортежами $p^1(Cl) = \{p_0^1, p_1^1, p_2^1 \cdots\}$ и $p^2(Cl) = \{p_0^2, p_1^2, p_2^2 \cdots\}$ сравниваются следующим образом. При выполнении одного из условий:

- $p_0^1 > p_0^2$
- $p_0^1 = p_0^2$ и при этом $p_1^1 < p_1^2$ (1)
- $p_0^1 = p_0^2$ и $p_i^1 = p_i^2$ при $0 < i < q$ и при этом $p_q^1 < p_q^2$.

считаем, что $P_1(Cl) > P_2(Cl)$.

Статистика распределения точек по слоям для других кластеров приведена в табл. 1, которая иллюстрирует тенденцию скопления большинства точек кластера в множестве $A_C \cup L_1 \cup L_2$. Например, для кластера «а» доля точек, не попавших в множество $A_C \cup L_1 \cup L_2$, составляет менее 7,5 %, а для оставшихся кластеров, перечисленных в табл. 1, она равна 0 или исчезающее мала.

В табл. 1 приведена статистика для кластеров, суммирование элементов которых производилось наложением на опорный элемент, являющийся первым из возможных кандидатов с наилучшей оценкой и словарным подтверждением. Вообще говоря, указанный способ суммирования не гарантирует оптимальности укладки с точки зрения условия (1). В табл. 2 для рас-

Таблица 2

Распределение по слоям при различных вариантах укладки

Номер опорного элемента	S_v	$\|A_C\|$	$\|L_1\|$	$\|L_2\|$	$\|L_3\|$	$\|L_4\|$	$\|L_5\|$
0	170	56	99	15	0	0	0
6	179	52	111	16	0	0	0
8	169	54	107	8	0	0	0
11	172	52	102	18	0	0	0
14	182	55	107	17	0	0	0
24	183	49	96	34	4	0	0
31	184	60	105	19	0	0	0
53	178	49	98	24	4	2	1
61	170	49	98	16	4	2	1

смотренного кластера, состоящего из 72 элементов «я», приведено несколько вариантов укладки, построенных для различных опорных элементов.

Очевидно, что варианты укладки, представленные в табл. 2, различны и могут быть упорядочены с помощью критерия сравнения (1).

Рассмотрим теперь другую модель суммы растров элементов кластера:

$$P^*(Cl, L_E, L_S) = A_C\,(L_S) \cup l_1 \cup l_2 \cup \ldots, \qquad (2)$$

где A_C — *общая область*, совпадающая с пороговым образом $Tr(L_S)$, а слои l_1, l_2, … содержат точки порогового образа $Tr(L_E)$, находящиеся на одинаковом расстоянии от $Tr(L_S)$. Порог L_E призван избавиться от искажений индивидуальных образов символов в $P(Cl)$, а порог L_S позволяет расширить общую область. Предполагается, что $k_E = L_E/\mu(Cl) < 1/2$, а $k_S = L_S/\mu(Cl) > 1/2$.

Пересчитаем варианты укладки из табл. 2 в виде представления (2) для $k_E = 0{,}2$, $k_S = 0{,}84$. Результаты, сведенные в табл. 3, показывают существенно меньшую зависимость вариантов укладки от выбора опорного элемента по отношению к данным табл. 2.

Аналогичные экспериментальные результаты, стабильное разбиение суммы растров на общую область и два слоя, получаются для всех кластеров большого объема. При этом пороги $k_E = 0{,}2$, $k_S = 0{,}84$ могут варьироваться, а выбор опорного элемента не оказывает существенного влияния.

Приведем рассчитанные слои и общие области для кластера сверток, описанного в разделе 1 и являющегося результатом численного эксперимента.

Таблица 4 иллюстрирует представление образа кластера в виде (2) и сходство структуры (2) образов кластеров (смоделированного и реального).

Таблица 3

Распределение по слоям при различных вариантах укладки для представления (2)

Номер опорного элемента	**$Tr(L_E)$**	**$\lvert A_C (L_S)\rvert$**	**$\lvert l_1\rvert$**	**$\lvert l_2\rvert$**	**$\lvert l_3\rvert$**	**$\lvert l_4\rvert$**	**$\lvert l_5\rvert$**
0	149	94	55	0	0	0	0
6	148	95	53	0	0	0	0
8	147	95	52	0	0	0	0
11	148	91	57	0	0	0	0
14	145	92	53	0	0	0	0
24	142	93	49	0	0	0	0
31	146	94	52	0	0	0	0
53	145	94	51	0	0	0	0
61	145	95	50	0	0	0	0

Таблица 4

Распределение по слоям при различных вариантах укладки для порогов $k_E = 0{,}2$, $k_S = 0{,}84$ для примера из раздела 1

Номер опорного элемента	**$Tr(L_E)$**	**$\lvert A_C (L_S)\rvert$**	**$\lvert l_1\rvert$**	**$\lvert l_2\rvert$**	**$\lvert l_3\rvert$**	**$\lvert l_4\rvert$**	**$\lvert l_5\rvert$**
0	326	252	74	0	0	0	0
1	324	249	75	0	0	0	0
2	322	249	73	0	0	0	0
3	322	249	73	0	0	0	0
4	349	250	99	0	0	0	0
5	332	243	89	0	0	0	0
6	327	218	108	1	0	0	0
7	328	252	76	0	0	0	0
8	326	252	74	0	0	0	0

3. Формирование эталонов

Таким образом, суммирование образов элементов кластера и введение двух порогов позволяет представить образ кластера в виде трех областей:

- $A_C(L_S)$ — общая область, содержащая точки образа кластера, значения которых превосходят L_S и которые должны принадлежать образам большинства элементов;
- l_1 — область, содержащая точки образа кластера, значения которых превосходят порог L_E и расстояние от каждой из которых до общей области составляет 1 (по-прежнему используем расстояние Хаусдорфа), эта область образуется благодаря эффектам оцифровки;
- l_2 — область, содержащая точки образа кластера, значения которых превосходят порог L_E и расстояние от каждой из которых до общей области составляет 2, эта область образуется из-за случайных искажений границ образов.

Для совокупности образов, представленных на рис. 4 и 5, общая область будет иметь ширину 2, будет образован слой l_1, а слой l_2 будет формироваться в соответствии с закономерностями, описанными в [3] и связанными с дефокусировкой.

Описанная процедура, состоящая в разбиении суммы образов на слои (2), позволяет извлечь из образа кластера $P(Cl)$ два бинарных образа:

- *общий образ* $GEN(Cl) = \|g_{ij}\|$, определенный границами $A_C(L_S)$, т. е.

$$g_{ij} = 1 \text{ при } p_{ij} \geq L_S, \text{ в противном случае } g_{ij} = 0;$$

- *расширенный образ* $COVER(Cl) = \|c_{ij}\|$, определенный границами l_2, т. е.

$$c_{ij} = 1 \text{ при } d_H(c_{ij}, A_C) \leq 2, \text{ в противном случае } c_{ij} = 0.$$

Отметим, что в расширенный образ могут входить точки, не принадлежащие образу кластера. Очевидно, что $GEN \subset COVER$.

Два образа *GEN* и *COVER* составляют *эталон*, соответствующий коду символа кластера, который будет использоваться в последующем для распознавания образов.

4. Распознавание образов символов с помощью эталонов

Распознавания символа с набором эталонов производится посредством сравнения с каждым из эталонов с помощью следующего алгоритма.

Символ помещается внутрь большого объемлющего растра размером N на M, центрируется и сравнивается с эталонами, извлеченными из кластеров. При вычислении расстояния между образом кластера и распозна-

ваемым образом штрафуются точки растра, не попавшие в расширенный образ кластера или в общий образ кластера.

Таким способом уменьшается оценка образа с отсутствующими в общем образе кластера точками, т. е. образа, в котором отсутствуют точки, принадлежащие почти всем элементам кластера. Также уменьшается оценка образа, обладающего точками, не попавшими в расширенный образ кластера, т. е. образа, которому принадлежат точки, отсутствующие в большинстве элементов кластера.

Пусть $R = \|r_{ij}\|$ — распознаваемый образ, $GEN(Cl) = \|g_{ij}\|$ — общий образ кластера Cl, $COVER(Cl) = \|c_{ij}\|$ — расширенный образ кластера Cl, $i \in \{1, \ldots, N\}, j \in \{1, \ldots, M\}, r_{ij} \in \{0, 1\}, s_{ij} \in \{0, 1\}, c_{ij} \in \{0, 1\}$.

Определим три множества:

$$C_0 = \{s_{ij} \mid s_{ij} > 0, d_H(s_{ij}, R) = 1, i \in \{1, \ldots, N\}, j \in \{1, \ldots, M\}\},$$
$$C_1 = \{r_{ij} \mid r_{ij} > 0, d_H(r_{ij}, COVER(Cl)) = 1, i \in \{1, \ldots, N\}, j \in \{1, \ldots, M\}\},$$
$$C_2 = \{r_{ij} \mid r_{ij} > 0, d_H(r_{ij}, COVER(Cl)) > 1, i \in \{1, \ldots, N\}, j \in \{1, \ldots, M\}\}.$$

Расстояние между растром R и кластером Cl вычисляем как

$$Dist(R, Cl) = |C_0| + |C_1| + 2|C_2|. \quad (3)$$

Оценка сходства рассматриваемого растра R и кластером Cl вычисляется следующим образом:

$$Conf(R, Cl) = \max(0, 255 - Dist(R, Cl)).$$

Такое вычисление проводится для нескольких взаимных положений растра и образов кластера — помимо центрированного положения исследуются также сдвиги образа $R = \|r_{ij}\|$ на единицу в разных направлениях, т. е. растры

$$R_h^{(\pm 1)} = \|r_{i\pm 1, j}\|,$$
$$R_v^{(\pm 1)} = \|r_{i, j\pm 1}\|,$$
$$R_{hv}^{(\pm 1)} = \|r_{i\pm 1, j\pm 1}\|.$$

В качестве степени сходства берется максимальная величина из полученных значений:

$$Conf_{\max}(R, Cl) = \max(Conf(R, Cl), Conf(R_h^{(\pm 1)}, Cl), Conf(R_v^{(\pm 1)}, Cl), Conf(R_{hv}^{(\pm 1)}, Cl)).$$

Результатом сравнения растра R с кластером Cl является код кластера и оценка $Conf_{\max}(R, Cl)$.

В результате сравнения растра R со всеми созданными эталонами возникает коллекция возможных альтернатив из имен тех эталонов, степень сходства с которыми больше нуля. Возможны случаи, когда эта коллекция может быть пустым множеством.

Пусть по-прежнему каждый из порогов L_E и L_S выбирается в виде $k\mu(Cl)$. Рассмотрим эксперимент, позволяющий выбрать оптимальные зна-

Таблица 5

Точность распознавания алгоритмов для различных порогов kE и kS

k_E \ k_S	**0,6**	**0,64**	**0,68**	**0,72**	**0,76**	**0,80**	**0,84**	**0,88**	**0,92**
0,09	98,966	99,15	99,219	99,564	99,633	99,679	99,656	99,702	99,541
0,12	99,127	99,265	99,38	99,610	99,679	99,725	99,702	99,702	99,633
0,15	99,127	99,38	99,449	99,587	99,656	99,725	99,702	99,679	99,610
0,18	99,127	99,38	99,449	99,587	99,656	99,725	99,702	99,679	99,610
0,21	99,449	99,541	99,541	99,610	99,702	99,748	99,725	99,725	99,633
0,24	99,449	99,541	99,541	99,610	99,702	99,748	99,725	99,748	99,633
0,27	99,495	99,610	99,61	99,633	99,702	99,725	99,725	99,748	99,633
0,3	99,518	99,633	99,656	99,679	99,748	99,771	99,771	99,771	99,656
0,33	99,518	99,633	99,656	99,679	99,748	99,771	99,771	99,771	99,656
0,36	99,495	99,610	99,633	99,656	99,725	99,725	99,725	99,725	99,610
0,39	99,472	99,587	99,633	99,633	99,702	99,702	99,702	99,748	99,610
0,42	99,472	99,587	99,633	99,633	99,702	99,702	99,702	99,725	99,610
0,45	99,495	99,587	99,656	99,656	99,702	99,702	99,702	99,725	99,610
0,48	99,495	99,587	99,656	99,656	99,702	99,702	99,702	99,725	99,633
0,51	99,495	99,587	99,656	99,656	99,702	99,702	99,702	99,725	99,633
0,54	99,449	99,541	99,633	99,610	99,679	99,702	99,702	99,725	99,633
0,57	99,449	99,541	99,633	99,633	99,702	99,725	99,725	99,748	99,610
0,60	99,403	99,495	99,541	99,610	99,679	99,702	99,702	99,702	99,518
0,63	99,403	99,564	99,564	99,610	99,679	99,702	99,702	99,702	99,518
0,66	99,403	99,564	99,564	99,610	99,679	99,679	99,679	99,702	99,518
0,69	99,426	99,564	99,564	99,610	99,679	99,679	99,679	99,702	99,518
0,72	99,449	99,587	99,587	99,633	99,679	99,679	99,679	99,725	99,541

чения порогов k_E и k_S, на некоторой тестовой последовательности образов. В диапазонах $k_E \in [0{,}0;\ 0{,}3]$ и $k_S \in [0{,}6;\ 1{,}0]$ вычислим точность описанного распознавания метода $\Re_c$.

Результаты эксперимента, приведенные в табл. 5, показывают стабильность точности распознавания при $k_E \in [0{,}18;\ 0{,}36]$ и $k_S \in [0{,}80;\ 0{,}88]$, рабочим приближением являются значения порогов $k_E \approx 0{,}21$ и $k_S \approx 0{,}84$. Отметим, что в диапазонах $k_E \in [0{,}18;\ 0{,}36]$ и $k_S \in [0{,}80;\ 0{,}88]$ оценка монотонности $M_{255} \approx 0$.

Разработанный алгоритм $\mathfrak{R}_c$ распознавания образов символов сравнением с эталонами, построенными на этапе кластеризации, является основой для повторного распознавания отдельных символов и повторной сегментации на втором проходе. Метод $\mathfrak{R}_c$ сравнения с эталонами способен сравнивать образы символов более точно, нежели шрифтонезависимые методы [7].

5. Комбинирование алгоритмов распознавания с методом $\mathfrak{R}_c$

В результате работы первого прохода распознавания у всех символов (как бывших отдельными компонентами, так и получившихся в результате сегментации) возникают альтернативы распознавания.

Задачей второго прохода распознавания является уточнение результатов первого прохода, подтверждение результатов распознавания либо его изменение, возможная иная сегментация некоторых слов.

В том случае, если оценка распознавания на первом проходе высока, символ подтвержден словарно (т. е. входит в состав достаточно длинного слова, присутствующего в словаре), дополнительной проверки этого символа не производится. В других случаях символ распознается с использованием построенных эталонов. Если результаты распознавания на первом и втором проходах совпадают, символ считается подтвержденным. Если же результаты отличаются, возникает вопрос — результатам какого прохода верить больше, как формировать альтернативы по результатам двух проходов?

Комбинирование основывается на распределениях ошибок двух алгоритмов: используемого на первом проходе шрифтонезависимого алгоритма $\mathfrak{R}_1$ [4] и алгоритма $\mathfrak{R}_c$ сравнения с эталонами, полученными в результате кластеризации. Каждое из полученных экспериментально распределений задано значениями вероятности ошибок $P_{err}(\mathfrak{R}, w)$ для возможных значений оценок $w \in [0, 255]$ метода $\mathfrak{R}$. Для замены альтернатив алгоритма $\mathfrak{R}_1$ альтернативами алгоритма $\mathfrak{R}_c$ необходимо выполнение следующего условия:

$$P_{err}(\mathfrak{R}_c, w_1) < P_{err}(\mathfrak{R}_1, w).$$

Пусть метод $\mathfrak{R}_1$ дал альтернативы распознавания символа $r^1 = \{(C^1_1, w^1_1), \ldots, (C^1_n, w^1_n)\}$, а метод $\mathfrak{R}_c$ — $r^c = \{(C_1, w_1), \ldots, (C_m, w_m)\}$.

Алгоритм комбинирования предусматривает следующие случаи:

- код символа C^1_1 отсутствует среди кодов $C_1, \ldots, C_m$, в частности, если коллекция пуста, $n = 0$. Кластерный результат берется в качестве основного, если $P_{err}(\mathfrak{R}_c, w_1) < P_{err}(\mathfrak{R}_1, w^1_1)$;

Таблица 6

Точность распознавания алгоритмов $\mathfrak{R}_1$ и $\mathfrak{R}_{1c}$ для различных стендов

Стенд обучения / Стенд распознавания	LS_1		LS_2		LS_3	
TS_1			$\mathfrak{R}_1$	99,25	$\mathfrak{R}_1$	99,25
			$\mathfrak{R}_{1c}$	99,77	$\mathfrak{R}_{1c}$	99,77
TS_2	$\mathfrak{R}_1$	99,25			$\mathfrak{R}_1$	99,25
	$\mathfrak{R}_{1c}$	99,63			$\mathfrak{R}_{1c}$	99,60
TS_3	$\mathfrak{R}_1$	99,59	$\mathfrak{R}_1$	99,59		
	$\mathfrak{R}_{1c}$	99,78	$\mathfrak{R}_{1c}$	99,61		

- код символа C^1_1 присутствует среди кодов C_1, …, C_m, т. е. $\exists\ C_i$: $C_i = C^1_1$, и при этом для кода C^1_1 не существует близких по начертанию символов. Если $w_1 \geq W_4$ и $P_{err}(\mathfrak{R}_c, w_i) \leq P_{err}(\mathfrak{R}_1, w^1_1)$, то в качестве результата берется коллекция r^c. Если $P_{err}(\mathfrak{R}_c, w_1) > P_{err}(\mathfrak{R}_1, w^1_1)$ и $w_1 \geq W_4$, то в качестве результата берется коллекция r^1. В оставшихся случаях коллекции r^1 и r^c объединяются;
- для близких по начертанию символов (C^1_1, w^1_1) и (C^n_j, w^n_j), для которых $\exists\ C_{i1}, C_{i2}$: $C_{i1} = C^1_1$, $C_{i2} = C^n_j$, в качестве результата берется коллекция r^1, если $|w_{i1} - w_{i2}| < \delta$, в противном случае выбирается код C_{i1} или C_{i2} с наибольшей оценкой;
- ряд близких по начертанию символов различается с помощью функций расстояния, основанных на исследовании специфичных областей в образах сходных по начертанию символов [6].

Целью описанного комбинированного алгоритма $\mathfrak{R}_{1c}$, основанного на алгоритмах $\mathfrak{R}_1$ и $\mathfrak{R}_c$, является повышение точности распознавания алгоритма $\mathfrak{R}_1$. Эксперименты, результаты которых приведены в табл. 6, показывают стабильность повышения качества распознавания при помощи комбинированного алгоритма $\mathfrak{R}_{1c}$ по отношению к качеству шрифтонезависимого алгоритма $\mathfrak{R}_1$. Точность в табл. 5 оценивалась как доля ошибочно распознанных символов по отношению к общему числу символов, при этом рассматривались только ошибки в символах с правильно определенными границами. На различных последовательностях (стендах) LS_1, LS_2, LS_3 определялись распределения $P_{err}(\mathfrak{R}_c, w)$ и $P_{err}(\mathfrak{R}_1, w)$, после чего точность оценивалась на последовательностях, отличных от обучающей.

Таблица 7

Монотонность оценок алгоритмов $\mathfrak{R}_1$ и $\mathfrak{R}_{2c}$ для различных стендов

Стенд \ Алгоритм	$\mathfrak{R}_1$		$\mathfrak{R}_{2c}$	
	M_{240}, %	M_{255}, %	M_{240}, %	M_{255}, %
TS_3	0,748	0,313	0,250	0,0
TS_6	0,780	0,161	0,370	0,0
TS_9	0,373	0,000	0,373	0,0

Другой способ комбинирования ориентирован не на повышение точности распознавания, а на повышение монотонности оценок распознавания алгоритма с высокой точностью.

Алгоритм комбинирования $\mathfrak{R}_{2c}$ состоит в замене оценки первой альтернативы коллекции, сформированной алгоритмом $\mathfrak{R}_1$, на оценку алгоритма кластерного наложения $\mathfrak{R}_c$. Оценим характеристики монотонности на различных последовательностях с помощью алгоритмов $\mathfrak{R}_1$ и $\mathfrak{R}_{2c}$. *Монотонность* в табл. 7 оценивалась как доля образов из тестовой последовательности, которые распознанные неверно с ошибкой, не ниже заданного уровня; рассматривались две оценки монотонности M_{240} и M_{255}, соответствующим уровням ошибок 240 и 255. При этом рассматривались только ошибки в символах с правильно определенными границами.

Экспериментально подсчитанная на стендах TS_1–TS_9 *скорость распознавания* алгоритма $\mathfrak{R}_1$ составляет 8000–12 000 символов в секунду (во всех экспериментах в настоящей работе использовался одноядерный процессор Intel с частотой FSB = 3 ГГц.), в то время как быстродействие алгоритма $\mathfrak{R}_{1c}$ варьируется в широком диапазоне от 1600 до 4600 символов в секунду. Последнее объясняется различным количеством эталонов, построенных для конкретных страниц. Значения *относительной скорости распознавания*, т. е. количество сравнений за одну секунду одного образа символа и одного эталона по формуле (3), варьируются в диапазоне от 180 000 до 220 000 сравнений в секунду.

Алгоритм, комбинирующий $\mathfrak{R}_{1c}$ и $\mathfrak{R}_{2c}$ и состоящий в первоначальной экспертизе оценок первого этапа распознавания быстрым алгоритмом $\mathfrak{R}_{2c}$ и, в случае низкой оценки, в применении медленного алгоритма $\mathfrak{R}$1c. Скорость распознавания алгоритма $\mathfrak{R}_{3c}$ составляет 10 000-15 000 символов в секунду.

Другим средством повышения быстродействия алгоритма $\mathfrak{R}_{1c}$ является знание набора шрифтов, применяемое для уменьшения количества эталонов при сравнении [1].

6. Использованный инструментарий

В работе был использован инструментарий, описанный в табл. 8.

Программное обеспечение из пунктов 1, 2 описано в работе [5], программное обеспечение пункта 5 реализовано Н. В. Котовичем. Разработка остальных компонент и модификация программного обеспечения пунктов 1, 2, 5 были произведены автором.

Выводы

Предложенный в статье способ извлечения эталонов, состоящий в разбиении образа кластера на слои и формировании двух бинарных растров, с которыми происходит в последующем сравнение неизвестного образа, хорошо зарекомендовал себя в составе комплекса Cuneiform [5].

Таблица 8

Использованный инструментарий

№	Наименование компонент	Описание	Назначение
1	APuma.dll и вызываемые модули	Ядро системы распознавания документов Cuneiform	Первичное распознавание; сохранение последовательностей образов в контейнере изображений
2	DPuma.dll	Пользовательский отладчик	Визуализация процессов распознавания символов; визуализация наложения образов и кластеров
3	CTB.dll	Контейнер изображений	Хранение бинарных и полутоновых изображений; сохранение и обновление записей
4	CTB_Man.exe	Редактор контейнера изображений	Просмотр и редактирование изображений; операции пользователя по суммированию кластеров
5	Fon32.dll	Библиотека кластеризации	Кластер анализ изображений; извлечение эталонов; распознавания с помощью эталонов
6	CTB_Test.exe	Потоковое обучение и распознавание	Обучение метода распознавания, реализованного в виде библиотеки-плагина, на обучающей последовательности из контейнера изображений; распознавание с помощью библиотеки-плагина тестовой последовательности из контейнера изображений

Однако возможны модификации как формулы (3), так и представления общего и расширенного образов кластера. Например, описанная в работе [6] модификация позволяет улучшать точность распознавания образов, отсканированных при малых разрешениях (150 dpi).

Другие возможные модификации способа вычисления расстояния между бинарным образом и кластером могут понадобиться при оптимизации точности распознавания или иных характеристик качества распознавания, например монотонности оценок надежности.

Литература

1. *Славин О. А.* Алгоритмы распознавания шрифтов в печатных документах // Информационные технологии и вычислительные системы. 2010. № 3. С. 3–13.
2. *Котович Н. В.* Алгоритмы кластеризации образов символов // В сб. трудов ИСА РАН «Обработка изображений и анализ данных». М.: Книжный дом «Либроком»/URSS, 2008. Т. 38. С. 241–251.
3. *Титов Ю. В.* Об искажении символов при сканировании // В сб. трудов ИСА РАН «Системный подход к управлению информацией». М.: КомКнига/URSS, 2006. Т. 23. С. 260–288.
4. *Славин О. А.* Комбинированные методы распознавания печатных и рукопечатных символов // В сб. трудов ИСА РАН «Документооборот. Концепции и инструментарий». М.: URSS, 2004. С. 151–172.
5. *Кочин Д. Ю., Хлебутин П. С.* Разработка многомодульных программных комплексов // В сб. трудов ИСА РАН «Развитие безбумажной технологии в организационных системах». М.: URSS, 1999.
6. *Славин О. А., Титов Ю. В.* Динамическое построение функций сравнения с идеальным образом в задаче адаптивного распознавания текстовых символов // Информационные технологии и вычислительные системы. 2007. № 1. С. 3–12.
7. *Арлазаров В. Л., Котович Н. В., Славин О. А.* Адаптивное распознавание // Информационные технологии и вычислительные системы. 2002. № 4. С. 11–22.