Настройка алгоритму розпізнавання тексту

Савчинський Б.Д., Камоцький О.В.

24 вересня 2004 р.

Вступ

Задача розпізнавання тексту є, напевно, найстарішою і найпопулярнішою серед задач розпізнавання документів. Традиційно, починаючи з робіт Ковалевського [1, 2], задача розпізнавання рядка тексту формулюється як задача пошуку найкращої послідовності еталонів літер і розв'язується за допомогою методів динамічного програмування.

Однак, сама лише ця постановка задачі не дає відповіді на запитання, яким чином будувати еталони літер. Традиційно [3, 4, 5] вважається, що зображення тексту спотворюється випадковим шумом певного типу. Цим самим задається статистична модель генерування зображення літери за її еталоном. На основі отриманої моделі та прикладів зображень літер їх еталони будуються як оцінки максимальної правдоподібності.

Оскільки ж сама задача побудови статистичної моделі зображень літер є доволі складною, то зазвичай ця модель фіксується вольовим чином, виходячи з певних апріорних міркувань.

В нашій роботі ми пропонуємо інший підхід до задачі побудови еталонів, який дозволяє уникнути оцінювання їх статистичних моделей. Сформульовані і розв'язані нами задачі є розвитком задач настройки параметрів автономного стохастичного автомата, вперше розглянутих у [6]. Для позначення цього підходу, так само, як це зроблено у [6], ми вживатимемо слово «настройка» на відміну від задач статистичного оцінювання, для яких прийнятим є термін «навчання».

Стаття складається з шести розділів. В першому розділі ми формулюємо основні означення та постановку задачі розпізнавання рядка тексту. У другому розділі наведено постановку задачі побудови еталонів літер як задачі настройки. Третій та четвертий розділи присвячені її розв'язку. П'ятий розділ містить результати експериментів. Висновки до статті сформульовані у заключному розділі.

1 Основні означення. Постановка задачі розпізнавання

Розглянемо задачу розпізнавання одного рядка тексту за його зображенням.

Назвемо *полем зору* T прямокутну підмножину двовимірної цілочисельної решітки. Її висоту та ширину позначатимемо відповідно H та W:

$$T = \{(i,j) \mid i \in \overline{1,H}, j \in \overline{1,W}\}.$$

Елементами поля зору T є координати пікселів зображення.

Нехай V – певна множина сигналів. *Зображенням* x називатимемо функцію вигляду $x : T \to V$. Яскравість піксела з координатами t на зображенні x позначатимемо $x(t)$ чи x_t. Множину усіх можливих зображень позначимо V^T.

Нехай A – скінчена множина, що складається з імен літер, а $\alpha \in A$ – це ім'я якоїсь літери. *Рядок тексту* розумітимемо як послідовність імен α.

Вважатимемо, що зображення усіх літер на полі зору мають однакову висоту H і ширину $d(\alpha)$, яка залежить лише від імені літери α. Таким чином, вважатимемо визначеною *функцію ширини літери* вигляду $d : A \to \mathbb{N}$.

Сегментом s називатимемо пару (α, j), де α – ім'я літери з A, а $j \in (1, 2, \dots, W)$. Кожен сегмент $s = (\alpha, j)$ визначає фрагмент $T(s)$ поля зору із іменем $\alpha(s)$, який має висоту H, ширину $d(\alpha(s))$ і починається зі стовпчика номер $j(s)$:

$$T(s) = \left\{ (i, j) \mid i \in \overline{1, H}, j \in \overline{j(s), j(s) + d(\alpha(s)) - 1} \right\}.$$

Таким чином, сегменти — це поіменовані фрагменти поля зору. Множину усіх сегментів позначатимемо S.

Усілякому текстовому рядку $(\alpha_1, \dots, \alpha_L)$ довжини L, зображеному на x, однозначно відповідає послідовність сегментів $(s_1, \dots, s_L)$, які щільно прилягають один до одного і разом покривають усе поле зору T:

$$\left\{ \begin{array}{rcll} \alpha(s_l) & = & \alpha_l & \forall l \in \overline{1, L} \\ j(s_1) & = & 1 & \\ j(s_{l+1}) & = & j(s_l) + d(\alpha(s_l)) & \forall l \in \overline{1, L-1} \\ j(s_L) + d(\alpha(s_L)) & = & W + 1 & \end{array} \right. \tag{1}$$

Послідовність $\overline{s} = (s_1, \dots, s_L)$, що задовільняє умови (1), назвемо *сегментацією.* Її довжину позначатимемо $L(\overline{s})$. Множину усіх можливих сегментацій на полі зору позначатимемо $\overline{S}$.

Введемо множину $E \subset \mathbb{R}^K$ векторів параметрів. Кожен вектор $\overline{e} \in E$ визначає еталонний вигляд зображень усіх літер $\alpha \in A$. Для визначення *схожості* зображення x у місці, визначеному сегментом s, на літеру $\alpha(s)$, введемо *локальну функцію відмінності* $f : E \times S \times V^T \to \mathbb{R}$. Чим більша схожість, тим меншим має бути її значення, яке позначатимемо $f_{\overline{e}}(s, x)$.

Міру *схожості зображення* x та рядка $(\alpha_1, \dots, \alpha_L)$, якому відповідає сегментація $\overline{s} = (s_1, \dots, s_L)$, ми визначатимемо як сумарну схожість кожного сегмента $\overline{s}$, тобто як $\sum_{l=1}^{L(\overline{s})} f_{\overline{e}}(s_l, x)$.

Задача 1 *розпізнавання зображення x текстового рядка полягає у знаходженні сегментації $\overline{s}^* = (s_1^*, s_2^*, \dots, s_L^*)$, на яку воно найбільш схоже:*

$$\overline{s}^* = \arg\min_{\overline{s} \in \overline{S}} \sum_{l=1}^{L(\overline{s})} f_{\overline{e}}(s_l, x) \tag{2}$$

Строго кажучи, функція $\sum_{l=1}^{L(\overline{s})} f_{\overline{e}}(s_l, x)$ може приймати мінімальне значення на деякій множині сегментацій $\overline{S}^*$. В такому разі вважатимемо, що $\overline{s}^*$ позначає будь-який елемент цієї множини. Ситуацію, коли розв'язок (2) обов'язково має бути єдиним, ми позначатимемо значком рівності $\stackrel{!}{=}$.

Задача 1 розв'язується методом динамічного програмування, що було вперше зроблено у роботі Ковалевського [1]. Формальний підхід до розпізнавання зображень, які розглядаються як послідовності простіших, менших своїх частин, дістав назву «метод еталонних послідовностей» [2].

Надалі вважатимемо, що в нас є готова процедура — *алгоритм розпізнавання* $\overline{s}^*_P(x, \overline{e})$ для знаходження сегментації $\overline{s}^*$ шляхом розв'язання задачі 1 для заданої трійки $P = \langle A, d, f_{\overline{e}} \rangle$ — множини імен літер A, функції ширини літери d та локальної функції відмінності $f_{\overline{e}}$, визначеної с точністю до вектора параметрів $\overline{e}$. Реалізація $\overline{s}^*_P(x, \overline{e})$ нас не цікавитиме, однак надалі ми розраховуватимемо, що вона дозволяє відстежувати випадки, коли розв'язок $\overline{s}^*$ не є єдиним.

На практиці P фіксується, виходячи з припущень про природу вхідного зображення. Наприклад, у машинописному тексті літери мають фіксовану ширину, множина їх імен A визначається наявними друкованими символами, а вектор параметрів $\overline{e}$ може містити еталонні, чітко надруковані зображення літер.

Зазвичай модель вхідного зображення передбачає, що на ідеальне зображення $\chi(\alpha)$ літери α діє випадковий адитивний шум r із сферично-симетричною функцією розподілу $p_{\text{ш}}(r)$:

$$x = \chi(\alpha) + r. \tag{3}$$

Тут x – зашумлене зображення літери. В такому разі можна застосувати кореляційний метод розпізнавання [3, 2], а за локальну функцію відмінності $f_{\overline{e}}$ взяти функцію правдоподібності. За припущення, що щільність імовірності $p_{\text{ш}}(r)$ шуму є деякою монотонно спадаючою функцією $f(\cdot)$ від суми квадратів його компонент

$$p_{\text{ш}}(r) = f\left(r^2\right), \tag{4}$$

функція $f_{\overline{e}}$ набуває вигляду евклідової відстані між векторами $x = \{x_{i,j} \mid (i,j) \in T(s)\}$ та $\overline{e} = \overline{\chi}(\alpha) = \left\{\chi^{\alpha}_{i,j} \mid (i,j) \in T(s)\right\}$:

$$f_{\overline{e}}(s, x) = \sum_{\substack{i \in \overline{1,H} \\ j = \overline{1, d(\alpha(s))}}} \left(\chi^{\alpha(s)}_{i,j} - x_{i, j(s)+j}\right)^2. \tag{5}$$

2 Вибір вектора параметрів. Задача настройки

Настройка алгоритму розпізнавання $\overline{s}^*_P(x, \overline{e})$, який розв'язує задачу 1, полягає у виборі значення вектора параметрів $\overline{e}$ для заданої трійки $P = \langle A, d, f_{\overline{e}} \rangle$.

Нехай задано множину тестових зображень $X^o = \{x^o_r \mid r \in \overline{1,R}\}$ та множину відповідних їм сегментацій $\overline{S}^o = \{\overline{s}^o_r \mid r \in \overline{1,R}\}$.

Задача 2 *настройки алгоритму розпізнавання полягає у знахоженні такого вектора параметрів $\overline{e}^*$, щоб результатом розпізнавання кожного зображення-зразка x^o_r була відповідна тестова сегментація $\overline{s}^o_r$ і тільки вона одна:*

$$\overline{s}^o_r \stackrel{!}{=} \arg\min_{\overline{s} \in \overline{S}} \sum_{l=1}^{L(\overline{s})} f_{\overline{e}^*}(s_l, x^o_r) \qquad \forall r \in \overline{1,R}.$$

Складність задачі полягає у тому, що мінімум шукається на множині сегментацій $\overline{S}$, потужність якої пропорційна кількості *усіх можливих текстів*, що можуть бути зображені на полі зору.

3 Розв'язання задачі настройки

Ми пропонуємо розв'язання задачі настройки 2 у випадку, коли функція відмінності є лінійною за параметрами $\overline{e} = \{e_1, \ldots, e_K\}$, тобто може бути представлена у вигляді:

$$f_{\overline{e}}(s, x) = \sum_{k=1}^{K} e_k \cdot \varphi_k(s, x) = \langle \overline{e}, \overline{\varphi}(s, x) \rangle . \tag{6}$$

Тут $\overline{\varphi}(s, x) = \{\varphi_1(s, x), \ldots, \varphi_K(s, x)\}$ – деякий вектор, який безпосередньо залежить від зображення x та сегмента s.

Зазначимо, що вигляд функції відмінності (6) охоплює широко вживаний випадок (5), коли компоненти $\overline{e}$ приймають значення із множини сигналів пікселів зображення V, а сам вектор параметрів безпосередньо містить ідеальні зображення усіх літер $\alpha \in A$. Однак, у загальному випадку, вектор $\overline{e}$ може мати більш складний і не візуальний зміст.

Позначимо множину пар тестових зображень та сегментацій, що їм відповідають, як Π:

$$\Pi = \left\{ (x_r^o, \overline{s}_r^o) \mid r \in \overline{1, R} \right\} .$$

Тепер задача настройки 2 набуває такого вигляду:

знайти $\overline{e}^*$ такий, щоб:

$$\overline{s}^o \stackrel{!}{=} \arg\min_{\overline{s} \in \overline{S}} \sum_{l=1}^{L(\overline{s})} \sum_{k=1}^{K} e_k^* \varphi_k(s_l, x^o) \qquad \forall\, (x^o, \overline{s}^o) \in \Pi. \tag{7}$$

Рівність (7) означає, що для кожної пари $(x^o, \overline{s}^o) \in \Pi$ має виконуватися система нерівностей:

$$\left\{ \sum_{l=1}^{L(\overline{s})} \sum_{k=1}^{K} e_k^* \varphi_k(s_l^o, x^o) < \sum_{l=1}^{L(\overline{s})} \sum_{k=1}^{K} e_k^* \varphi_k(s_l, x^o) \qquad \forall \overline{s} \neq \overline{s}^o, \right.$$

що еквівалентно

$$\left\{ \sum_{l=1}^{L(\overline{s})} \sum_{k=1}^{K} e_k^* \varphi_k(s_l, x^o) - \sum_{l=1}^{L(\overline{s})} \sum_{k=1}^{K} e_k^* \varphi_k(s_l^o, x^o) > 0 \qquad \forall \overline{s} \neq \overline{s}^o. \right. \tag{8}$$

Перепишемо (8), згрупувавши компоненти вектора параметрів $\overline{e}^*$ і позначивши усі члени при e_k^* як $\varphi'_k(\overline{s}, \overline{s}^o, x^o)$:

$$\left\{ \sum_{k=1}^{K} e_k^* \varphi'_k(\overline{s}, \overline{s}^o, x^o) > 0 \qquad \forall \overline{s} \neq \overline{s}^o. \right. \tag{9}$$

Таким чином, задача настройки 2 полягає у розв'язанні (9) відносно вектора $\overline{e}^*$.

Позаяк лінійна система (9) містить $\left|\overline{S}\right| - 1$ нерівностей, для її розв'язання неможливо використати класичні методи лінійної оптимізації, оскільки їх складність безпосередньо залежить від кількості нерівностей. Однак, розв'язок (9) можна знайти за допомогою методів розмежування множин точок гіперплощиною, складність яких не залежить від потужності системи, а залежить від інших її властивостей.

В нашій задачі розпізнавання рядка тексту ми використовували алгоритми Козинця та персептрона. Їх особливістю є те, що пошук розв'язку системи (9) відбувається шляхом ітеративного корегування значення вектору параметрів на основі будь-якої однієї нерівності, що не виконується. Тому для своєї роботи вони потребують лише ефективного методу пошуку такої нерівності. Збіжність цих алгоритмів за скінчену кількість кроків визначають відповідні теореми, описані у [6].

У нашому випадку, невиконання будь-якої нерівності у системі (9), для поточного значення $\overline{e}$, означає, що для деякої тестової пари $(x^o, \overline{s}^o)$ не виконується рівність (7), тобто якесь тестове зображення x^o розпізнається неправильно. Отже, для перевірки виконання (9) необхідно пересвідчитися у виконанні (7), тобто розв'язати задачу розпізнавання 1 для усіх $x^o \in X^o$.

Таким чином, алгоритми Козинця та персептрона ітеративно корегують вектор параметрів $\overline{e}$, оперуючи при цьому процедурою $\overline{s}^*_P(x, \overline{e})$. За визначеного $P = \langle A, d, f_{\overline{e}} \rangle$, вона знаходить розв'язок задачі розпізнавання 1, який визначає ту нерівність системи (9), що не виконується для поточного значення $\overline{e}$.

Алгоритм 1 Алгоритм Козинця для знаходження розв'язку (9)

1: Вибираємо довільний вектор $\overline{e} \neq 0$ із випуклої оболонки множини векторів $\{\overline{\varphi}'\}$. Наприклад, беремо його рівним будь-якому $\overline{\varphi}'(\overline{s}, \overline{s}^o_r, x^o_r) \neq 0$, $r \in \overline{1,R}$, $\overline{s} = \overline{s}^*_P(x^o_r, 0)$.

2: Шукаємо нерівність системи, яка не виконується для поточного значення $\overline{e}$:

$$\text{знайти } r \in \overline{1,R}, \text{ для якого } \overline{s}^o_r \overset{!}{\neq} \overline{s}^*_P(x^o_r, \overline{e}).$$

3: Якщо такого r немає, то **кінець**. Вектор $\overline{e}^*$ знайдено.

4: Обчислити нове значення вектора параметрів як перпендикуляр, опущений на відрізок, що з'єднує вектори $\overline{e}$ та $\overline{\varphi}' = \overline{\varphi}'(\overline{s}^*_P(x^o_r, \overline{e}), \overline{s}^o_r, x^o_r)$:

$$\overline{e} := k \cdot \overline{e} + (1 - k) \cdot \overline{\varphi}', \tag{10}$$

$$\text{де } k = \frac{(\overline{\varphi}', \overline{\varphi}')}{(\overline{e}, \overline{e}) + 2(\overline{e}, \overline{\varphi}') + (\overline{\varphi}', \overline{\varphi}')}. \tag{11}$$

Перейти на крок 2.

Алгоритм персептрона дуже схожий на алгоритм Козинця, але виглядає простішим. Застосований для розв'язку системи (9), він має наступний вигляд:

Алгоритм 2 Алгоритм персептрона для знаходження розв'язку (9)

1: Покласти $\overline{e} = 0$.

2: Шукаємо нерівність системи, яка не виконується для поточного значення $\overline{e}$:

$$\text{знайти } r \in \overline{1,R}, \text{ для якого } \overline{s}^o_r \overset{!}{\neq} \overline{s}^*_P(x^o_r, \overline{e}).$$

3: Якщо такого r немає, то **кінець**. Вектор $\overline{e}^*$ знайдено.

4: Обчислити нове значення вектора параметрів $\overline{e}$ за формулою:

$$\overline{e} := \overline{e} + \overline{\varphi}'(\overline{s}^*_P(x^o_r, \overline{e}), \overline{s}^o_r, x^o_r)$$

та перейти на крок 2.

Для того, щоб вважати задачу настройки алгоритму розпізнавання тексту повністю сформульованою і розв'язаною, нам залишилося конкретизувати лише локальну функцію відмінності $f_{\overline{e}}$. Її вигляд визначатиме спосіб обчислення вектора $\overline{\varphi}'(\overline{s}, \overline{s}^o, x)$, котрим оперують алгоритми 1 та 2.

4 Вибір локальної функції відмінності

Як було показано Ковалевським [2], припущення (3) та (4) щодо моделі зображення не надто звужують область практичного використання — алгоритм розпізнавання, який добре працює за умов сферично-симетричного шуму, буде задовільно працювати при будь-якому шумі з незалежними компонентами і не надто великою дисперсією.

Оскільки локальна функція відмінності (5) не є лінійною за параметром $\overline{e}$, тобто не може бути представлена у вигляді (6), ми розширили її до наступного вигляду:

$$f_{\overline{e}}(s, x) = \sum_{\substack{i \in \overline{1,H} \\ j = \overline{1, d(\alpha(s))}}} \ddot{e}^{\alpha}_{i,j} \cdot x^2_{i,j(s)+j} + \dot{e}^{\alpha}_{i,j} \cdot x_{i,j(s)+j} + e^{\alpha}_{i,j} \tag{12}$$

При цьому вектор параметрів формується з усіх коефіцієнтів e:

$$\overline{e} = \left\{ \ddot{e}^{\alpha}_{i,j}, \dot{e}^{\alpha}_{i,j}, e^{\alpha}_{i,j} \mid \alpha \in A, i \in \overline{1, H}, j = \overline{1, d(\alpha)} \right\},$$

а вектор $\overline{\varphi}'(\overline{s}, \overline{s}^o, x)$ складається із відповідних компонент:

$$\overline{\varphi}'(\overline{s}, \overline{s}^o, x) = \left\{ \ddot{\varphi}'^{\alpha}_{i,j}, \dot{\varphi}'^{\alpha}_{i,j}, \varphi'^{\alpha}_{i,j} \mid \alpha \in A, i \in \overline{1, H}, j = \overline{1, d(\alpha)} \right\}, \tag{13}$$

які обчислюються наступним чином:

$$\begin{aligned}
\ddot{\varphi}'^{\alpha}_{i,j} &= \sum_{\substack{s \in \overline{s} \\ \alpha(s) = \alpha}} x^2_{i,j(s)+j} - \sum_{\substack{s \in \overline{s}^o \\ \alpha(s) = \alpha}} x^{o2}_{i,j(s)+j}, \\
\dot{\varphi}'^{\alpha}_{i,j} &= \sum_{\substack{s \in \overline{s} \\ \alpha(s) = \alpha}} x_{i,j(s)+j} - \sum_{\substack{s \in \overline{s}^o \\ \alpha(s) = \alpha}} x^{o}_{i,j(s)+j}, \\
\varphi'^{\alpha}_{i,j} &= \sum_{\substack{s \in \overline{s} \\ \alpha(s) = \alpha}} 1 - \sum_{\substack{s \in \overline{s}^o \\ \alpha(s) = \alpha}} 1.
\end{aligned}$$

Експерименти з алгоритмами 1 та 2 настройки вектора параметрів $\overline{e}$ локальної функції відмінності (12) виявили, що на реальних зображеннях настройка відбувається за час, завеликий для практичного застосування цих алгоритмів. Значну кількість ітерацій, необхідних алгоритмам Козинця та персептрона, можна пояснити наступними емпіричними міркуваннями. Різні компоненти вектора параметрів $\overline{e}$ у (12) нерівномірно впливають на значення локальної функції відмінності $f_{\overline{e}}(s, x)$. Внаслідок цього, кожна зміна значення коефіцієнта $\ddot{e}^{\alpha}_{i,j}$ вимагає адекватного налаштування $\dot{e}^{\alpha}_{i,j}$, а зміна $\dot{e}^{\alpha}_{i,j}$ аналогічним чином призводить до необхідності додаткової настройки $e^{\alpha}_{i,j}$.

Однаковий вплив різних компонент вектора параметрів $\overline{e}$ на значення $f_{\overline{e}}(s,x)$ забезпечується ортонормованим базисом, в якому може бути записана функція $f_{\overline{e}}$:

$$f_{\overline{e}}(s,x) = \sum_{\substack{i\in\overline{1,H} \\ j=\overline{1,d(\alpha(s))}}} \ddot{e}^{\alpha}_{i,j}\cdot\psi_2\left(x_{i,j(s)+j}\right) + \dot{e}^{\alpha}_{i,j}\cdot\psi_1\left(x_{i,j(s)+j}\right) + e^{\alpha}_{i,j}\cdot\psi_0\left(x_{i,j(s)+j}\right), \tag{14}$$

де функції $\psi_i(x)$ – ортонормовані поліноми Чебишева. Кожна $\psi_i(x)$ є поліномом i-го порядку від сигналу x. При цьому вони пов'язані між собою наступним обмеженням:

$$\int\limits_{x\in V} \psi_i(x)\cdot\psi_j(x)\,dx = \begin{cases} 1, & i=j; \\ 0, & \text{інакше.} \end{cases}$$

Компоненти (13) вектора $\overline{\varphi}'(\overline{s},\overline{s}^o,x)$ в цьому базисі набувають такого вигляду:

$$\begin{aligned}
\ddot{\varphi}'^{\alpha}_{i,j} &= \sum_{\substack{s\in\overline{s} \\ \alpha(s)=\alpha}} \psi_2\left(x_{i,j(s)+j}\right) - \sum_{\substack{s\in\overline{s}^o \\ \alpha(s)=\alpha}} \psi_2\left(x^o_{i,j(s)+j}\right); \\
\dot{\varphi}'^{\alpha}_{i,j} &= \sum_{\substack{s\in\overline{s} \\ \alpha(s)=\alpha}} \psi_1\left(x_{i,j(s)+j}\right) - \sum_{\substack{s\in\overline{s}^o \\ \alpha(s)=\alpha}} \psi_1\left(x^o_{i,j(s)+j}\right); \\
\varphi'^{\alpha}_{i,j} &= \sum_{\substack{s\in\overline{s} \\ \alpha(s)=\alpha}} \psi_0\left(x_{i,j(s)+j}\right) - \sum_{\substack{s\in\overline{s}^o \\ \alpha(s)=\alpha}} \psi_0\left(x^o_{i,j(s)+j}\right).
\end{aligned}$$

При використанні локальної функції відмінності (14), час роботи алгоритмів 1 та 2 при настройці на реальних зображеннях зменшився у три-п'ять разів.

5 Результати експериментів

Ми перевірили роботу алгоритмів настройки 1 та 2 на зображеннях двох типів: власноруч згенерованих та реальних, що виникли за певних прикладних обставин.

За допомогою штучних зображень ми змоделювали ситуацію, коли еталонні зображення літер недоступні. У цьому випадку за невідомого характеру шуму можна використати припущення (3) та (4) для постановки задачі оцінювання еталонів літер на основі прикладів їх зображень. Її розв'язання методом максимальної правдоподібності зводиться до пошуку усереднених шаблонів літер, доступних у навчальній вибірці зображень текстових рядків. Використання отриманих еталонів у якості вектора параметрів алгоритму розпізнавання із локальною функцією відмінності (5) ми у подальшому називатимемо *«підхід із усередненням шаблонів літер»*.

Для локальної функції відмінності вигляду (12) або (14), компоненти вектора параметрів $\overline{e}$ не можуть розглядатися як еталонні зображення літер. Тому його пошук пропонується здійснювати за допомогою розв'язання задачі настройки 2.

5.1 Приклади, коли усереднення шаблонів літер не працює

У випадку, коли припущення підходу із усередненням шаблонів літер виявляються неправильними, отримане значення вектора параметрів $\overline{e}$ не дає правильного розпізнавання навіть тих навчальних зображень, які використовувались для усереднення. Водночас, алгоритм розпізнавання, отриманий в результаті розв'язання задачі настройки 2, позбавлений цієї вади.

Рисунки 1 та 2 демонструють описану ситуацію. Для побудови еталонів літер використовувалося навчальне зображення 1, яке містить один зразок літери «С» та сім зразків літери «Е», останній з яких було спотворено. Після усереднення спотворене зображення літери «Е» виявляється «більш схожим» на еталон літери «С» і тому на зображенні 2 розпізнається неправильно. На навчальному зображенні ця помилка теж має місце — воно розпізнається як «СЕЕЕЕЕЕС».

Рисунок 1: Зображення для настройки усередненням шаблонів літер «С» та «Е»

$\Rightarrow$ CCE

Рисунок 2: Спотворене зображення літери «Е» розпізнається неправильно

При використанні запропонованих алгоритмів настройки та локальних функцій відмінності, зображення 1 та 2 розпізнаються правильно.

Кількісно, але не якісно складніший приклад показано на рисунках 3 та 4. Усереднення знову призводить до неправильного розпізнавання, а запропонований підхід дає безпомилковий результат.

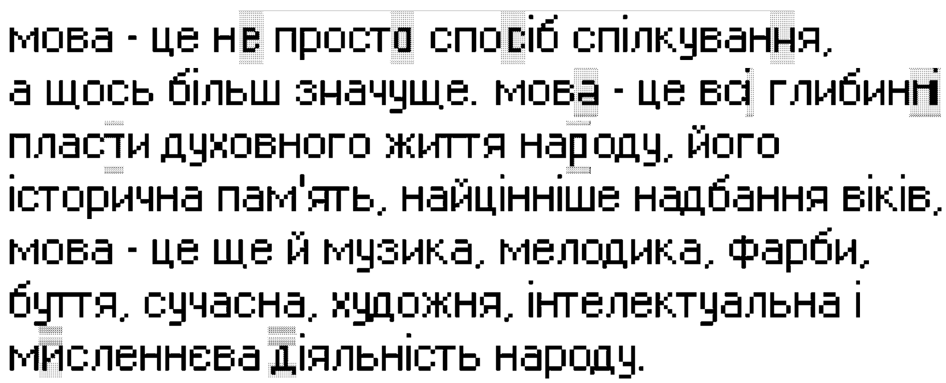


Рисунок 3: Зображення для настройки. Спотворені зображення літер позначено сірим

грандіозні речі робляться грандіозними
засобами, одна понрода робить велике даром

$$\Downarrow$$

граидідзн'_реч'_рдблч_.ься_граидідзиимн_
зесдбакіи,_ддиа_понрдда_рдбить'ітелике_оардкі

Рисунок 4: Неправильне розпізнавання на основі шаблонів, усереднених за зображенням 3

5.2 Настройка на реальних зображеннях

Зображення 5 використовувалося для настройки параметрів локальної функції відмінності вигляду (14). Отриманий вектор оптимальних параметрів використовувався для розпізнавання іншого зображення, результат якого наведений на рисунку 6. Настройка відбувалася за допомогою алгоритму Козинця. Для вектора параметрів, знайденого за допомогою алгоритму персептрона, результат розпізнавання зображення 6 аналогічний. Як бачимо, він містить деякі помилки. Їх можна пояснити тим, що деякі літери на навчальному зображенні 5 зустрічалися лише один-два рази, або ж взагалі не зустрічалися.

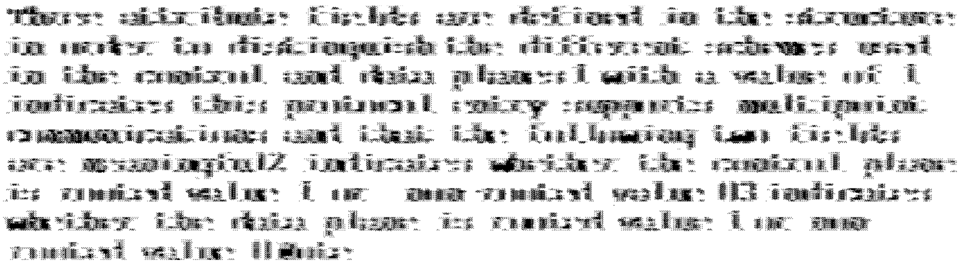
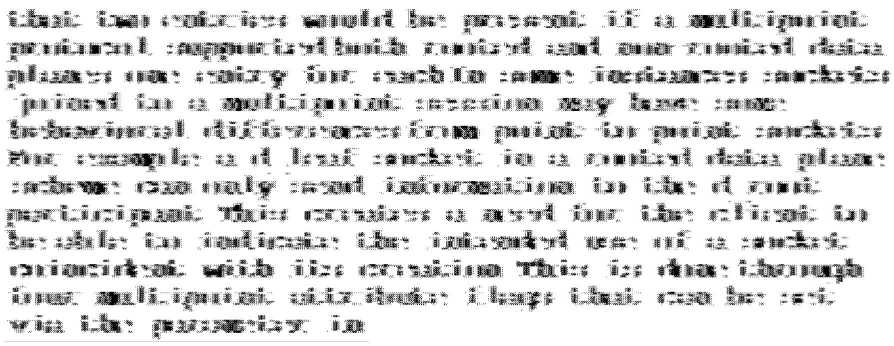


Рисунок 5: Приклад реального зображення для настройки



$$\Downarrow$$

that_two_entries_wodd_he_present_if_a_multipoint
protocol_supported_hoth_rooted_and_nonerooted_data
planes_one_entry_for_each1n_some_instances_socbets
ioined_to_a_multtipoint_session_mup_h_oe_some
heharioral_differences_from_point_so_point_socaets
ror_enample_a_d_leaf_socaet_in_a_rooted_data_plane
scheme_can_only_send_information_to_the_d_root
participont_1his_creates_a_need_for_the_client_to
he_ahle_to_indicate_the_intended_use_of_a_socaet
coiucident_with_its_creation_This_is_done_through_four
multipoint_attrihute_flags_that_can_he_set
via_the_parameter_in

Рисунок 6: Приклад розпізнавання реального зображення

5.3 Порівняння часу роботи алгоритмів настройки

У своєму розпорядженні ми мали реальні зображення трьох рівнів зашумленості, що дозволило нам порівняти час роботи алгоритмів настройки Козинця та персептрона за використання локальної функції відмінності вигляду (14) та (12).

Результати представлені в таблиці 1.

Зображення	вигляд $f_{\overline{e}}$	Козинець		персептрон	
		ітерації	час	ітерації	час
cGood	(14)	845	00:08:04	688	00:05:54
	(12)	3221	00:31:30	4897	00:42:10
cBad	(14)	3335	00:55:30	4096	01:08:00
	(12)	18778	03:02:10	34417	05:02:00
cVeryBad	(14)	34078	09:30:00	59396	16:30:00

Табл. 1: Час роботи алгоритмів настройки Козинця та персептрона на реальних зображеннях різних рівнів зашумленості

З'ясувалося, що час роботи алгоритмів настройки збільшується із зростанням рівня зашумленості навчального зображення-зразка. Алгоритм Козинця загалом виявився швидшим за алгоритм персептрона — лише на зображенні найкращої якості простіший алгоритм персептрона закінчив роботу першим. Із погіршенням якості зображень виграш у часі алгоритму Козинця зростає. На найгірших зображеннях алгоритм Козинця виявився майже удвічі швидшим за алгоритм персептрона.

Висновки

Результати експериментів свідчать, що запропонований підхід настройки алгоритмів розпізнавання може бути використаний для побудови систем розпізнавання тексту, особливо коли вхідні зображення містять шум із невизначеними характеристиками.

Запропонований підхід дозволяє створювати системи розпізнавання, в яких параметри не є фіксованими, а корегуються в процесі експлуатації системи. Результати корекції помилок розпізнавання, виконаної користувачем чи автоматично, можна подавати як вхідні дані для алгоритму настройки і, таким чином, зменшувати кількість подібних помилок у подальшому. Відокремлення вектора параметрів $\overline{e}$ дозволяє створювати набори його оптимальних значень, спеціалізовані для зображень зі спотвореннями різного типу.

Використання запропонованих алгоритмів настройки вимагає ретельного вибору базису локальної функції відмінності. В нашому випадку, перехід до ортонормованих поліномів Чебишева визначальним чином вплинув на можливість роботи із зображеннями реальних текстів.

Література

[1] Ковалевский В.А. Оптимальный алгоритм распознавания некоторых последовательностей изображений. *Кибернетика*, (4), 1967.

[2] Ковалевский В.А. *Методы оптимальных решений в распознавании изображений.* Наука, Москва, 1976.

[3] Ковалевский В.А. Корреляционный метод распознавания изображений. *Журнал вычислительной математики и математической физики*, 2(4), 1962.

[4] G.E. Kopec and Lomelin M. Document-specific character template estimation. In *IS&T/SPIE 1996 Intl. Symposium on Electronic Imaging: Science & Technology*, San Jose, CA, Jan. 27–Feb. 2, 1996.

[5] Philip A. Chou, Gary E. Kopec. A stochastic attribute grammar model of document production and its use in document image decoding. In H. Baird L. Vincent, editor, *Document Recognition II*, volume 2422 of *SPIE Proc.*, pages 66–73, 1995.

[6] Michail I. Schlesinger, Vaclav Hlavač. *Ten lectures on statistical and structural pattern recognition.* Kluwer Academic Publishers, Dordrecht/Boston/London, 2002.