Towards a segmentation and recognition-free approach for content-based
document image retrieval of handwritten queries

Houssem Chatbri∗, Keisuke Kameyama† and Paul Kwan‡
∗Department of Computer Science, Graduate School of Systems and Information Engineering
†Faculty of Engineering, Information and Systems
∗†University of Tsukuba, Tsukuba City 305-8573, Japan
‡School of Science and Technology, University of New England, Armidale NSW 2351, Australia
∗chatbri@adapt.cs.tsukuba.ac.jp , †kame@adapt.cs.tsukuba.ac.jp, ‡paul.kwan@une.edu.au

Abstract

We introduce a method for content-based document image retrieval (CBDIR) of handwritten queries that is both segmentation and recognition-free. We first demonstrate that our method is underpinned by a theoretical model that exploits the Bayes’ rule. Next, we present an algorithmic implementation that takes into account real world retrieval challenges caused by handwriting fluctuations and style variations. Our algorithm operates as follows: First, a number of connected components of the query are matched against the connected components of the document image using shape features. A similarity threshold is used to select the connected components of the document image that are most similar to the query components. Then, the selected components are used to detect candidate occurrences of the query in the document image by using size-adaptive bounding boxes. Finally, a score is calculated for each candidate occurrence and used for ranking. We conduct a comparative evaluation of our method on a dataset of 200 printed document images, by executing 40 printed and 200 handwritten queries of mathematical expressions. Experimental results demonstrate competitive performances expressed by $P$-Recall = 100%, $A$-Recall = 99.95% for printed queries, and $P$-Recall = 73.5%, $A$-Recall = 57.92% for handwritten queries, outperforming a state-of-the-art CBDIR algorithm.

1. Introduction

Digital document images are a widely used medium of storing information that benefits from the availability of inexpensive storage media and the achievements in image processing and pattern recognition technologies. Contrary to conventional archives, digital document images save a significant amount of time and labor work by providing input for applications such as automatic document indexing and classification, which makes image retrieval in large databases feasible in a reasonable time [7].

In addition to retrieval systems that use text queries, content-based approaches have been introduced to allow retrieval using queries that cannot be expressed in text (e.g. drawings, mathematical expressions, signatures) [11]. In case of printed documents and text queries, retrieval is often done by using optical character recognition (OCR) to perform recognize-then-retrieve [5]. On the other hand, when documents are degraded, handwritten, and in case of non-text queries, different approaches are used. In the latter scenario, features are extracted from the query and used to spot its occurrences inside the document image [9].

Recognition-free methods usually proceed by segmenting the document image into words and characters. Then, the query is matched with the segmented words using shift estimation algorithms [9]. Some methods incorporate a priori knowledge about the document’s language such as Chinese by using the proximity between character strokes [6] and Arabic by using specific character features (e.g. diacritics, loops, etc.) [10]. Approaches for non-word queries have been introduced [2]. For instance, signature-based document retrieval is achieved using characteristic multi-scale structural saliency of signatures [15]. Mathematical expression-based document retrieval is achieved using document image X-Y cutting and query matching inside a tree representation of the document image [13].

In this work, we introduce a CBDIR method that is both segmentation and recognition-free. Our method is underpinned by a theoretical model that exploits the Bayes’ rule (Sec. 2) and introduces an algorithmic implementation that takes into account handwriting fluctuations and noises (Sec. 3). Using a dataset of 200 printed document images and ex-
cutting 40 printed queries and 200 handwritten queries, our approach outperforms a state-of-the-art method (Sec. 4).

2. Theoretical model

A CBDIR method relies on a spotting stage that is finding the location of a query inside a larger document image. Our method for spotting mimics the intuitive way humans follow to perform the same task when unable to read or identify the query. The human’s analogy can be illustrated by the example of a foreign tourist who tries to find a station’s name on a map that is written in the local language script, to which we suppose the tourist is totally unfamiliar. Our method for spotting mimics the intuitive way humans follow to perform the same task when unable to read or identify the query.

2.1. Prior knowledge

The query image and the document image contain equations, words, figures, drawing, etc. When considered from a low level point of view, the image contains connected components that can be alphabets, symbols, geometrical primitives, etc. The connected components, or simply components, of the query image \( I_Q \) and the document image \( I_{DOC} \) are denoted \( \{C_i^Q\}_{i=1}^M \) and \( \{C_j^{DOC}\}_{j=1}^N \) respectively, where \( M \) and \( N \) are the number of components in \( I_Q \) and \( I_{DOC} \).

Each query component \( C_i^Q \) defines a class \( \omega_i \). \( \{C_j^{DOC}\}_{j=1}^N \) are treated as patterns to be classified into a class among \( \{\omega_i\}_{i=1}^M \), corresponding to \( \{C_i^Q\}_{i=1}^M \).

A component classifier is used to calculate \( P(\omega_i|C_j^{DOC}) \), which is the probability that \( C_j^{DOC} \) corresponds to the class \( \omega_i \). Each component \( C_j^{DOC} \) is then assigned the class \( \omega_i \) having the largest probability. For each document image component \( C_j^{DOC} \), we have

\[
\sum_{i=1}^M P(\omega_i|C_j^{DOC}) = 1.
\]

After attribution to a class among \( \{\omega_i\}_{i=1}^M \), the components \( \{C_j^{DOC}\}_{j=1}^N \) are used to form candidates of \( I_Q \) occurrences in \( I_{DOC} \). A candidate is a set \( A \) of document image components and it is defined as follows:

\[
A = \{C_j^{DOC}\}_{\phi(i)}^M \quad (1)
\]

where \( \phi(i) \) is a function that returns the index \( j \) of \( C_j^{DOC} \) that is assigned to \( \omega_i \). \( \phi(i) \) insures that \( A \) has a document image component from each class \( \omega_i \).

At this stage, \( A \) is a relevant candidate if it contains components from the majority of the classes \( \{\omega_i\}_{i=1}^M \). \( A^R \) denotes the event that a set \( A \) is a relevant candidate. We take as the initial probability of \( A^R \) as follows:

\[
P(A^R) = \prod_{i=1}^M P(\omega_i|C_j^{DOC}) \quad (2)
\]

Here, we assume that \( \{P(\omega_i|C_j^{DOC})\}_{i=1}^M \) are independent.

2.2. Observation

Eq. 2 does not take into account the locations of components relative to each other inside a candidate \( A \). Therefore, multiple (\( K \)) observations \( x = [x_1 \ldots x_K] \) concerning the global resemblance and suitability of \( A \) are introduced by way of a likelihood function \( p(x|A^R) \).

2.3. Inference

The evidence provided by \( x \) is used to update the relevance probability using Bayes’ theorem:

\[
P(A^R|x) = \frac{p(x|A^R) \times P(A^R)}{p(x)} \quad (3)
\]

which shows that the posterior probability \( P(A^R|x) \) is maximized when the quantity \( p(x|A^R) \times P(A^R) \) is maximized. We have \( P(A^R|x) \propto p(x|A^R) \times P(A^R) \).

2.4. Decision function

Using the \( \ln \) operator, the decision function is expressed as follows:

\[
D(A) = \ln(P(A^R|x)) = \ln(p(x|A^R)) + \sum_{i=1}^M \ln(P(\omega_i|C_j^{DOC})) \quad (4)
\]

Therefore, a candidate \( A \) that maximizes \( D(A) \) can be judged to be relevant to query \( I_Q \).

3. Algorithmic implementation

Our algorithm proceeds as follows: First, features are extracted from the components of \( I_Q \) and \( I_{DOC} \) (Sec. 3.1) and used to detect candidate occurrences of \( I_Q \) in \( I_{DOC} \) (Sec. 3.2). Next, a score is calculated for each candidate to express its relevance to the query (Sec. 3.3).
3.1. Component feature extraction and matching

A feature vector $V$ is produced for each component of $I_Q$ and $I_{DOC}$. Features are extracted by calculating the distribution of pixels inside a bounding circular layout of which the origin is the component’s centroid (Fig. 1). The similarity between two components $C_i$ and $C_j$ is equivalent to the histogram intersections between their corresponding vectors, which is calculated as follows:

$$S(C_i, C_j) = \sum_{r=0}^{R-1} \sum_{\theta=0}^{\Theta-1} \min(V_{r,\theta}, V_{r,\theta})$$

where $R$ and $\Theta$ refer to the radial and angular number of sections. Two components $C_i$ and $C_j$ are considered similar if they satisfy $S(C_i, C_j) \geq \alpha$, where $\alpha \in [0, 1]$ is a similarity threshold. $S(C_i, C_j)$ is the practical implementation of $P(\omega_i | C_j^{DOC})$ defined in Sec. 2.1.

3.2. Detection of query occurrence candidates

One component of the query, that we call main component $\hat{Q}$, is determined and used as a seed for candidate occurrence detection. In this implementation, $\hat{Q}$ is chosen as the largest component in terms of number of pixels. Then, components of $I_{DOC}$ which are similar to $\hat{Q}$ are detected. The set of components of $I_{DOC}$ which are similar to $\hat{Q}$ is denoted $B = \{C^{DOC}_j | S(\hat{Q}, C_j) \geq \alpha : 1 \leq j \leq N\}$. The neighboring components of an element of $B$ possibly belong to an occurrence of $I_Q$ in $I_{DOC}$ and they are extracted to form a candidate $A$. Neighboring components extraction is done using a bounding box that is calculated using the query’s dimensions $(W_Q, H_Q)$ and $Q$ (Fig. 2). The bounding box’s dimensions are calculated as follows:

$$(W, H) = (W_Q, H_Q) \times \frac{\text{size of } \hat{Q}}{\text{size of } \hat{Q}} \times \beta$$

where $\hat{Q}$ denotes a match of $Q$ in $B$, size of a component is expressed by the number of its pixels, and $\beta$ is a parameter to control the size of the bounding box which is introduced to account for handwriting fluctuations. The normalization using the components’ sizes makes the boxes size-adaptive.

In order to account for component disconnectedness or merging, spotting is done using a number $N_Q$ of main components instead of one. The extracted main components are the $N_Q$ largest components of $I_Q$. In addition, a candidate $A$ is discarded when $|N_A - M| > \eta$, where $N_A$ and $M$ refer to the number of components in $A$ and $I_Q$ respectively, and $\eta \in [0, N_A]$ is a threshold.

3.3. Candidate score

The last step is to compute a score for each set $A$ that expresses its relevance as a query occurrence candidate. For this purpose, $p(x|A^R)$ is estimated for a multidimensional observation $x = [a \ b \ c]$, where:

$$\phi_0(x) = 1 \quad \phi_1(x) = x$$
$$\phi_2(x) = \frac{3x^2 - 1}{2} \quad \phi_3(x) = \frac{5x^3 - 3x}{2}$$
$$\phi_4(x) = \frac{35x^4 - 30x^2 + 3}{8} \quad \phi_5(x) = \frac{63x^5 - 70x^3 + 15x}{8}$$

Figure 1. Feature extraction from connected components: (a) Feature extraction from connected components. (b) Illustration of a feature vector (the brighter the bin region the larger the value).

Figure 2. Illustration of the bounding box-based spotting procedure: (a) An example of a handwritten query with the main component $\hat{Q}$ highlighted in green. (b) Matches of $\hat{Q}$ are highlighted in green. The blue bounding box refers to a relevant candidate, and the two red bounding boxes refer to irrelevant candidates (other red bounding boxes are omitted for clarity).

- $a = S(A, I_Q)$ is the matching score between the image produced by $A$ and the query $I_Q$ using a shape descriptor (Sec. 3.1). Specifically here, the feature extraction layout’s centroid corresponds to the centroid of $\hat{G}$ instead of the global centroid of $A$, and all components’ points located inside the circular layout are considered.
- $b$ is equivalent to the maximum value of $a$ when calculated for the large components of $A$. The large components of $A$ are the components having their sizes superior to the average component size in $A$. $b$ is introduced to account for component disconnectedness and merging.
- $c$ is equal to the number of query components that have similar counterparts in the candidate divided by the total number of query components. $c$ is introduced to penalize cases when a single component of the query is matched to several components by mistake.

The scores $a$, $b$, and $c$ are normalized and fall in the interval $[0, 1]$. Large values indicate similarity between $A$ and $I_Q$ while small values indicate dissimilarity.

Assuming that the components of $a$ are independent, we integrate $p(x|A^R) = p(a|A^R) \ p(b|A^R) \ p(c|A^R)$ in Eq. 4, which gives:

$$D(A) = \ln(P(a|A^R)) + \ln(P(b|A^R)) + \ln(P(c|A^R))$$
$$+ \sum_{i=1}^{M} \ln(P(\omega_i | C^{DOC}_{\phi(i)}))$$

(7)
Each candidate $A$ is assigned a score that expresses its relevance as a query occurrence candidate as follows:

$$score(A, \gamma) = \ln(1 + a) + \ln(1 + b) + \ln(1 + c) + \frac{\gamma}{N_A} \sum_{u=1}^{N_A} \ln(1 + \max_{1 \leq i \leq M} S(C_i, C_u))$$

(8)

where $1$ is added to avoid the $\ln$ of a zero probability. Finally, the candidates are ranked in their descending scores.

$score(A, N_Q)$ is a direct implementation of the theoretical model (Eq. 4). When component disconnectedness and merging in a candidate $A$ are significant, the quantity of the score $\sum_{u=1}^{N_A} \ln(1 + \max_{1 \leq i \leq M} S(C_i, C_u))$ accumulates incorrect similarity values that eventually increase the score and cause $A$ to be judged as relevant to query $I_Q$ incorrectly. The parameter $\gamma \in [0, N_Q]$ is thus introduced to mitigate this effect.

4. Experimental Results

We compare our method with Zanibbi and Yu’s CBDIR method [13]. This method is adequate for comparison as it is based on segmentation using horizontal and vertical cutting and tree-based feature representation, which are often used in CBDIR of mathematical expressions [12]. We conduct retrieval experiments using their dataset which contains 40 printed queries, 200 handwritten queries provided by 10 users, and 200 printed document images that are collected from CVPR 2008 conference proceedings. The document images’ size are $2560 \times 3310$ pixels and resolution is 300dpi. Binarization is applied on the document images [8]. Then, thick and large components are filtered out. A component is considered thick if its contour pixels are less than 30% of its total pixels, and large if the total number of pixels exceeds 1000 pixels. This procedure filters out on average 22.85% of each document image foreground pixels corresponding mostly to figures. Thinning is applied on queries in order to extract skeletons [14] which are an adequate compact shape representation for handwriting [3]. In order to maintain a reasonable processing time, we apply dataset indexing using connected component clustering [4].

4.1. Evaluation procedure

The best results have been obtained with the empirical setting of parameters as follows: The number of query main components is $N_Q = 10$, and the candidates recovered from a main component are discarded if their number exceeds 2000. The scale normalization parameter is $\beta = 1.1$. The radial and angular numbers of sections of the shape descriptor are $R = 5$ and $\Theta = 10$. The parameter for controlling the minimum number of components in a candidate $A$ is candidate-dependent and set $\eta = \frac{N_A}{N_e}$, where $N_A$ is the number of components in $A$. The parameter for mitigating component disconnectedness and merging is set $\gamma = 2$.

Table 1. Average values of P-Recall and A-Recall calculated for $n = 1, 5, 10$ and for $\alpha = 0.5$. Boldface indicates the best results.

<table>
<thead>
<tr>
<th>Queries</th>
<th>Printed</th>
<th>Handwritten</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our method</td>
<td>$n$</td>
<td>P-Recall</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>85.0%</td>
</tr>
<tr>
<td>5</td>
<td>100.0%</td>
<td>94.29%</td>
</tr>
<tr>
<td>10</td>
<td>100.0%</td>
<td>98.56%</td>
</tr>
<tr>
<td>(\gamma = 2)</td>
<td>1</td>
<td>100%</td>
</tr>
<tr>
<td>5</td>
<td>100%</td>
<td>99.95%</td>
</tr>
<tr>
<td>10</td>
<td>100%</td>
<td>99.95%</td>
</tr>
<tr>
<td>Zanibbi and Yu [13]</td>
<td>1</td>
<td>.</td>
</tr>
<tr>
<td>5</td>
<td>.</td>
<td>90%</td>
</tr>
<tr>
<td>10</td>
<td>.</td>
<td>90%</td>
</tr>
</tbody>
</table>

Evaluation is done using the $P$-Recall and $A$-Recall metrics similarly to [13]. They are calculated for $n$ retrieved images as follows:

$$P\text{-Recall} = \frac{|\text{Relevant retrieved images}|}{|\text{Relevant images in dataset}|} \times 100 \quad (9)$$

$$A\text{-Recall} = \frac{\text{Candidate bound. box area}}{\text{Ground truth bound. box area}} \times 100 \quad (10)$$

$P$-Recall indicates the algorithm’s ability to retrieve the document images containing the query, and $A$-Recall indicates the ability to spot the correct area of the relevant query’s occurrence in the document image. Since $n$ is fixed, both metrics express the precision of the algorithm.

4.2. Results

Table 1 shows results of using our method compared with Zanibbi and Yu’s method. Adapting the model by setting $\gamma = 2$ in $score(A, \gamma)$ (Eq. 8) largely outperforms the direct implementation of the model, which shows that a small weight should be given to similarity values between CCs in order to mitigate their frequent disconnectedness and merging. When $n = 10$ images are retrieved and $\alpha = \{0.4, 0.5\}$, retrieval performances are expressed by $P$-Recall = 100%, $A$-Recall = 99.95% for printed queries and $P$-Recall = 73.5%, $A$-Recall = 57.92% for handwritten queries.

Performances in case of printed and handwritten queries behave differently when the similarity threshold $\alpha$ is changed (Fig. 3). For printed queries, performances remain stable for values of $\alpha$ up to 0.6, while performances of handwritten queries start to decrease when $\alpha > 0.5$. This result is explained by the neat quality of printed fonts contrary to fluctuations and noise in handwritten queries. Maximal performances for both printed and handwritten queries can be achieved by loosely setting $\alpha$ in the range $[0.4, 0.5]$. However, this increases matching computations as a large number of CCs will be recovered from the indexed dataset.
5. Conclusion and future work

In this paper, we presented a CBDIR method using handwritten queries that is both segmentation and recognition-free. Our method is underpinned by a theoretical model that exploits Bayes’ rule and introduces an algorithmic implementation that copes with handwriting fluctuations and noises. Comparative evaluation with a state-of-the-art method demonstrated that our method is competitive.

In addition to being segmentation and recognition-free, our method is also highly modular. Future improvement will focus on enhancing each of its modules, and conducting evaluation in different problem domains. On the other hand, parameter setting is sought to be made automatically data-driven by analyzing the similarity values between the query’s CCs and the CCs in the indexed dataset.

Fig. 4 shows retrieval performances per user. When the handwriting fluctuations and component displacement are limited, performances are high (e.g. user 5 having $P$-Recall = 90% and $A$-Recall = 81.02%). Lower performances by other users were caused by significant component alteration and displacement. An example is user 7 who led to lower results because of their compact writing style, reported similarly by Zanibbi and Yu’s [13].

Our method outperforms Zanibbi and Yu’s method in case of printed and handwritten queries for $n = \{1, 5, 10\}$. This result is due to two fundamental differences between the methods: (1) Zanibbi and Yu’s algorithm uses an X-Y cutting-based segmentation step that produces a tree index of the query and the document image. The authors point to the weakness of X-Y cutting when handling handwritten queries especially to variations of the gap between characters. In contrast, our method is segmentation-free, which spares it from erroneous segmentation results. (2) After building the tree indexes, Zanibbi and Yu’s method uses a set of features to represent indexed regions. Among the features, they rely on structural features such as tree depth and the number of nodes. Structural features are vulnerable to noisy patterns such as handwriting [1]. On the other hand, our method relies essentially on statistical features.

References