Keyword Spotting in Historical Documents Based on Handwriting Graphs and Hausdorff Edit Distance

Mohammad R. Ameri [a] , Michael Stauffer [b,c] , Kaspar Riesen [b] , Tien D. Bui [a] and Andreas Fischer [d,e]

[a] *Concordia University, Department of Computer Science and Software Engineering, Montréal, Québec, H3G 1M8, Canada*

[b] *University of Applied Sciences and Arts Northwestern Switzerland, Institute for Information Systems, 4600 Olten, Switzerland*

[c] *University of Pretoria, Department of Informatics, 0083 Pretoria, South Africa*

[d] *University of Fribourg, Department of Informatics, 1700 Fribourg, Switzerland*

[e] *University of Applied Sciences and Arts Western Switzerland, Institute for Complex Systems, 1705 Fribourg, Switzerland*

mo_amer@encs.concordia.ca, michael.stauffer@fhnw.ch, kaspar.riesen@fhnw.ch, bui@cse.concordia.ca, andreas.fischer@unifr.ch

Abstract. Keyword spotting facilitates the indexation and organization of scanned historical manuscripts. In order to represent the structure and the shape of handwritten words, several graph-based approaches have been proposed recently with a view to template-based keyword spotting. Yet the high time complexity for graph matching, typically cubic-time or worse, imposes computational limits for processing large volumes of scanned documents. In this paper, we propose a novel quadratic-time method for graph-based keyword spotting using the Hausdorff edit distance. In an experimental evaluation on the George Washington benchmark dataset, we demonstrate that graph matching based on Hausdorff edit distance is a promising alternative to the standard sequence matching approach based on dynamic time warping, both in terms of accuracy and runtime.

Keywords: Keyword Spotting, Handwriting Graphs, Graph Matching, Hausdorff Edit Distance.

1. Introduction

Modelling and recognition of handwritten text is far more challenging than printed text, as handwriting does not have a fixed structure and exhibits variable character shapes. Often, an automatic transcription is not feasible or is not accurate enough for historical manuscripts, especially when facing ancient scripts and languages with a lack of training data. Keyword spotting [1] has been suggested as an alternative for indexing scanned manuscripts without performing a complete transcription.

In general, we can distinguish *template-based* and *learning-based* keyword spotting methods. Template-based methods [2, 3, 4, 5] match one or several instances of a keyword directly with the document image, while learning-based methods [6, 7, 8, 9] aim to learn models from labeled training samples. Typically, character models are learned because they are shared across different learning samples. Learning-based methods can achieve a significantly better performance but are less flexible as they need labeled training data. In this paper, we focus on word image matching for template-based keyword spotting.

Taking into account the variable width of the handwriting, a widely adopted approach to template matching is to move a sliding window over the word images for extracting a sequence of feature vectors, such as projection profiles [2], gradient histograms [3, 4], or, more recently, deep learning features [10]. Feature vector sequences can then be matched efficiently by means of dynamic time warping (DTW), which has a quadratic time complexity with respect to the length of the sequences.

A limitation of feature vector sequences is that they capture the structure of the handwriting only in one dimension, yet handwriting has a rich two-dimensional structure. While a fixed number of features can describe this structure indirectly, a more natural and explicit representation is achieved by means of graphs. In structural pattern recognition, graphs can be used to describe parts of an object with nodes and relations between these parts with edges [11]. In recent work, several graph-based approaches have been proposed for template-based keyword spotting, using keypoints [5, 12, 13] or basic strokes [14, 15] as nodes and connecting them with edges if they are connected in the image.

However, the high representational power of graphs comes at the cost of a high computational complexity. In particular, the graph matching method employed in [5, 12, 13, 14, 15] is the well-established bipartite approximation (BP) [16] of graph edit distance (GED), which has a cubic time complexity with respect to the size of the graphs. Thus, significant computational constraints are imposed for keyword spotting.

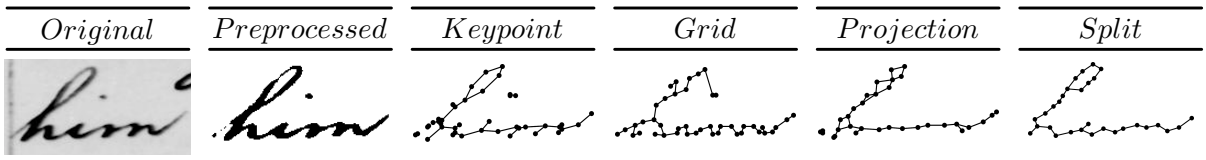

Fig. 1. Handwriting graphs.

In this paper, we propose a novel graph-based keyword spotting system with quadratic time complexity, which relies on the Hausdorff edit distance (HED) [17] for matching two word graphs. The main contribution is to introduce graph matching with HED as a promising alternative to sequence matching with DTW for template-based keyword spotting, both in terms of accuracy and runtime.
In the remainder, we first describe the graph-based handwriting representation, then introduce our keyword spotting system, and finally present an experimental evaluation on the George Washington benchmark dataset, to empirically compare the proposed method with BP and DTW.

2. Handwriting Graphs

Formally, a labeled graph is a four-tuple $g = (V, E, \mu, \nu)$ where V is the finite set of nodes, $E \subseteq V \times V$ is the set of edges, $\mu : V \to L_V$ is the node labeling function, and $\nu : E \to L_E$ is the edge labeling function. L_V and L_E can be arbitrary label alphabets, such as symbols, numbers, or vectors.
We consider four types of handwriting graphs introduced in [18]. They are extracted from binarized and skeletonized word images. Nodes represent distinct points in the image and are labeled with image coordinates $(x, y) \in L_V = \mathbb{R}^2$. Edges are unlabeled and undirected.

- `Keypoint.` The nodes of this graph represent keypoints in the skeleton image, such as junctions and end points. In addition, equidistant points along the skeleton are added to the set of nodes. Nodes are connected with an edge when they are connected in the skeleton image.
- `Grid.` This graph is based on a grid-wise segmentation of the binary image. For every non-empty segment, a node is inserted and labeled with the center of mass. Edges are inserted between neighboring segments and are reduced with the minimal spanning tree algorithm.
- `Projection.` The nodes of this graph are also inserted at the center of mass similar to `Grid` graphs. Instead of a regular grid, an adaptive segmentation is performed based on horizontal and vertical projection profiles. Edges are inserted when the segments are connected in the skeleton image.
- `Split.` This graph is constructed similar to `Projection` graphs, but the segmentation is performed iteratively in horizontal and vertical direction until the width and height of all segments is below a certain threshold. Again, edges are inserted when the segments are connected in the skeleton image.

The four handwriting graphs are illustrated in Figure 1. For more details, we refer to [18].

3. Keyword Spotting System

The proposed keyword spotting system requires segmentation, binarization, and skeletonization of word images. To focus on the word matching method, we consider a manually corrected segmentation of the page into words. Binarization relies on a difference of Gaussians and skeletonization is achieved with a 3×3 thinning operator as detailed in [18].
After representing the word images with graphs (see previous section), a structural comparison is performed for template matching. This comparison is based on the concept of graph edit distance (GED), which computes the minimum-cost transformation of graph g_1 into graph g_2 by means of deletions $(u \to \epsilon)$, insertions $(\epsilon \to v)$, and substitutions $(u \to v)$ of nodes $u \in V_1$ and $v \in V_2$. Similar edit operations are defined for the edges. Instead of the exact GED, we compute the Hausdorff edit distance (HED) [17], a recently introduced lower bound of GED that can be computed in quadratic time $O(|V_1| \cdot |V_2|)$:

$$\mathrm{HED}(g_1, g_2) = \sum_{u \in V_1} \min_{v \in V_2 \cup \{\epsilon\}} \mathcal{C}(u, v) + \sum_{v \in V_2} \min_{u \in V_1 \cup \{\epsilon\}} \mathcal{C}(u, v) \tag{1}$$

Equation 1 computes the optimal assignment of nodes in V_1 to nodes in V_2 and *vice versa*. The cost function $\mathcal{C}(u, v)$ takes into account the node edit cost $c(u \to v)$ as well as the edge edit cost $c(q \to r)$ for edges $q \in E_1$ adjacent to u and edges $r \in E_2$ adjacent to v. For more details, we refer to [17].
We consider an Euclidean cost model for HED-based keyword spotting. Similar to [5], node labels are first normalized to zero mean and unit variance with respect to the current graph. Afterwards, the substitution cost is defined as $c(u \to v) = \sqrt{\beta \cdot \sigma_x (x_1 - x_2)^2 + (1 - \beta) \cdot \sigma_y (y_1 - y_2)^2}$, where σ_x and σ_y are the standard deviations of the node labels in the original graph. The parameter $\beta \in [0, 1]$ weights the horizontal and vertical distance. Finally, the non-negative insertion and deletion costs are set to C_n for nodes and C_e for edges, respectively.

Table 1
Comparison with BP on the George Washington database. Median and maximum number of nodes, mean average precision for BP and HED, relative gain in percentage, mean runtime per graph pair in milliseconds for BP and HED, and speedup factor. The best results are highlighted in bold font.

Graph	$\|V\|_{med}$	$\|V\|_{max}$	MAP_{BP}	MAP_{HED}	Gain	T_{BP}	T_{HED}	Speedup
`Keypoint`	74	366	66.08	**69.28**	+4.83	303.0	**3.2**	95.3
`Grid`	90	509	60.02	62.78	+4.59	707.9	6.1	116.0
`Projection`	74	391	61.43	66.71	+8.59	344.1	3.9	88.1
`Split`	80	434	60.23	65.12	+8.12	480.2	4.4	108.1

Table 2
Comparison with DTW on the George Washington database. The best result is highlighted in bold font.

System	Rodriguez'08 [3, 10]	Terasawa'09 [4, 10]	Wicht'16 [10]	Proposed (HED)
MAP	63.39	64.80	68.64	**69.28**

In order to make HED comparable across different keyword sizes, we normalize it with respect to the maximum edit distance, which corresponds to the deletion of all nodes and edges in g_1 and the insertion of all nodes and edges in g_2. When several templates are available for a keyword, the minimum distance is considered.

4. Experimental Evaluation

To compare the proposed HED-based system with BP and DTW, we consider the well-known George Washington benchmark dataset containing 20 pages of letters written in 1755 by George Washington and his associates. The ground truth, which includes polygonal boundaries around the words, is adopted from [18]. Four standard cross-validations with 105 keywords, 2447 training words (used for obtaining keyword template images), and 1224 test words per cross-validation are performed as in [5, 10], among others. The performance is assessed with the mean average precision (MAP) over all keywords.

The parameters of the Euclidean cost model are optimized during validation both for HED and BP with respect to 10 selected keywords, 1000 words of the target document, and a grid search over $\beta \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$ and $C_n, C_e \in \{1, 4, 8, 16, 32\}$. The optimal configuration for HED was achieved using `Keypoint` graphs with $\beta = 0.1$, $C_n = 8$, and $C_e = 4$. Note that in [18], `Projection` and `Split` graphs performed better than `Keypoint` graphs for word recognition, suggesting that observations made for a recognition task cannot be transferred directly to a keyword spotting task.

A comparison with BP [5] is provided in Table 1. The MAP values are averaged over the four test sets of the cross-validations. The runtime is reported with respect to a Java implementation, a 4 GHz Intel Core i7 processor, and 32 GB memory. Reducing the computational complexity from cubic to quadratic time leads to a significant speedup factor of around 100 by the proposed method, considering graphs with a median number of 74-90 nodes. Furthermore, quite surprisingly, we observe that the MAP is not reduced by the faster algorithm. Instead, a relative improvement of around 4-8% could be achieved in all cases.

A comparison with DTW [10] is provided in Table 2. In terms of MAP, the proposed HED-based system is able to outperform several recent DTW-based systems, including an approach relying on deep learning features that are obtained without labeled training samples by means of unsupervised learning [10]. In terms of computational effort, the complexity of DTW is quadratic regarding the length of the matched sequences and the complexity of HED is quadratic regarding the size of the matched graphs. Considering a median sequence length of 134 for sliding window features and a median graph size of 74 for `Keypoint` graphs on the George Washington dataset, our proposed system also reduces the computational effort when compared with DTW.

5. Conclusions

The experimental evaluations on the George Washington dataset indicate that the proposed HED-based method for template-based keyword spotting is not only faster but also slightly more accurate than BP as well as several recent DTW-based systems.

When compared with BP, the reduction from cubic to quadratic time complexity enables our system to process about a hundred times more manuscripts. When compared with DTW, we could, in particular, achieve a higher accuracy than a recent approach based on deep learning [10], which relies on features learned from unlabeled data. In contrast, our system considers only (x, y) coordinates as labels, which emphasizes the promising potential of the suggested structural approach.

In order to extend the initial study presented in this paper, future work includes a more comprehensive experimental evaluation on more datasets, including recent competition benchmarks. Furthermore, there

are several promising lines of future research. First, it is important to note that HED is not constrained to Euclidean labels. Instead, it can cope with any label alphabet using an appropriate cost model. Hence it will be interesting to develop, compare, and combine different graph representations of handwritten words. Secondly, there is room for improvement regarding the matching method. For example, the use of structural node context [19] might improve the performance. Finally, we aim to investigate potential applications of HED for character modeling in the context of learning-based keyword spotting.

References

[1] Raghavan Manmatha, C. Han, and E.M. Riseman. Word spotting: A new approach to indexing handwriting. In *Proc. Int. Conf. on Computer Vision and Pattern Recogn.*, pages 631—637, 1996.

[2] T. M. Rath and R. Manmatha. Word spotting for historical documents. *Int. Journal on Document Analysis and Recognition*, 9:139–152, 2007.

[3] J.A. Rodriguez and F. Perronnin. Local gradient histogram features for word spotting in unconstrained handwritten documents. In *Proc. 11th Int. Conf. on Frontiers in Handwriting Recognition*, pages 7–12, 2008.

[4] K. Terasawa and Y. Tanaka. Slit style HOG features for document image word spotting. In *Proc. 10th Int. Conf. on Document Analysis and Recognition*, pages 116–120, 2009.

[5] M. Stauffer, A. Fischer, and K. Riesen. Graph-based keyword spotting in historical handwritten documents. In *Proc. Int. Workshop on Structural, Syntactic, and Statistical Pattern Recognition*, pages 564–573, 2016.

[6] A. Fischer, A. Keller, V. Frinken, and H. Bunke. HMM-based word spotting in handwritten documents using subword models. In *Proc. 20th Int. Conf. on Pattern Recogn.*, pages 3416–3419, 2010.

[7] V. Frinken, A. Fischer, R. Manmatha, and H. Bunke. A novel word spotting method based on recurrent neural networks. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 34(2):211–224, 2012.

[8] Jon Almazan, Albert Gordo, Alicia Fornes, and Ernest Valveny. Word spotting and recognition with embedded attributes. *IEEE Trans. PAMI*, 36(12):2552–2566, 2014.

[9] Sebastian Sudholt and Gernot A. Fink. PHOCNet: a deep convolutional neural network for word spotting in handwritten documents. In *Proc. 15th Int. Conf. on Frontiers in Handwriting Recognition*, pages 277–282, 2016.

[10] B. Wicht, A. Fischer, and J. Hennebert. Deep learning features for handwritten keyword spotting. In *Proc. 23rd Int. Conf. on Pattern Recognition*, pages 3423–3428, 2016.

[11] D. Conte, P. Foggia, C. Sansone, and M. Vento. Thirty years of graph matching in pattern recognition. *Int. Journal of Pattern Recognition and Artificial Intelligence*, 18(3):265–298, 2004.

[12] P. Wang, V. Eglin, C. Garcia, C. Largeron, J. Llados, and A. Fornes. A novel learning-free word spotting approach based on graph representation. In *Proc. 11th Int. Workshop on Document Analysis Systems*, pages 207–211, 2014.

[13] P. Wang, V. Eglin, C. Garcia, C. Largeron, J. Llados, and A. Fornes. A coarse-to-fine word spotting approach for historical handwritten documents based on graph embedding and graph edit distance. In *Proc. 22nd Int. Conf. on Pattern Recognition*, pages 3074–3079, 2014.

[14] Q. A. Bui, M. Visani, and R. Mullot. Unsupervised word spotting using a graph representation based on invariants. In *Proc. 13th Int. Conf. on Document Analysis and Recognition*, pages 616–620, 2015.

[15] P. Riba, J. Llados, and A. Fornes. Handwritten word spotting by inexact matching of grapheme graphs. In *Proc. 13th Int. Conf. on Document Analysis and Recognition*, pages 781–785, 2015.

[16] K. Riesen and H. Bunke. Approximate graph edit distance computation by means of bipartite graph matching. *Image and Vision Computing*, 27(4):950–959, 2009.

[17] A. Fischer, C. Suen, V. Frinken, K. Riesen, and H. Bunke. Approximation of graph edit distance based on Hausdorff matching. *Pattern Recognition*, 48(2):331—343, 2015.

[18] M. Stauffer, A. Fischer, and K. Riesen. A novel graph database for handwritten word images. In *Proc. Int. Workshop on Structural, Syntactic, and Statistical Pattern Recognition*, pages 553–563, 2016.

[19] A. Fischer, S. Uchida, V. Frinken, K. Riesen, and H. Bunke. Improving Hausdorff edit distance using structural node context. In *Proc. 10th Int. Workshop on Graph-based Representations in Pattern Recognition*, pages 148–157, 2015.