Graph Similarity Features for HMM-Based Handwriting Recognition in Historical Documents

Andreas Fischer, Kaspar Riesen, and Horst Bunke
Institute of Computer Science and Applied Mathematics
University of Bern
Neubrückstrasse 10, 3012 Bern, Switzerland
{*afischer,riesen,bunke*}*@iam.unibe.ch*

***Abstract*—Automatic transcription of historical documents is vital for the creation of digital libraries. In this paper we propose graph similarity features as a novel descriptor for handwriting recognition in historical documents based on Hidden Markov Models. Using a structural graph-based representation of text images, a sequence of graph similarity features is extracted by means of dissimilarity embedding with respect to a set of character prototypes. On the medieval Parzival data set it is demonstrated that the proposed structural descriptor significantly outperforms two well-known statistical reference descriptors for single word recognition.**

***Keywords*-Handwriting recognition; Hidden Markov models**

I. Introduction

Handwriting recognition of scanned or photographed images is still a widely unsolved problem in pattern recognition and an active area of research. In particular, the interest in the recognition of handwritten text in historical documents has grown strongly in recent years [1]. Automatic recognition and transcription is needed to make large collections of historical handwritings amenable to searching and browsing in digital libraries [2].

The recognition of historical documents is an offline task. In contrast to online recognition, where time information about the writing process is available, offline recognition is performed solely on text images. Here, a high recognition accuracy can be achieved for small, specific domains such as address or bankcheck reading. For unconstrained text recognition, only few systems exist that are able to cope with an extensive vocabulary and different writing styles. For a survey, see [3]. A widely used type of recognizer suited for this task is Hidden Markov Models (HMM). Examples of HMM-based recognizers can be found in [4]–[6] for modern scripts and in [7], [8] for historical documents. In this paper, we consider the well-known sliding window approach to HMM-based handwriting recognition that relies on a sequence of feature vectors extracted by a local window moving from left to right over the text images. Examples of this approach can be found in [5], [6], [8].

Considering the two-dimensional nature of handwriting, it is difficult to capture the inherent two-dimensional information adequately in a feature vector. For example, no context information can be exploited within a local window. In this paper, we propose a novel feature set based on graph similarity for HMM-based recognition in historical documents. In contrast to other descriptors proposed in the literature, e.g., in [5], [6], [9], this novel set of features is based on a structural representation of text images using graphs. From such a structural representation we expect that two-dimensional information is better preserved and contextual relationships can be more adequately modelled.

Previous works using graphs in handwriting recognition can be found in the context of single character recognition. For example, graph-based recognition of Chinese characters was reported in [10]. Complete text recognition based on graphs has been attempted for distorted machine printed documents [11] and online handwriting [12]. In [11], complete text lines are recognized by means of error-tolerant subgraph isomorphisms from character prototype graphs to text line graphs. In this segmentation-free approach, graphs have proven successful to handle both touching and broken characters.

In contrast to machine printed documents, handwritten text is more ambiguous and this ambiguity is difficult to resolve with error-tolerant subgraph isomorphism. Instead, we propose to extract a sequence of feature vectors from handwriting graphs and feed these vectors into an HMM for recognition.

The transformation of graphs into feature vectors is a challenging problem. A promising direction is given by spectral decomposition [13]. However, spectral methods are usually only applicable to unlabeled graphs or labeled graphs with constrained label alphabets [14]. A more versatile approach is given by dissimilarity space embedding. Originally proposed for feature vectors [15], it was recently extended to graphs [16]. In this approach, a graph is represented by a feature vector, the components of which represent the dissimilarity of the graph to a set of prototype graphs.

The proposed graph similarity features rely on the idea of first transforming the image of a handwritten text into a large graph. Then local subgraphs are extracted using a sliding window that moves from left to right over the large graph.

Each local subgraph is compared to a set of n prototype graphs (each representing a letter from the alphabet) using a well-known graph distance measure. This process results in a vector consisting of n distance values for each local subgraph. Finally, the sequence of vectors obtained for the complete text image is input to an HMM-recognizer.

In an experimental evaluation, we consider the task of HMM-based single word recognition on the medieval Parzival data set [8]. It is demonstrated that the proposed graph similarity features significantly outperform two well-known reference feature sets, namely, the geometrical features proposed by Marti & Bunke in [5] and the pixel count features proposed by Vinciarelli et al. in [6].

To the knowledge of the authors, the proposed graph similarity features are the first attempt to use graph-based representation for segmentation-free offline handwriting recognition. This can be regarded as a step towards overcoming the limitations of conventional sliding window techniques that have limitations in adequately capturing two-dimensional information.

The remainder of this paper is structured as follows. In Section II, the Parzival dataset and image preprocessing are discussed. Section III presents the graph-based representation of text images from which the graph similarity features are extracted as described in Section IV. HMM-based recognition and the reference descriptors are explained in Section V and the experimental results for single word recognition are presented in Section VI. Finally, Section VII draws some conclusions.

II. Data Set and Preprocessing

The Parzival data set [8] considered in this paper is based on a digitized medieval manuscript from the 13th century. It contains the epic poem *Parzival* by Wolfram von Eschenbach, one of the most significant epics of the European Middle Ages. The original manuscript is kept in the Abbey Library of Saint Gall, Switzerland (Cod. 857). The manuscript is written in the Middle High German language with ink on parchment. Although several writers have contributed to the manuscript, the different writing styles are very similar. From the 323 folia (sheets), transcriptions are available for 45 pages. In Figure 1, an extract from a text column is given.

For HMM-based word recognition using graph similarity features, skeleton word images are required, i.e., binary images that represent the text foreground with one pixel wide medial curves. Because this paper is focused on handwriting recognition, we assume that all text lines have already been segmented into isolated words. The skeleton images are obtained in two steps.

In the first step, normalized binary word images are generated. Text foreground is retrieved with respect to a global luminosity threshold from document images that are locally enhanced by means of a Difference of Gaussian (DoG) edge

Figure 1. Extract from Prazival Cod. 857, page 36.

detection [8]. As proposed in [5], the handwriting images are normalized in order to cope with different writing styles. The skew, i.e., the inclination of the text line, is corrected, vertical scaling is applied with respect to the upper and lower baseline, and a horizontal scaling operation is performed using the mean distance of black-white transitions in a text line. For more details, we refer to [8].

In the second step, a word skeleton image is obtained by thinning. In this paper, the 3×3 thinning operator proposed in [17] is applied. Based on two sub-iterations on a checkerboard pattern, one pixel wide medial curves are extracted while preserving connectivity. An implementation is given by Matlab's `bwmorph` function.

III. Graph Representation

Skeleton word images are represented by graphs based on keypoints that include endpoints, intersections, and corner points of circular structures. Not only keypoints, but also their connections are represented by graph nodes. This results in graphs without edges. Not using edges has the advantage that an optimal graph edit distance, which is used for the extraction of graph similarity features described in Section IV, can be calculated in polynomial instead of exponential time. At the same time, if the density of the nodes is high enough, the essential structural information is still preserved, although no edges are present.

To derive a word graph from a word skeleton, a node is added to the word graph for each skeleton keypoint and is labelled with its position $(x, y) \in \mathbb{R}^2$. Keypoints include endpoints with one direct neighbor, intersections with more than two direct neighbors, and the upper left corner point of circular structures.

After all keypoints have been included in the word graph, connection points are added. Given a skeleton connection between two keypoints in form of a pixel chain $(x_1, y_1), \ldots, (x_m, y_m)$, connection points are inserted at regular distance D along that chain. Hereby, the distance

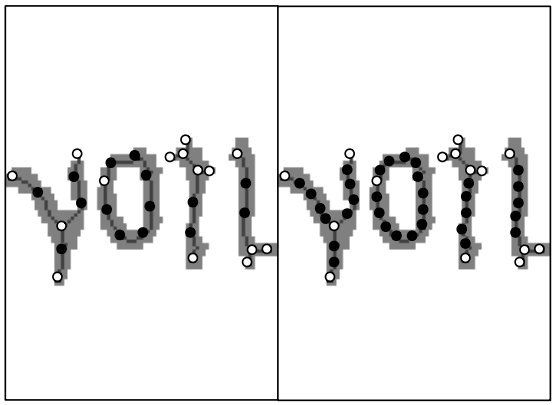

Figure 2. Handwriting graphs of the word "von" for $D = 9.0$ (left) and $D = 5.0$ (right). The text foreground is shown light gray and the skeleton dark gray. Keypoint nodes are indicated with white circles and connection point nodes with black circles.

$d(i, j)$ between two chain points (x_i, y_i) and (x_j, y_j) with $i < j$ is given by the sum

$$d(i,j) = \sum_{k=i}^{j-1} ||(x_{k+1}, y_{k+1}) - (x_k, y_k)||$$

of Euclidean distances along the chain. Depending on the connection point distance D, word graphs with different degrees of exactness or resolution are obtained, as shown in Figure 2 for the word "von".

IV. Graph Similarity Features

To extract graph similarity features for HMM-based recognition, a sliding window is moved from left to right over the word graph. At each window position, graph similarity features are computed, which capture the similarity between the subgraph currently inside the window and all character prototypes.

For each character, a prototype graph is manually extracted from template images during the training phase. At runtime, at each window position, the window subgraph is transformed into a vectorial description by calculating the graph edit distance between the window subgraph and each character prototype graph. The resulting feature vectors are normalized, using the maximum local distance, and the vector sequence that results for a complete word image graph by moving the sliding window in discrete steps from left to right is input to the HMM-recognizer. In the following subsections we describe the individual steps of this processing chain in more detail.

A. Graph Edit Distance

To calculate the dissimilarity $d(g_1, g_2)$ between two graphs g_1 and g_2, representing the subgraph inside the current position of the window and a character prototype, we use the graph edit distance [18], a well established method

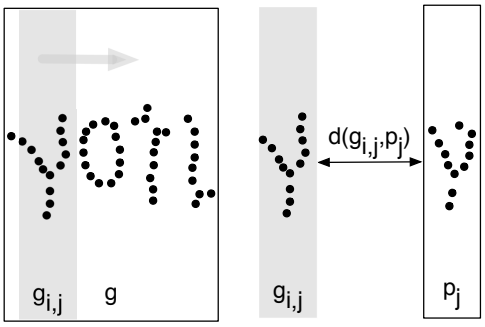

Figure 3. Embedding of the word graph "von" with respect to the character prototype graph "v" (D=5.0).

for error-tolerant graph matching. The edit distance is given by the minimum cost of edit operations needed to transform g_1 into g_2. Possible edit operations include the insertion, deletion and substitution of nodes and edges.

Because no edges are used for the proposed graph-based text representation, only edit operations on the nodes have to be considered. We use a constant cost C for node insertion as well as deletion, and the Euclidean distance between two nodes $||(x_1, y_1) - (x_2, y_2)||$ for node substitution.

Typically, the graph edit distance is calculated with the A^* algorithm [18] which performs a tree search with an exponential time complexity. In the absence of edges, however, the problem of graph edit distance is reduced to an assignment problem that can be optimally solved by the Hungarian algorithm [19] in polynomial time.

B. Transforming Graphs into Feature Vectors

In the training phase, a prototype graph is generated for each character class. The selection of the prototype character images is done manually, while the generation of the prototype graphs from the selected character images follows exactly the same steps as outlined in Sections II and III. During runtime, for each character prototype graph p_j with $1 \leq j \leq n$ a sliding window w_j with the same width as the corresponding character template image is moved over the word graph from left to right as illustrated in Figure 3 for the character "v" and the word "von". At each window center position i with $1 \leq i \leq N$, the subgraph $g_{i,j}$ that is captured by the window w_j is extracted and compared to the corresponding prototype graph p_j. This results in a sequence of feature vectors $\mathbf{x}_1, \ldots, \mathbf{x}_N$ with

$$\mathbf{x}_i = (d(g_{i,1}, p_1), \ldots, d(g_{i,n}, p_n));\ 1 \leq i \leq N$$

where $d(g_{i,j}, p_j)$ is the graph edit distance between the subgraph $g_{i,j}$ and the character prototype graph p_j.

In contrast to a fixed-size sliding window, different context widths are considered as shown in Figure 4. At each window center position i, the local context subgraphs $g_{i,1}, \ldots, g_{i,n}$ are extracted with respect to the width of the corresponding

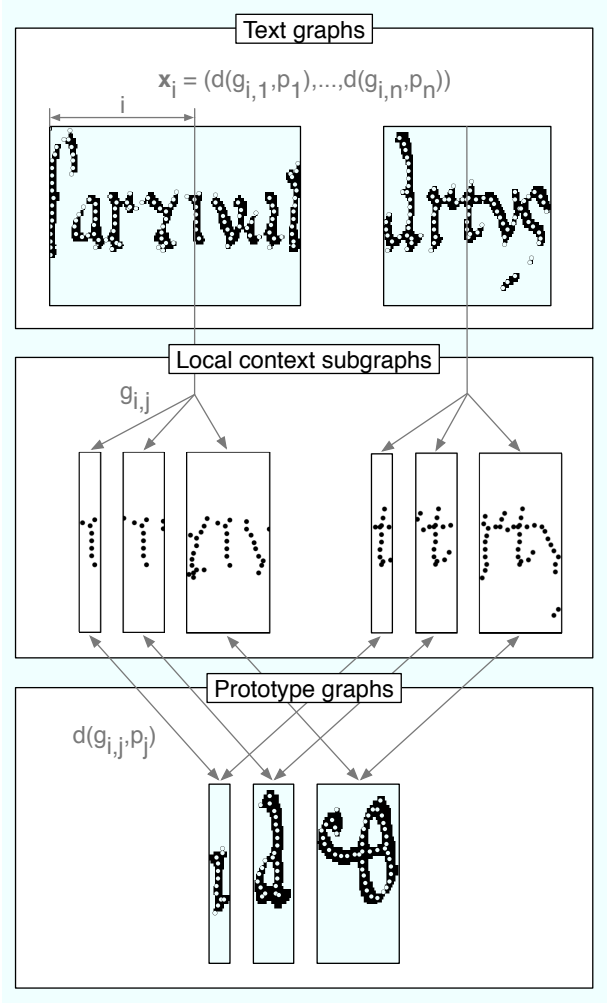

Figure 4. Text graphs, local context subgraphs, and prototype graphs for the words "Parzival" and "Artus" and the prototypes "i", "A", and "D" ($D = 5.0$).

prototypes $p_1, \ldots, p_n$. That is, the feature vector $\mathbf{x}_i = (d(g_{i,1}, p_1), \ldots, d(g_{i,n}, p_n))$ is given by the local structural dissimilarity of the text graph at position i to each prototype taking different context widths into account.

The choice of prototypes has, of course, a large influence on the resulting features. In this paper, we follow the straight-forward approach to manually select one prototype per character in order to ensure that the structural vocabulary of the data set is represented in the set of prototypes. Note, however, that the prototype graphs are not required to represent characters of the alphabet. We assume that an unsupervised prototype selection or an artificial prototype generation could also provide a representative structural vocabulary for feature extraction.

C. Normalization

The features used by the HMM-recognizer are obtained from the sequence of dissimilarity features in two normalization steps. The first normalization step puts the graph edit distance into relation with the maximum local graph edit distance that is given by the cost of deleting all nodes of the subgraph $g_{i,j}$ and inserting all nodes of the prototype graph p_j. It is given by

$$s(g_{i,j}, p_j) = 1 - \frac{d(g_{i,j}, p_j)}{C \cdot (|g_{i,j}| + |p_j|)}$$

where C is the node insertion and deletion cost and $|g|$ is the size of the graph g, i.e., the number of nodes. By subtracting this relative distance from 1, $s(g_{i,j}, p_j)$ becomes, in fact, a similarity measure that is bounded by $0 \leq s(g_{i,j}, p_j) \leq 1$ with $s(g_{i,j}, p_j) = 1$ iff $g_{i,j}$ and p_j are isomorphic.

In the second normalization step, the similarity measure $s(g_{i,j}, p_j)$ of the character prototype graph p_j is weighted with respect to all other prototypes using $\frac{s(g_{i,j},p_j)}{\sum_{k=1}^{n} s(g_{i,k},p_k)}$. This normalization results in the final graph similarity features

$$\hat{s}(g_{i,j}, p_j) = \frac{s(g_{i,j}, p_j)^2}{\sum_{k=1}^{n} s(g_{i,k}, p_k)}$$

whose sequence $\mathbf{x}_i = (\hat{s}(g_{i,1}, p_1), \ldots, \hat{s}(g_{i,n}, p_n))$ for $1 \leq i \leq N$ is input to the HMM-recognizer.

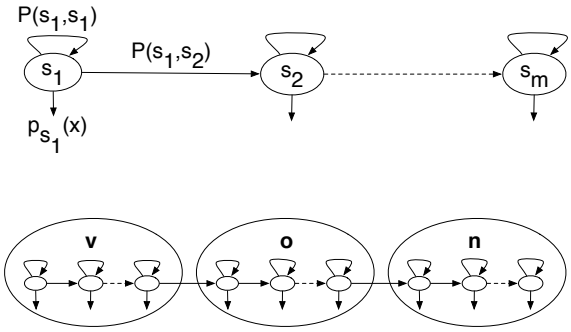

Figure 5. Hidden Markov Models.

V. HMM-Based Recognition

A. Recognition System

The HMM-based recognizer used in this paper is similar to the one presented in [5]. It is based on character models with several hidden states $s_1, \ldots, s_m$ arranged in a linear topology. An illustration of a single character HMM is given in Figure 5 (top). The states s_j with $1 \leq j \leq m$ emit observable feature vectors $\mathbf{x} \in \mathbb{R}^n$ with output probability distributions $p_{s_j}(\mathbf{x})$ given by a mixture of Gaussians. Starting from the first state s_1, the model either rests in a state or changes to the next state with transition probabilities $P(s_j, s_j)$ and $P(s_j, s_{j+1})$, respectively.

During training of the recognizer, word HMMs are built by concatenating single character HMMs as illustrated in Figure 5 (bottom) for the word "von". The probability of a word HMM to emit the observed feature vector sequence

$\mathbf{x}_1, \ldots, \mathbf{x}_N$ is then maximized by iteratively adapting the initial output probability distributions $p_{s_j}(\mathbf{x})$ and the transition probabilities $P(s_j, s_j)$ and $P(s_j, s_{j+1})$ with the Baum-Welch algorithm [20].

For single word recognition, each possible word is modeled by an HMM built from the trained character HMMs, and the most probable word is chosen with the Viterbi algorithm [20] for an unknown input sequence $\mathbf{x}_1, \ldots, \mathbf{x}_N$. Because we focus on feature evaluation, a closed vocabulary is used without a language model, i.e., all word classes are treated with equal a priori probability. For Baum-Welch training and Viterbi decoding, the HTK[1] implementation is used.

B. Reference Systems

For HMM-based single word recognition, the proposed graph similarity features are compared with two feature sets known from the literature. For both reference descriptors, a feature vector sequence is extracted with a sliding window from non-thinned, normalized binary word images.

The first reference system is the column descriptor proposed by Marti & Bunke in [5]. Using a sliding window width of one pixel, nine geometric features are extracted. Three global features capture the fraction of black pixels, the center of gravity, and the second order moment. The remaining six local features consist of the position of the upper and lower contour, the gradient of the contours, the number of black-white transitions, and the fraction of black pixels between the contours.

The second reference system is the pixel count descriptor proposed by Vinciarelli et al. in [6]. A sliding window with a width of 16 pixels is used that is adjusted to the area actually containing foreground pixels at each window position. The 16-dimensional descriptor is then given by the fraction of foreground pixels in 4×4 regular window cells.

VI. Experimental Results

For experimental evaluation, HMM-based single word recognition is performed on the Parzival data set. The proposed graph similarity features are compared with the two reference descriptors of Section V-B. 11,743 word images are considered that contain 3,177 word classes and 87 characters including special characters that occur only once or twice in the data set.

A. Setup

First, the word images are divided into three distinct sets for training, validation, and testing. Half of the words, i.e., each other word, is used for training and a quarter of the words for validation and testing, respectively. For each of the 74 characters present in the training set, a prototype graph is extracted from a manually selected template image. For five characters, two prototypes are chosen because two completely different writing styles were observed, resulting in a set of 79 prototypes. Consequently, the graph similarity features have a dimension of 79.

[1] http://htk.eng.cam.ac.uk/

Table I
Word accuracy on the test set. All improvements are statistically significant (t-test, $\alpha = 0.05$).

Descriptor	Acc.	Parameters
Marti & Bunke	88.69	G=7
Vinciarelli	90.49	G=5
Graph Similarity	94.00	D=3.0,C=3.0,G=29

Parameters that are optimized with respect to the validation accuracy include the connection point distance $D \in \{3.0, 5.0, 7.0, 9.0\}$ for the graph-based representation, the node insertion and deletion cost $C \in \{0.4D, 0.6D, \ldots, 1.4D\}$ for graph similarity feature extraction, and the number of Gaussian mixtures $G \in \{1, 2, \ldots, 30\}$ for HMM-based recognition. An optimized number of HMM states for the Parzival characters is adopted from previous work [8].

B. Results and Discussion

The word recognition accuracy on the test set and the optimal parameter values are given in Table I for the proposed graph similarity features and both reference descriptors. The reference systems are significantly outperformed on the Parzival data set.

The parameters D and C of the graph similarity features had a significant impact on the validation accuracy. The optimal choice was given by the most dense connection point distance of $D = 3.0$ and the same node insertion and deletion cost $C = 1.0D = 3.0$. We interpret the high optimal number of Gaussian mixtures $G = 29$ found for graph similarity features as an indicator for a good overall feature quality, because the HMMs could be closely adapted to the training set without suffering from overfitting.

The feature normalization discussed in Section IV-C has proven to be very important. A test accuracy of only 84.19 is reported for the plain graph edit distance features. For the unnormalized similarity features, a test accuracy of 90.60 is reported. Both results are significantly below the result achieved with the normalized graph similarity features.

Although the accuracy of the reference descriptors is significantly worse, their effectiveness is still appealing in terms of computational speed. The graph similarity features rely on a sliding window for each character prototype and are extracted in $O((n+m)^3)$ time with respect to the number of subgraph nodes n and the number of prototype graph nodes m. The reference descriptors, on the other hand, are extracted by a single sliding window in only $O(w \cdot h)$ time with respect to the window width w and height h.

VII. Conclusions

In this paper, graph similarity features are proposed as a novel descriptor for HMM-based handwriting recognition in historical documents. They are extracted from a graph-based representation of text images. For HMM-based recognition, the handwriting graphs are transformed into a sequence of feature vectors by means of the graph edit distance between the window subgraph and character prototype graphs.

From the chosen structural representation one expects that the two-dimensional nature of handwriting is better preserved and contextual relationships can be more adequately modelled. To the knowledge of the authors, the graph similarity features proposed in this paper are the first attempt to use graph-based representation for segmentation-free offline handwriting recognition.

In an experimental evaluation, the task of HMM-based single word recognition on the medieval Parzival data set is considered. The proposed descriptor has significantly outperformed two well-known reference descriptors. Although having a lower recognition accuracy, the effectiveness of the reference systems with respect to computational speed is still appealing.

In future research, other graph representations of handwriting can be investigated, for example, taking edges and more complex labels into account. Furthermore, feature extraction can be improved by prototype selection, feature selection and dimensionality reduction. Finally, Lipschitz embedding could be valuable to improve the robustness for different writing styles [16].

Acknowledgements

This work has been supported by the Swiss National Science Foundation (Project CRSI22_125220).

References

[1] A. Antonacopoulos and A. Downton (eds.), "Special issue on the analysis of historical documents," *Int. Journal on Document Analysis and Recognition*, vol. 9, no. 2–4, pp. 75–77, 2007.

[2] G. Nagy and D. Lopresti, "Interactive document processing and digital libraries," in *Proc. 2nd Int. Workshop on Document Image Analysis for Libraries (DIAL 2006)*. IEEE Computer Society, 2006, pp. 2–11.

[3] R. Plamondon and S. Srihari, "Online and off-line handwriting recognition: A comprehensive survey," *IEEE Trans. PAMI*, vol. 22, pp. 63–84, 2000.

[4] A. El Yacoubi, M. Gilloux, R. Sabourin, and C. Suen, "An HMM-based approach for off-line unconstrained handwritten word modeling and recognition," *IEEE Trans. PAMI*, vol. 21, no. 8, pp. 752–760, 1999.

[5] U.-V. Marti and H. Bunke, "Using a statistical language model to improve the performance of an HMM-based cursive handwriting recognition system," *Int. Journal of Pattern Rec. and Art. Intelligence*, vol. 15, pp. 65–90, 2001.

[6] A. Vinciarelli, S. Bengio, and H. Bunke, "Offline recognition of unconstrained handwritten texts using HMMs and statistical language models," *IEEE Trans. PAMI*, 2004.

[7] J. Edwards, Y. W. Teh, D. Forsyth, R. Bock, M. Maire, and G. Vesom, "Making latin manuscripts searchable using gHMM's," in *Advances in Neural Information Processing Systems 17*, L. Saul, Y. Weiss, and L. Bottou, Eds. MIT Press, 2005, pp. 385–392.

[8] A. Fischer, M. Wüthrich, M. Liwicki, V. Frinken, H. Bunke, G. Viehhauser, and M. Stolz, "Automatic transcription of handwritten medieval documents," in *Proc. 15th Int. Conf. on Virtual Systems and Multimedia (VSMM)*, vol. 1. IEEE, September 2009, pp. 137–142.

[9] J. Rodriguez and F. Perronnin, "Local gradient histogram features for word spotting in unconstrained handwritten documents," in *Proc. IEEE Int. Conf. on Frontiers in Handwriting Recognition (ICFHR 2008)*, 2008.

[10] S. Lu, Y. Ren, and C. Suen, "Hierarchical attributed graph representation and recognition of handwritten chinese characters," *Pattern Recognition*, vol. 24, no. 7, pp. 617–632, 1991.

[11] J. Rocha and T. Pavlidis, "Character recognition without segmentation," *IEEE Trans. PAMI*, vol. 17, no. 9, pp. 903–909, 1995.

[12] V. Chakravarthy and B. Kompella, "The shape of handwritten characters," *Pattern Recognition Letters*, vol. 24, no. 12, pp. 1901–1913, 2003.

[13] R. Wilson, E. Hancock, and B. Luo, "Pattern vectors from algebraic graph theory," *IEEE Trans. PAMI*, vol. 27, no. 7, pp. 1112–1124, 2005.

[14] F. Chung, *Spectral Graph Theory*. American Mathematical Society, 1997.

[15] E. Pekalska and R. Duin, *The Dissimilarity Representations for Pattern Recognition: Foundations and Applications*. World Scientific, 2005.

[16] K. Riesen and H. Bunke, *Graph Classification and Clustering Based on Vector Space Embedding*, ser. Series in Machine Perception and Artificial Intelligence. World Scientific, 2010, vol. 77.

[17] Z. Guo and R. Hall, "Parallel thinning with two-subiteration algorithms," *Communications of the ACM*, vol. 32, no. 3, pp. 359–373, 1989.

[18] H. Bunke and G. Allermann, "Inexact graph matching for structural pattern recognition," *Pattern Recognition Letters*, vol. 1, no. 4, pp. 245–253, 1983.

[19] J. Munkres, "Algorithms for the assignment and transportation problems," *Journal of the Society for Industrial and Applied Mathematics*, vol. 5, no. 1, pp. 32–38, 1957.

[20] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–285, Feb. 1989.