Graph-Based Keyword Spotting in Historical Documents Using Context-Aware Hausdorff Edit Distance

Michael Stauffer*§, Andreas Fischer†‡ and Kaspar Riesen*
*University of Applied Sciences and Arts Northwestern Switzerland, Institute for Information Systems,
4600 Olten, Switzerland, Email: {michael.stauffer,kaspar.riesen}@fhnw.ch
†University of Fribourg, Department of Informatics,
1700 Fribourg, Switzerland, Email: andreas.fischer@unifr.ch
‡University of Applied Sciences and Arts Western Switzerland, Institute for Complex Systems,
1705 Fribourg, Switzerland
§University of Pretoria, Department of Informatics, 0083 Pretoria, South Africa

***Abstract*—Scanned handwritten historical documents are often not well accessible due to the limited feasibility of automatic full transcriptions. Thus, Keyword Spotting (KWS) has been proposed as an alternative to retrieve arbitrary query words from this kind of documents. In the present paper, word images are represented by means of graphs. That is, a graph is used to represent the inherent topological characteristics of handwriting. The actual keyword spotting is then based on matching a query graph with all document graphs. In particular, we make use of a fast graph matching algorithm that considers the contextual substructure of nodes. The motivation for this inclusion of node context is to increase the overall KWS accuracy. In an experimental evaluation on four historical documents, we show that the proposed procedure clearly outperforms diverse other template-based reference systems. Moreover, our novel framework keeps up or even outperforms many state-of-the-art learning-based KWS approaches.**

***Keywords*-Handwritten Keyword Spotting; Graph Representation; Hausdorff Edit Distance; Ensemble Methods**

I. Introduction

The accessibility of handwritten historical documents is often limited, especially as an automatic and full transcription is often negatively affected by high writing variations and degraded documents. Hence, *Keyword Spotting (KWS)* has been proposed as an alternative to a full transcription [1]. KWS allows us to retrieve arbitrary keywords in handwritten historical documents.

KWS approaches are either based on *template-based* or *learning-based* algorithms. In the former case, a query image is directly matched against a set of document images. To this end word images are often represented by sequences of features vectors and then matched by means of *Dynamic Time Warping (DTW)* [2], [3], [4], [5]. In the latter case, the extracted features are used to train a statistical model like for example *Hidden Markov Models* [2], [4], [5], *Support Vector Machine (SVM)* [6], or *Convolutional Neural Networks (CNN)* [7]. Learning-based approaches often achieve higher accuracies than template-based approaches. However, this advantage is accompanied by a loss of flexibility which is due to the *a priori* learning and the need for labelled training data.

The present template-based KWS approach makes use of graphs — rather than sequences of feature vectors — for the representation of handwritten words. Graphs actually offer a natural way to represent the inherent topological characteristics of handwriting. Moreover, several fast matching algorithms applicable to large and complex graphs have been proposed in recent years (e.g. [8], [9]). Both observations make graphs to a valid alternative to vectorial representations.

In previous graph-based KWS approaches (e.g. [10], [11]), the employed matching algorithms rely on finding correspondences of local substructures (i.e. nodes and their adjacent edges). To make the matching of local structures more precise, the current paper employs a novel graph matching algorithm [12] that considers not only adjacent edges but a more comprehensive context of nodes, viz. hierarchical subgraphs. The second major contribution of this paper is that we combine several context levels and graph representations by means of *ensemble methods* [13] to further improve the KWS accuracy.

The remainder of this paper is organised as follows. In Section II, the graph representation for handwriting is introduced. The actual graph-based KWS system is defined in Section III. An experimental evaluation and comparison with template- and learning-based system is given in Section IV. Finally, Section V concludes the paper and outlines possible future research activities.

II. Graph-based Representation of Handwriting

In Fig. 1, the proposed graph-based KWS framework is illustrated. First, scanned document images are enhanced to reduce the influence of noisy background. In the same step, handwritten document images are binarised and segmented into single word images[1]. Optionally, word images are skeletonised by means of a thinning operator. For details with respect to the image preprocessing we refer to [14].

On the basis of preprocessed word images, graphs are extracted by means of two different representation formalisms. The actual keyword spotting is then based on matching graphs (see Sect. III).

Generally, a graph g is defined as a four-tuple $g = (V, E, \mu, \nu)$ where V and E are finite sets of nodes and

[1]The present framework focuses on KWS that operates on perfectly segmented word images. Thus, the achieved end-to-end performance can be seen as an upper-bound.

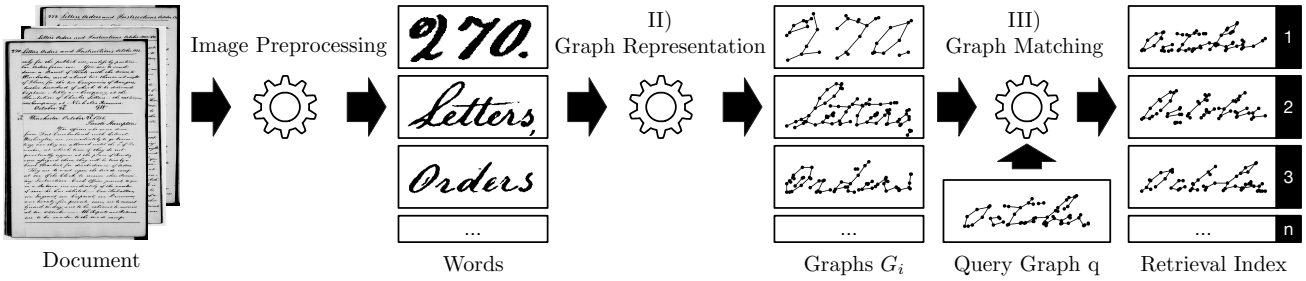

Figure 1: Process of Graph-based Keyword Spotting of the Word "October"

edges, and $\mu : V \rightarrow L_V$ as well as $\nu : E \rightarrow L_E$ are labelling functions for nodes and edges, respectively. Graphs can be divided into *undirected* and *directed* graphs, where pairs of nodes are either connected by undirected or directed edges. Additionally, graphs are often distinguished into *unlabelled* and *labelled* graphs. In the latter case, both nodes and edges can be labelled with numerical, vectorial, or symbolic labels from L_V or L_E, respectively. In the former case we assume empty label alphabets, i.e. $L_V = L_E = \{\}$.

On the basis of binarised and/or skeletonised word images, graphs are derived by means of two different representation formalisms as originally proposed in [14]. Both graph extraction methods result in undirected graphs with labelled nodes and unlabelled edges, i.e. $L_V = \mathbb{R}^2$ and $L_E = \{\}$.

- `Keypoint (K)`: The first graph extraction algorithm makes use of characteristics points (so called keypoints) in skeletonised word images. These keypoints are represented as nodes that are labelled with the corresponding (x, y)-coordinates. Between pairs of keypoints (which are connected on the skeleton) further intermediate points are converted to nodes and added to the graph at equidistant intervals. Finally, undirected edges are inserted into the graph for each pair of nodes that is directly connected by a stroke.
- `Projection (P)`: The second graph extraction algorithm works on an adaptive and threshold-based segmentations of binarised word images. Basically, this segmentation is computed on horizontal and vertical projection profiles. The resulting segmentation is further refined in the horizontal and vertical direction by means of two distance-based thresholds. A node is inserted into the graph for each segment and labelled by the (x, y)-coordinates of the corresponding centre of mass. Undirected edges are inserted into the graph for each pair of nodes that is directly connected by a stroke in the original word image.

For both graph types, the (x, y)-coordinates of the node labels $\mu(v)$ are normalised by a z-score. Formally,

$$\hat{x} = \frac{x - \mu_x}{\sigma_x} \text{ and } \hat{y} = \frac{y - \mu_y}{\sigma_y} \quad ,$$

where (μ_x, μ_y) and (σ_x, σ_y) represent the mean and standard deviation of all (x, y)-coordinates in the graph under consideration.

III. Graph Matching

The proposed keyword spotting system is based on the pairwise matchings of a query graph q (used to represent a given keyword) with all available document graphs $G = \{g_1, \ldots, g_N\}$. In general, graphs can either be matched by exact or inexact graph matching algorithms [15]. However, in the current case graphs are used to represent the topological characteristics of handwriting, and thus, affected by small variations in both their structure and labels. As a consequence, inexact graph matching is feasible only.

A. Graph Edit Distance

In the last decades, several approaches have been proposed for inexact graph matching [15]. *Graph Edit Distance (GED)* is regarded as one of the most flexible and powerful paradigms available [16]. GED measures the minimum amount of distortion needed to transform graph g_1 into graph g_2 using a sequence of edit operations like *insertions*, *deletions*, and *substitutions* of both nodes and edges. Such a sequence $(e_1, \ldots, e_t)$ is called an *edit path* $\lambda(g_1, g_2)$ between g_1 and g_2. Formally, the graph edit distance $d_{\text{GED}}(g_1, g_2)$, or d_{GED} for short, between g_1 and g_2 is defined by

$$d_{\text{GED}}(g_1, g_2) = \min_{\lambda \in \Upsilon(g_1, g_2)} \sum_{e_i \in \lambda} c(e_i) \quad ,$$

where $\Upsilon(g_1, g_2)$ is the set of all valid edit paths between g_1 and g_2.

The graph edit distance crucially depends on the definition of a specific cost function $c(e)$ for every node and edge edit operation e. This cost function is generally used to embed certain domain-specific knowledge to the matching procedure. Thus, the cost function should correspond to the strength of a certain modification of the graph. The current cost model is based on a constant cost for both node and edge deletions/insertions, i.e. $\tau_v \in \mathbb{R}^+$ and $\tau_e \in \mathbb{R}^+$, while the cost for node substitutions is given by a weighted Euclidean distance between the corresponding node labels, i.e. (x, y)-coordinates. Formally, the cost for substituting a node v_i with node v_j with $\mu(v_i) = (x_i, y_i)$ and $\mu(v_j) = (x_j, y_j)$ is given by

$$\sqrt{\alpha\, \sigma_x (x_i - x_j)^2 + (1 - \alpha)\, \sigma_y (y_i - y_j)^2} \quad ,$$

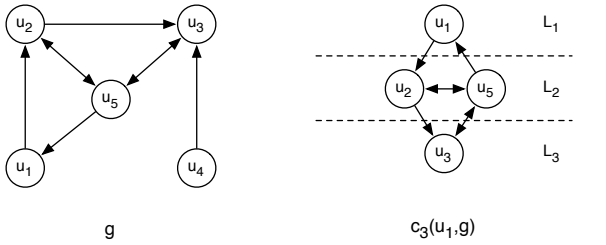

Figure 2: Structural context of node u_1 with degree $n = 3$.

where $\alpha \in [0, 1]$ denotes a parameter to weight the importance of the x- and y-coordinate of a node, while σ_x and σ_y denote the standard deviation of all node coordinates in the current query graph (in x- and y-direction, respectively). Finally, a weighting factor $\beta \in [0, 1]$ is used in our model to balance the sum of all node and edge edit costs in a given edit path by β and $(\beta - 1)$, respectively.

B. Context-Aware Hausdorff Edit Distance

The number of possible edit paths between two graphs — and therefore also the time complexity of the computation of d_{GED} — is exponential with respect to the number of nodes of the involved graphs. Actually, the computation of GED is an instance of the *Quadratic Assignment Problem (QAP)* [17], which in turn belongs to the class of $\mathcal{NP}$-complete problems[2]. To reduce this high computational complexity, several fast but suboptimal algorithms for GED have been proposed in the last decade (see [15] for an exhaustive review).

In this paper, we consider the Hausdorff Edit Distance (HED) [9] for graph matching. It reduces the problem of GED to a set matching problem between local substructures, similar to the Hausdorff distance between finite sets. With respect to substructures that consist of nodes and their adjacent edges, a lower bound of GED is computed in quadratic time. Formally, $d_{\text{HED}}(g_1, g_2)$ is defined as

$$\sum_{u \in V_1} \min_{v \in V_2 \cup \{\epsilon\}} h(u, v) + \sum_{v \in V_2} \min_{u \in V_1 \cup \{\epsilon\}} h(u, v) \ ,$$

where $h(u, v)$ is the assignment cost between the local substructures around the nodes $u \in V_1$ and $v \in V_2$. The terms $h(u, \epsilon)$ and $h(\epsilon, v)$ are the costs for deleting and inserting a substructure, respectively.

The Context-Aware Hausdorff Edit Distance (CED) [12] extends the local substructures beyond the adjacent edges, in order to better integrate the global context into the set matching process, thus reducing the approximation error with respect to the exact GED.[3]

The structural node context of node u with degree n is defined as

$$c_n(u, g) = (L_1, \ldots, L_n) \ ,$$

with respect to the subgraphs L_i containing all nodes that have the shortest path distance of $(i - 1)$ from u. Figure 2 illustrates the structural node context with an example.

Next, the distance between two structural node contexts is defined as

$$d(c_n(u, g_1), c_n(v, g_2)) = \sum_{i=1}^{n} d_{HED}(L_i, K_i) \ ,$$

and the graph distance d_{CED} is computed as

$$\sum_{u \in V_1} \min_{v \in \{b_n(u, g_2), \epsilon\}} h(u, v) + \sum_{v \in V_2} \min_{u \in \{b_n(v, g_1), \epsilon\}} h(u, v) \ ,$$

where $b_n(u, g_2)$ is the node in g_2 with the minimum distance from u with respect to the structural node context. Similarly, $b_n(v, g_1)$ is the nearest neighbour of v in g_1. For more details on CED, we refer the reader to [12].

Finally, the distance d_{CED} between query graph q and a document graph $g \in G$ is normalised by the sum of the maximum cost edit path between $q = (V_q, E_q, \mu_q, \nu_q)$ and $g = (V_g, E_g, \mu_g, \nu_g)$, i.e. the sum of the edit path costs that results from deleting all nodes and edges of q and inserting all nodes and edges in g. Formally,

$$\hat{d}_{\text{CED}}(q, g) = \frac{d_{\text{CED}}(q, g)}{(|V_q| + |V_g|)\, \tau_v + (|E_q| + |E_g|)\, \tau_e} \quad ,$$

where τ_v and τ_e denote the node and edge insertion/deletion costs. In case a query consists of a set of graphs $\{q_1, \ldots, q_t\}$ that represents the same keyword, the normalised graph edit distance $\hat{d}_{\text{CED}}$ is actually given by the minimal distance achieved on all t query graphs, i.e. $\min_{q_i \in \{q_1, \ldots, q_t\}} \hat{d}_{\text{CED}}(q_i, g)$.

C. Ensemble Method

In order to combine the graph edit distances of different context levels and/or graph representations, we make use of the mean-ensemble method [13] in three different types of settings (resulting in six ensemble methods in total).

- mean-Graph(n): The first type of ensemble methods averages both graph edit distances derived on `Keypoint` and `Projection` graphs. We repeat this combination for three context radii $n \in \{1, 3, 5\}$.
- mean-Context(m): The second type of ensemble methods averages all graph edit distances derived with the three different context radii. This combination is individually carried out for `Keypoint` and `Projection` graphs (i.e. $m \in \{\texttt{K}, \texttt{P}\}$).
- mean-All: Finally, the last ensemble method averages the graph edit distances of both representations and all three context radii.

D. Retrieval Method

Finally, the graph edit distances (or the mean of graph edit distances in case of ensemble methods) between the query graph q and all document graphs $G = \{g_1, \ldots, g_N\}$ are used to form a retrieval index for KWS. Formally,

$$r(q, g) = -\hat{d}_{\text{CED}}(q, g) \quad .$$

[2] That is, an exact and efficient algorithm for the graph edit distance problem can not be developed unless $\mathcal{P} = \mathcal{NP}$.

[3] The lower bound property, however, is lost.

IV. Experimental Evaluation

A. Datasets

The experimental evaluation is based on two well-known manuscripts, viz. *George Washington (GW)*[4] and *Parzival (PAR)*[5], as well as two documents of a recent KWS benchmark competition[6], viz. *Alvermann Konzilsprotokolle (AK)* and *Botany (BOT)*. GW is written in English and based on twenty pages with minor variations in writing and degradation. PAR is written in Middle High German and based on 45 pages with low writing variations but markable signs of degradation. AK is written in German and based on 18,000 pages with minor variations and signs of degradation. Finally, BOT is written in English and based on ten pages with high writing variation and markable signs of degradation.

In Fig. 3, an exemplary word image of each document as well as the corresponding graph representation for `Keypoint` and `Projection` is given.

B. Reference Systems

We consider three types of reference systems, viz. two graph-based systems, four template-based approaches, and three learning-based systems.

1) Graph-based Reference Systems: The first graph-based reference system [10] makes use of the bipartite GED (denoted by BP) [8]. BP reduces the QAP of GED to a *Linear Sum Assignment Problem (LSAP)*, that can be optimally solved in cubic time. This assignment takes the local node structure into consideration only, and thus, the derived GED constitutes an upper bound of the actual GED. The second graph-based reference system [11] is based on the *Hausdorff Edit Distance (HED)* [9], which is a lower bound of the GED and can be solved in quadratic time complexity. In particular, HED reduces GED to a set matching problem between local substructures (nodes and their adjacent edges)[7].

[4]George Washington Papers at the Library of Congress, 1741-1799: Series 2, Letterbook 1, pp. 270-279 & 300-309, http://memory.loc.gov/ammem/gwhtml/gwseries2.html

[5]Parzival at IAM historical document database, http://www.fki.inf.unibe.ch/databases/iam-historical-document-database/parzival-database

[6]Alvermann Konzilsprotokolle and Botany at ICFHR2016 benchmark database, http://www.prhlt.upv.es/contests/icfhr2016-kws/data.html

[7]CED is an extension of HED, that takes not only the adjacent edges of a node but a certain context into consideration (see Section III-B).

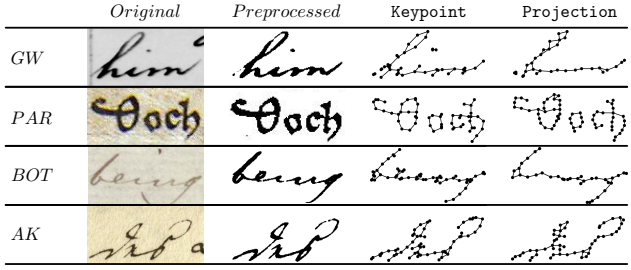

Figure 3: Exemplary graph representations of the Alvermann Konzilsprotokolle (AK), Botany (BOT), George Washington (GW), and Parzival (PAR) dataset.

2) Template-based Reference Systems: The employed template-based reference systems make use of a DTW matching approach applied on sequences of feature vectors. The features are acquired by means of a sliding window approach. The different systems mainly differ in the features extracted from word images. That is, DTW'01 [18] employs geometrical features, while DTW'08 [2] and DTW'09 [3] both rely on Histogram of Oriented Gradient features. Finally, DTW'16 [5] uses Deep Learning features.

3) Learning-based Reference Systems: The first learning-based reference system CVCDAG is based on *Pyramidal Histogram Of Characters labels (PHOC)* features used in conjunction with an SVM [6]. For the second system, termed PRG, the same features are used to train a CNN, the so called PHOCNet [7]. Finally, another CNN is used in the third reference system QTOB by means of a triplet network approach.

Note that the results of the template-based reference systems are available for GW and PAR only, while the results of all learning-based systems are available for AK and BOT only.

C. Experimental Setup

For parameter optimisation, ten different keywords (with different word lengths) are selected on all four datasets. These keywords are manually retrieved on independent validation sets that consist of 10 random instances per keyword instance and 900 additional random words (for each dataset). The optimised systems are eventually evaluated on the same training and test sets as used in [4] and [19] for GW/PAR and AK/BOT, respectively. All templates of a keyword present in the training set are used for KWS. In Table I a summary of the datasets is given.

Table I: The number of keywords as well as the size of the training and test sets for all four documents.

Dataset	Keywords	Train	Test
GW	105	2,447	1,224
PAR	1,217	11,468	6,869
BOT	150	1,684	3,380
AK	200	1,849	3,734

The performance of all KWS systems is measured by the *Mean Average Precision (MAP)*, which is the mean area under all recall-precision curves of all individual keywords.

For the basic cost model for graph matching, we evaluate 25 pairs of constants for node and edge deletion/insertion costs ($\tau_v/\tau_e \in \{1, 4, 8, 16, 32\}$) in combination with the weighting parameters $\alpha/\beta \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$ (see Section III-A). In Table II the optimal cost function parameters for three different context levels (i.e. $n = \{1, 3, 5\}$) are given for both graph representations and all datasets.

D. Results and Discussion

Our novel KWS framework is first compared with the two previous graph-based reference systems (i.e. BP and HED) as shown in Table III. On all four datasets we observe a clear improvement of the KWS accuracy using CED rather than HED or BP. In case of GW and PAR, a

Table II: Optimal cost function parameters and scaling factors for graph edit distance computation.

		τ_v			τ_e			α			β		
	n	1	3	5	1	3	5	1	3	5	1	3	5
GW	Keypoint	8	8	16	4	4	8	0.1	0.1	0.1	0.5	0.5	0.7
	Projection	8	8	8	1	1	4	0.1	0.1	0.1	0.3	0.3	0.5
PAR	Keypoint	8	4	4	4	1	1	0.5	0.5	0.3	0.3	0.1	0.1
	Projection	4	4	8	4	4	4	0.5	0.3	0.3	0.5	0.5	0.5
BOT	Keypoint	4	8	32	16	4	8	0.1	0.1	0.1	0.5	0.1	0.3
	Projection	16	32	8	32	4	16	0.1	0.1	0.1	0.5	0.1	0.3
AK	Keypoint	4	32	8	16	16	4	0.3	0.3	0.3	0.3	0.7	0.1
	Projection	32	32	8	4	16	32	0.1	0.1	0.1	0.1	0.5	0.5

context radius of $n = 1$ is ideal, while in case of BOT and AK a context radius of $n = 5$ leads to the best results.

Next, we compare the single graph-based approaches (using the three context levels) as well as the six ensemble methods with four template-based approaches on the GW and PAR dataset as shown in Table IV.

Generally, an accuracy improvement can be observed for all CED approaches when compared with the DTW approaches. That is, for context level $n = 1$ our novel graph-based framework outperforms all DTW systems on both datasets. The ensemble methods further improve the KWS accuracies in our framework. Especially the ensemble mean-Graph ($n = 1$) and mean-All clearly excel all other template-based reference systems.

Finally, the novel graph-based KWS systems are compared with three state-of-the-art learning-based reference systems. In case of single graph-based approaches, we observe a clear advantage of learning-based systems on BOT, while the proposed graph-based methods can keep up on AK. In case of ensemble methods, a substantial increase of the performance can be observed on both BOT and AK. Overall, we can observe that graph-based ensemble methods can keep up (or even outperform) two out of three learning-based reference systems. This is quite astonishing as the proposed methods — in contrast to the learning-based approaches — are not trained on labelled data.

V. Conclusion

In the present paper graphs are used for the representation of handwritten historical documents in the context of keyword spotting. That is, the inherent topological characteristics of a handwritten word are represented by means of a graph. Representing words by means of graphs makes graph matching necessary. However, the time complexity of general graph matching procedures, such as graph edit distance, is typically exponential with respect to the number of nodes. Thus, several fast but suboptimal approaches have been proposed in recent years. These suboptimal graph matchings often neglect the global structure of graphs during the matching procedure and rely on local information only.

In this paper we make use of a recent matching algorithm that considers a larger node context than previous methods. This leads to a better approximation of the resulting graph distance, and thus to a higher accuracy of the keyword spotting system. In particular, the proposed approach outperforms several state-of-the-art template-based methods and most of the learning-based frameworks. This underlines the high potential of structural approaches. Especially as learning-based approaches are depending on the acquisition of large sets of labelled training data, which is a labour-intensive and costly procedure in case of handwritten historical documents.

In future work we focus on two lines of research. We aim at combining graph-based approaches with statistical approaches (applied on vector space embedded graphs). Second, it would be a rewarding avenue to verify whether graphs can be used in the context of learning-based KWS methods.

Acknowledgment

This work has been supported by the Hasler Foundation Switzerland.

References

[1] R. Manmatha, Chengfeng Han, and E. Riseman, "Word spotting: a new approach to indexing handwriting," in *Computer Vision and Pattern Recognition*, 1996, pp. 631–637.

[2] J. A. Rodríguez-Serrano and F. Perronnin, "Local gradient histogram features for word spotting in unconstrained handwritten documents," in *International Conference on Frontiers in Handwriting Recognition*, 2008, pp. 7–12.

[3] K. Terasawa and Y. Tanaka, "Slit Style HOG Feature for Document Image Word Spotting," in *International Conference on Document Analysis and Recognition*, 2009, pp. 116–120.

[4] A. Fischer, A. Keller, V. Frinken, and H. Bunke, "Lexicon-free handwritten word spotting using character HMMs," *Pattern Recognition Letters*, vol. 33, no. 7, pp. 934–942, 2012.

[5] B. Wicht, A. Fischer, and J. Hennebert, "Deep Learning Features for Handwritten Keyword Spotting," in *International Conference on Pattern Recognition*, 2016.

[6] J. Almazan, A. Gordo, A. Fornes, and E. Valveny, "Word Spotting and Recognition with Embedded Attributes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 12, pp. 2552–2566, 2014.

[7] S. Sudholt and G. A. Fink, "PHOCNet: A Deep Convolutional Neural Network for Word Spotting in Handwritten Documents," in *International Conference on Frontiers in Handwriting Recognition*, 2016, pp. 277–282.

[8] K. Riesen and H. Bunke, "Approximate graph edit distance computation by means of bipartite graph matching," *Image and Vision Computing*, vol. 27, no. 7, pp. 950–959, 2009.

[9] A. Fischer, C. Y. Suen, V. Frinken, K. Riesen, and H. Bunke, "Approximation of graph edit distance based on Hausdorff matching," *Pattern Recognition*, vol. 48, no. 2, pp. 331–343, 2015.

[10] M. Stauffer, A. Fischer, and K. Riesen, "Graph-based Keyword Spotting in Historical Handwritten Documents," in *International Workshop on Structural, Syntactic, and Statistical Pattern Recognition*, 2016.

Table III: Mean average precision (MAP) for graph-based KWS using the bipartite graph matching (BP), the Hausdorff edit distance (HED), as well as HED with context radius (CED). With $\pm$ we indicate the relative percental gain or loss in the accuracy when compared with BP. The overall best results per dataset is shown in bold face.

		GW		PAR		BOT		AK	
Method		MAP	$\pm$	MAP	$\pm$	MAP	$\pm$	MAP	$\pm$
BP	`Keypoint`	66.08		62.04		45.06		77.24	
	`Projection`	61.43		66.23		49.57		76.02	
HED	`Keypoint`	69.28	+3.19	69.23	+7.19	51.74	+6.68	79.72	+2.48
	`Projection`	66.71	+5.28	72.82	+6.59	51.69	+2.12	81.06	+5.04
CED ($n = 1$)	`Keypoint`	**71.23**	+5.15	68.39	+6.35	51.59	+6.53	80.73	+3.49
	`Projection`	67.04	+5.62	**74.10**	+7.87	51.98	+2.41	80.28	+4.26
CED ($n = 3$)	`Keypoint`	68.46	+2.37	68.64	+6.60	52.18	+7.12	81.43	+4.19
	`Projection`	64.51	+3.08	73.99	+7.76	52.02	+2.45	82.42	+6.40
CED ($n = 5$)	`Keypoint`	65.23	−0.85	67.19	+5.15	**52.45**	+7.39	**82.69**	+5.45
	`Projection`	60.25	−1.18	72.45	+6.22	50.59	+1.02	81.58	+5.56

Table IV: Mean average precision (MAP) for graph-based KWS systems in comparison with four template-based reference systems on the George Washington (GW) and Parzival (PAR) dataset. The first, second, and third best systems are indicated by (1), (2), and (3), respectively.

Method	GW		PAR		Average	
Reference (Template)						
- DTW'01	45.26		46.78		46.02	
- DTW'08	63.39		47.52		55.46	
- DTW'09	64.80		73.49		69.15	
- DTW'16	68.64		72.38		70.51	
Graph (Single)						
- CED ($n = 1$)	71.23		74.10		72.67	
- CED ($n = 3$)	68.46		73.99		71.22	
- CED ($n = 5$)	65.23		73.60		69.41	
Graph (Ensemble)						
- mean-Graph ($n = 1$)	74.62	(1)	80.06	(2)	77.34	(1)
- mean-Graph ($n = 3$)	73.22	(3)	79.79	(3)	76.50	(3)
- mean-Graph ($n = 5$)	69.67		79.11		74.39	
- mean-Context ($m =$ `K`)	74.27	(2)	77.36		75.82	
- mean-Context ($m =$ `P`)	67.35		78.66		73.00	
- mean-All	73.06		81.08	(1)	77.07	(2)

Table V: Mean average precision (MAP) for graph-based KWS systems in comparison with three state-of-the-art learning-based reference systems on the Alvermann Konzilsprotokolle (AK) and Botany (BOT) datasets. The first, second, and third best systems are indicated by (1), (2), and (3), respectively.

Method	BOT		AK		Average	
Reference (Learning)						
- CVCDAG	75.77	(2)	77.91		76.84	
- PRG	89.69	(1)	96.05	(1)	92.87	(1)
- QTOB	54.95		82.15		68.55	
Graph (Single)						
- CED ($n = 1$)	51.98		80.73		66.36	
- CED ($n = 3$)	52.18		82.42		67.30	
- CED ($n = 5$)	52.45		82.69		67.57	
Graph (Ensemble)						
- mean-Graph ($n = 1$)	71.83		81.52		76.68	
- mean-Graph ($n = 3$)	72.43		84.22		78.33	(2)
- mean-Graph ($n = 5$)	69.17		85.20	(2)	77.19	
- mean-Context ($m =$ `K`)	72.51	(3)	83.42		77.97	
- mean-Context ($m =$ `P`)	67.60		85.46	(3)	76.53	
- mean-All	71.67		84.49		78.08	(3)

[11] M. Ameri, M. Stauffer, K. Riesen, T. Bui, and A. Fischer, "Keyword Spotting in Historical Documents Based on Handwriting Graphs and Hausdorff Edit Distance," in *International Graphonomics Society Conference*, 2017.

[12] A. Fischer, S. Uchida, V. Frinken, K. Riesen, and H. Bunke, "Improving Hausdorff edit distance using structural node context," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9069, 2015, pp. 148–157.

[13] M. Stauffer, A. Fischer, and K. Riesen, "Ensembles for Graph-based Keyword Spotting in Historical Handwritten Documents," in *International Conference on Document Analysis and Recognition*, 2017.

[14] ——, "A Novel Graph Database for Handwritten Word Images," in *International Workshop on Structural, Syntactic, and Statistical Pattern Recognition*, 2016.

[15] P. Foggia, G. Percannella, and M. Vento, "Graph Matching and Learning in Pattern Recognition in the last 10 Years," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 28, no. 01, p. 1450001, 2014.

[16] K. Riesen, *Structural Pattern Recognition with Graph Edit Distance*, ser. Advances in Computer Vision and Pattern Recognition, Cham, 2015.

[17] T. C. Koopmans and M. Beckmann, "Assignment Problems and the Location of Economic Activities," *Econometrica*, vol. 25, no. 1, p. 53, 1957.

[18] U.-V. Marti and H. Bunke, "Using a statistical language model to improve the performance of an HMM-based cursive handwriting recognition systems," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 15, no. 01, pp. 65–90, 2001.

[19] I. Pratikakis, K. Zagoris, B. Gatos, J. Puigcerver, A. H. Toselli, and E. Vidal, "ICFHR2016 Handwritten Keyword Spotting Competition (H-KWS 2016)," in *International Conference on Frontiers in Handwriting Recognition*, 2016, pp. 613–618.