Ensembles for Graph-based Keyword Spotting in Historical Handwritten Documents

Michael Stauffer*§, Andreas Fischer†‡ and Kaspar Riesen*
*University of Applied Sciences and Arts Northwestern Switzerland,
Institute for Information Systems, Riggenbachstr. 16, 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, Pretoria, South Africa

***Abstract*—Keyword Spotting (KWS) offers a convenient way to improve the accessibility to historical handwritten documents by retrieving search terms in scanned document images. The approach for KWS proposed in the present paper is based on segmented word images that are represented by means of different types of graphs. The actual keyword spotting is based on matching a query graph with a set of document graphs using the concept of graph edit distance. In particular, we propose to employ ensemble methods for KWS with graphs. That is, a query graph is not matched against one but several different graphs representing the same document word. Eventually, we use different strategies to combine these individual graph dissimilarities. In an experimental evaluation on two benchmark datasets, the proposed ensemble methods outperform the individual ensemble members as well as four state-of-the-art reference systems based on dynamic time warping.**

I. Introduction

In the last years a trend towards digitalisation of handwritten historical documents can be observed. Yet, there is a lack of accessibility with respect to browsing and searching, especially as fully automatic transcriptions are often not feasible on this type of documents. Hence, *Keyword Spotting (KWS)* has been proposed as a more flexible approach [1]–[4]. Basically, KWS allows to retrieve any instances of a given keyword in a (large) document. In case of historical documents, KWS is restricted to an *offline* task based solely on document images, without temporal information about the writing process.

KWS methods can be roughly subdivided into *template-based* and *learning-based* algorithms. In case of template-based KWS, a query image is directly matched against a set of document images using matching algorithms like, for instance, *Dynamic Time Warping (DTW)* [2], [5], [6]. In case of learning-based KWS, a statistical model is trained *a priori* on a sufficiently large set of training data [3], [4], [7]. Generally, learning-based approaches achieve a higher accuracy, while template-based approaches are characterised by a higher flexibility as they do not require training data. In the present paper, the focus is on template-based KWS using diverse graph-based representations for handwritten words.

For a wide range of applications, graphs have been used as powerful and flexible representation formalism [8], [9]. Yet, with respect to handwriting recognition and especially KWS, only few attempts can be observed so far [10]–[14]. This is somehow surprising, as graphs offer a comprehensive formalism for the representation of handwriting. In particular, graphs are able to adapt their structure and size to the underlying handwritten words. Moreover, a great variety of approaches for measuring the dissimilarity of graphs (termed *graph matching*) have been proposed and applied during the last decades [9].

In the present paper, the actual keyword spotting is based on matching a query graph with a set document graphs (as proposed by the authors in [13]). However, rather than formalising a handwritten word by a single graph representation, a word is represented by several different graph representations as introduced in [14]. Thus, several query graphs (representing the same query word) can be matched with several document graphs (representing the same document word). Consequently, different strategies can be applied to combine the individual graph dissimilarities (derived on the different representations). This approach is a well known strategy from the field of *multiple classifier systems* [15], also referred to as *ensemble methods*.

Only limited ensemble methods have been proposed for KWS to date [16]–[18]. Existing ensemble methods for KWS are either based on dissimilarity measures of individual algorithms such as neural networks [16] or the ranking information of different retrieval indices [17], [18].

The remainder of this paper is organised as follows. In Section II, the proposed ensemble method for graph-based KWS is introduced. An experimental evaluation and comparison with individual graph-based systems and four reference systems based on DTW is given in Section III. Finally, Section IV concludes the paper and outlines possible future research activities.

II. Ensembles for Graph-based Keyword Spotting

In the present paper we extend the framework for graph-based KWS proposed in [13] towards ensemble strategies. The extended framework consists of three different processing steps as illustrated in Fig. 1. In the following subsections these three steps are presented in detail.

A. Image Preprocessing

First, original document images are preprocessed in order to minimise the influence caused by variations like skew, noisy background and document degradation. Eventually, on the basis of deskewed and binarised document images, single word images are automatically segmented (and manually corrected if necessary)[1]. For details on the preprocessing and segmentation step we refer to [13].

B. Graph Representation

Graphs are defined on the basis of preprocessed and segmented word images. A graph g is defined as a four-tuple $g = (V, E, \mu, \nu)$ where V and E are finite sets of nodes and 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, respectively. Additionally, graphs are often distinguished into *unlabelled* and *labelled* graphs. In the latter case, both nodes and edges can be labelled with an arbitrary numerical, vectorial, or symbolic label from L_v or L_e, respectively. In the former case we assume empty label alphabets, i.e. $L_v = L_e = \{\}$.

All of the following four graph extraction algorithms (originally presented in [14]) result in graphs where nodes are labelled with two-dimensional numerical labels, while edges remain unlabelled, i.e. $L_V = \mathbb{R}^2$ and $L_E = \{\}$.

- `Keypoint (K)`: The first graph extraction algorithm makes use of keypoints in the word images such as start, end, and junction points. These keypoints are represented as nodes that are labelled with the corresponding (x, y)-coordinates. Between pairs of keypoints further intermediate points are converted to nodes and added to the graph in equidistant intervals. Finally, undirected edges are inserted into the graph for each pair of nodes that is directly connected by a stroke.
- `Grid (G)`: The second graph extraction algorithm is based on a grid-wise segmentation of the word images. For every segment, a node is inserted into the graph and labelled by the (x, y)-coordinates of the centre of mass of this segment. Undirected edges are inserted between two neighbouring segments that are actually represented by a node. Eventually, the inserted edges are reduced by means of a *Minimal Spanning Tree* algorithm.
- `Projection (P)`: The next graph extraction algorithm works similar as `Grid`. However, this method is based on an adaptive segmentation of word images by means of horizontal and vertical projection profiles. A node is inserted into the graph for every 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.
- `Split (S)`: The last graph extraction algorithm is based on an iterative segmentation of word images. Similar to `Projection`, segments are iteratively split into smaller subsegments until the width and height of all segments is below a certain threshold. A node is inserted into the graph and labelled by the (x, y)-coordinates of the point closest to the centre of mass of every segment. For edges a similar procedure as for `Projection` is applied.

For every graph representation, an exemplary word of the employed datasets (i.e. the *George Washington letters (GW)* and the *Parzival manuscript (PAR)*) is shown in Fig. 2.

The dynamic range of the (x, y)-coordinates of each node label $\mu(v)$ is normalised with a z-score (regardless the extraction algorithm). 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.

C. Graph-based KWS Ensemble

In this section we present the details of the proposed graph-based KWS ensemble. We start with a brief review of the general graph matching paradigm used in our framework. Afterwards, we discuss the possibilities of obtaining and combining different graph dissimilarities for building a KWS ensemble. Finally, we describe the process of transforming a matching score into a retrieval index for KWS.

1) Approximate Graph Edit Distance: Given two graphs g_1 and g_2, the basic idea of graph edit distance is to transform g_1 into g_2 using a sequence of edit operations. A standard set of edit operations is given by *insertions*, *deletions*, and *substitutions* of both nodes and edges. A set $\{e_1, \ldots, e_k\}$ of k edit operations e_i that transform g_1 completely into g_2 is called an *edit path* $\lambda(g_1, g_2)$ between g_1 and g_2.

Let $\Upsilon(g_1, g_2)$ denote the set of all edit paths between two graphs g_1 and g_2. To find the most suitable edit path out of $\Upsilon(g_1, g_2)$, one commonly introduces a cost $c(e)$ for every edit operation e, measuring the strength of the corresponding operation. The idea of such a cost is to define whether or not an edit operation e represents a strong modification of the graph. Given an adequate cost model, 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

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

To compute d_{GED} often A*-based search techniques using some heuristics are employed [19]–[22]. Yet, this exhaustive search procedure is exponential with respect to the number

[1]The present KWS approach neglects any segmentation errors and can therefore be seen as an upper-bound solution.

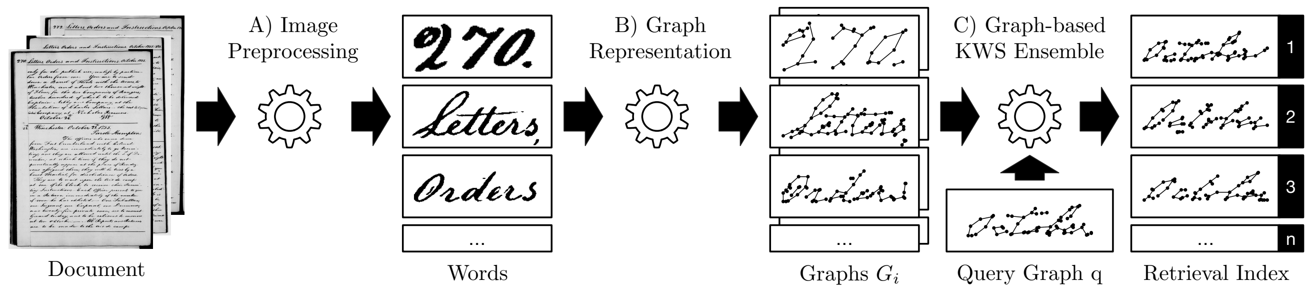

Fig. 1. Process of Graph-based Keyword Spotting of the Word "October"

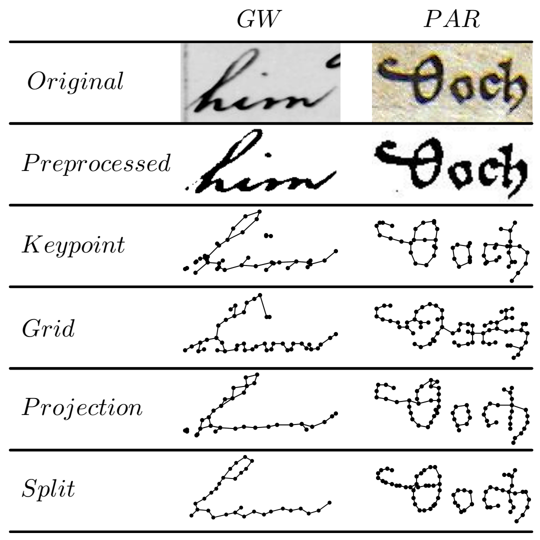

Fig. 2. Different representations of a sample word from both datasets.

of nodes of the involved graphs. In fact, graph edit distance belongs to the family of *Quadratic Assignment Problems (QAPs)* [23], which in turn belong to the class of $\mathcal{NP}$-complete problems[2].

However, the graph edit distance approximation framework introduced in [24] reduces the QAP of graph edit distance computation to an instance of a *Linear Sum Assignment Problem* (*LSAP*). Similar to QAPs, LSAPs deal with the question how the entities of two sets can be optimally assigned to each other. Yet, LSAPs can be – in contrast with QAPs – quite efficiently solved in polynomial time (see [25] for an exhaustive survey on LSAP solving algorithms).

For the remainder of this paper we make use of this approximation algorithm for graph edit distance computation and name the corresponding suboptimal distance $d_{\text{BP}}(g_1, g_2)$, or d_{BP} for short[3]. The used cost model is based on constant cost for node and edge deletions and insertions, i.e. $\tau_v \in \mathbb{R}^+$ and $\tau_e \in \mathbb{R}^+$, respectively. The cost for node substitutions should reflect the dissimilarity of the associated label attributes. In our application the nodes are labelled with (x, y)-coordinates and we use a weighted Euclidean distance on these labels to model the substitution cost. Formally,

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

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. Finally, we make use of a weighting factor $\beta \in [0, 1]$ between the total node and edge edit costs, respectively.

2) KWS Ensemble: For spotting keywords, in [13] the authors propose to match a query graph q with all graphs from a set of document graphs $G = \{g_1, \dots, g_N\}$ using a normalised version of the approximation d_{BP}. In particular, the graph edit distance d_{BP} is normalised by the sum of the maximum cost edit path between q and g, i.e. the sum of the edit path that results from deleting all nodes and edges of q and inserting all nodes and edges in g. Formally,

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

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

In the present paper we propose to use an ensemble for KWS. That is, we represent both the query graph q as well as the document graphs $g_i \in G$ with all formalisms introduced in Subsection II-B. Hence, a query is now represented by four graphs q_K, q_G, q_P, and q_S, i.e. one query graph per graph extraction method (`Keypoint (K)`, `Grid (G)`, `Projection (P)`, and `Split (S)`). The same accounts for the documents which are now represented by four sets of document graphs $\{G_K, G_G, G_P, G_S\}$. Rather than matching one query graph against one document graph, in our extension for one query we match q_K, q_G, q_P, and q_S with the corresponding set of document graphs. Consequently, four graph dissimilarities are obtained for each pair (q, g) of a query word q and a document word g.

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

[3]BP stands for *bipartite* since the LSAP is also termed *bipartite matching problem*.

The first ensemble strategy employed in this paper considers all four graph extraction methods by either choosing the minimal (termed $\min$), maximal (termed $\max$), or mean (termed mean) graph edit distance returned on the four representations. Formally, for one query word q represented by q_K, q_G, q_P, and q_S and one document word g represented by g_K, g_G, g_P, and g_S we define

$$\hat{d}_{\mathrm{BP}_{\min}}(q,g) = \min_{i\in\{K,G,P,S\}} \hat{d}_{\mathrm{BP}}(q_i,g_i) \quad ,$$
$$\hat{d}_{\mathrm{BP}_{\max}}(q,g) = \max_{i\in\{K,G,P,S\}} \hat{d}_{\mathrm{BP}}(q_i,g_i) \quad ,$$
$$\hat{d}_{\mathrm{BP}_{\mathrm{mean}}}(q,g) = \frac{1}{4} \sum_{i\in\{K,G,P,S\}} \hat{d}_{\mathrm{BP}}(q_i,g_i) \quad .$$

The second ensemble strategy only considers the two most promising individual graph extraction methods proposed in [14], viz. `Keypoint` and `Projection`. Two different weighted sums are applied to combine the respective distances with each other (termed sum_α and $\mathrm{sum}_{\mathrm{map}}$). Formally,

$$\hat{d}_{\mathrm{BP}_{\mathrm{sum}_\alpha}}(q,g) = \gamma\, \hat{d}_{\mathrm{BP}}(q_K,g_K) + (1-\gamma)\, \hat{d}_{\mathrm{BP}}(q_P,g_P) \quad ,$$
$$\hat{d}_{\mathrm{BP}_{\mathrm{sum}_{\mathrm{map}}}}(q,g) = \delta\, \hat{d}_{\mathrm{BP}}(q_K,g_K) + \epsilon\, \hat{d}_{\mathrm{BP}}(q_P,g_P) \quad ,$$

where γ denotes a user defined weighting factor, and δ and ϵ denote weighting factors based on the mean average precision of the original KWS systems operating on `Keypoint` and `Projection` graphs, respectively.

The five combinations of bipartite graph edit distances introduced in this section can now be used for KWS (for the remainder of the present paper $\hat{d}_{\mathrm{BP}_{\dots}}$ stands for any of the five combined distances).

3) Retrieval Index: The actual keyword spotting relies on retrieval indices which are based on $\hat{d}_{\mathrm{BP}_{\dots}}$. We optimise, retrieval indices for *local* and *global* threshold scenarios (see Section III-C). In a real world scenario, local thresholds are used in case of a vocabulary of common keywords, while a global threshold is used for arbitrary out- of-vocabulary keywords. Generally, global thresholds are regarded as the more realistic but also more difficult scenario. That is, in case of local thresholds, the KWS accuracy is independently measured for every keyword, while in case of global thresholds, the KWS accuracy is measured for every keyword with one single threshold.

The distance $\hat{d}_{\mathrm{BP}_{\dots}}$ is used to derive a first retrieval index for local thresholds by

$$r_1(q,g) = -\hat{d}_{\mathrm{BP}_{\dots}}(q,g) \quad .$$

For the second retrieval index r_2 for global thresholds, $\hat{d}_{\mathrm{BP}_{\dots}}$ is further normalised by using the average distance of a query graph q to its k nearest document graphs, i.e. the document graphs $\{g_{(1)},\dots,g_{(k)}\}$ with smallest distance values to q. Formally, we use

$$\bar{d}_k(q) = \frac{1}{k}\sum_{i=1}^{k} \hat{d}_{\mathrm{BP}_{\dots}}(q,g_{(i)}) \quad .$$

to derive

$$\hat{\hat{d}}_{\mathrm{BP}_{\dots}}(q,g) = \frac{\hat{d}_{\mathrm{BP}_{\dots}}(q,g)}{\bar{d}_k(q)} \quad .$$

Eventually, the distance $\hat{\hat{d}}_{\mathrm{BP}}$ is used to derive the second retrieval index by

$$r_2(q,g) = -\hat{\hat{d}}_{\mathrm{BP}_{\dots}}(q,g) \quad .$$

Rather than defining k as a constant, we dynamically adapt k to every query graph q. We define k such that the distance $d_{\mathrm{BP}_{\dots}}(q,g_{(k)})$ of q to its k-th nearest document graph $g_{(k)}$ is equal to

$$d_{\mathrm{BP}_{\dots}}(q,g_{(k)}) = \bar{d}_m(q) + \theta\,(\bar{d}_N(q) - \bar{d}_m(q)) \quad ,$$

where $m \in \mathbb{N}$ and $\theta \in [0,1]$ are user defined parameters and N refers to the number of document graphs. The value of $\bar{d}_m(q)$ refers to the mean distance of q to its m nearest neighbours and $\bar{d}_N(q)$ refers to the mean distance to all document graphs available. This sum reflects the level of the dissimilarities of q to the graphs in its direct neighbourhood. If the sum is large, k is automatically defined large, too. This in turn increases $\bar{d}_k(q)$, which ultimately increases the scaling for $\hat{\hat{d}}_{\mathrm{BP}_{\dots}}$.

III. Experimental Evaluation

In the following subsections both the experimental protocol and the evaluations are explained in detail. First, the datasets employed are introduced in Subsection III-A. Second, the reference systems are presented in Subsection III-B. Finally, the optimisation of the proposed ensembles and the comparison with several reference systems are described in Subsection III-C and III-D, respectively.

A. Datasets

The experimental evaluation is carried out on two historical document collections, viz. the George Washington letters and the Parzival manuscript.

GW is based on letters that are written in English and consists of twenty pages with a total of 4,894 handwritten words[4]. Variations caused by both degradation and writing style are low.

PAR is based on a manuscript that is written in Middle High German and consists of 45 pages with a total of 23,478 handwritten words[5]. There are markable variations caused by degradation, while variations caused by writing style are low.

To optimise the parameters of the KWS framework, ten different keywords (with different word lengths) are manually selected on both datasets. Furthermore, we define a validation set that consists of 10 random instances per keyword instance

[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

and 900 additional random words (in total 1,000 words)[6]. For the final experiment using the optimised parameter settings we use the same training and test sets as proposed in [4]. In Table I, the number of keywords, as well as the size of the training- and test set are shown for both datasets.

TABLE I
NUMBER OF KEYWORDS AS WELL AS THE SIZE OF THE TRAINING AND TEST SET FOR BOTH BENCHMARK DATASETS.

Dataset	Keywords	Train	Test
GW	105	2,447	1,224
PAR	1,217	11,468	6,869

B. Reference Systems

We evaluate our ensembles in two different experiments. First, we compare the results of our ensemble framework with the results of the individual members of the ensemble. Second, we compare the overall best ensemble with four KWS systems based on *Dynamic Time Warping* (termed DTW'01 [26], DTW'08 [5], DTW'09 [27], and DTW'16 [28]).

DTW has been widely used in the field of KWS and can be regarded as one of the standard methods for KWS. DTW operates on sequences of features vectors acquired by means of a sliding window over handwritten word or line images. For matching two sequences of features vectors $X = \{\mathbf{x}_1, \dots, \mathbf{x}_m\}$ and $Y = \{\mathbf{y}_1, \dots, \mathbf{y}_n\}$, DTW optimally aligns (warps) this sequences along one common time axis using a dynamic programming approach.

The four reference systems based on DTW actually differ with respect to the features extracted from the word or line images. DTW'01 [26] makes use of geometrical features, while DTW'08 [5] and DTW'09 [27] make use of Histogram of Oriented Gradient features. Finally DTW'16 [28] employs Deep Learning features.

C. Optimisation of the Parameters

The performance of all KWS systems is indicated for local and global thresholds by the *Recall (R)* and *Precision (P)*

$$R = \frac{TP}{TP + FN} \text{ and } P = \frac{TP}{TP + FP} \quad ,$$

which are based on the number of *True Positives (TP)*, *False Positives (FP)*, and *False Negatives (FN)*.

In case of global thresholds, the *Average Precision (AP)* is measured, which is the area under the Recall-Precision curve for all keywords given a single (global) threshold. In case of local thresholds, the *Mean Average Precision (MAP)* is measured, that is the mean over the AP of each individual keyword given a (local) threshold.

The optimisation of the parameters is conducted for local and global thresholds independently in three subsequent steps.

First, the parameters for graph edit distance are optimised. That is, we evaluate 25 pairs of constants for node and edge deletion/insertion costs ($\tau_v = \tau_e = \{1, 4, 8, 16, 32\}$) in combination with the weighting parameters $\alpha = \{0.1, 0.3, 0.5, 0.7, 0.9\}$ and $\beta = \{0.1, 0.3, 0.5, 0.7, 0.9\}$. Hence, we evaluate a total of 625 parametrisations per graph extraction method and dataset (resulting in 5,000 settings in total). In Table II the optimal cost function parameters are given for all graph extraction algorithms.

[6]The small number of keywords is due to the fact that we optimise a large number of parameters.

TABLE II
OPTIMAL COST FUNCTION PARAMETER FOR GRAPH EDIT DISTANCE COMPUTATION.

	Method	τ_v	τ_e	α	β
GW	`Keypoint`	4	1	0.1	0.5
	`Grid`	4	1	0.1	0.7
	`Projection`	4	1	0.1	0.5
	`Split`	4	1	0.1	0.5
PAR	`Keypoint`	4	4	0.3	0.5
	`Grid`	4	1	0.5	0.7
	`Projection`	4	1	0.5	0.5
	`Split`	4	1	0.3	0.3

Second, the parameters of the ensemble method are optimised using the optimised graph extraction and graph matching parameters (these parameters have been optimised before in [13], [14]). Hence, the weighting factor $\gamma \in \{0.1, \dots, 0.9\}$ for the ensemble sum_α is the sole parameter that needs to be optimised (all other ensemble strategies need no parameter tuning).

In Table III, the MAP is given for the tested parameter settings for γ on both benchmark datasets. Note that the best performing parameter setting is indicated in bold face.

TABLE III
OPTIMAL γ FOR THE sum_α ENSEMBLE.

	MAP	
γ	GW	PAR
0.1	73.21	**100.00**
0.2	73.34	99.19
0.3	**75.23**	99.19
0.4	71.84	99.19
0.5	71.75	99.19
0.6	71.34	96.33
0.7	72.00	94.65
0.8	72.10	94.28
0.9	72.05	93.95

Finally, the retrieval index r_2 (used for global thresholds) needs to be optimised. In particular, parameter m and threshold scaling factor θ are individually optimised for each ensemble (N is defined by the number of document graphs). We tested 1,000 parameters pairs (m, θ) with $m \in \{10, 20, \dots, 90, 100\}$ and $\theta \in \{0.01, 0.02, \dots, 0.99, 1.00\}$. In Table IV, the optimal parameter settings for r_2 are given for all ensembles.

Generally, we aim to reduce interclass variations in order to optimise the AP for a single global threshold. Thus, the differences of the optimal parameter settings are due to different distributions of the graph edit distances for GW and

TABLE IV
OPTIMAL m AND θ FOR RETRIEVAL INDEX r_2.

	Method	m	θ
GW	min	70	0.08
	max	40	0.10
	mean	60	0.06
	sum_α	70	0.04
	$\mathrm{sum}_\mathrm{map}$	90	0.02
PAR	min	10	0.72
	max	10	0.61
	mean	10	0.64
	sum_α	10	0.61
	$\mathrm{sum}_\mathrm{map}$	10	0.61

PAR, respectively. In case of GW, the graph edit distances for global thresholds are optimised by considering a rather large neighbourhood m and small weighting factor θ. In case of PAR, we can observe the opposite case.

D. *Results and Discussion*

Using the optimal parameter configurations, the proposed KWS ensembles are first compared with the four individual KWS systems proposed in [13], [14] (actually used as ensemble members). The MAP (for local thresholds) and the AP (for global thresholds) are given for all individual methods and all ensembles in Table V. We indicate the absolute gain or loss in the accuracy of the ensembles when compared with the best performing individual system (i.e. `Keypoint` for GW and `Projection` for PAR).

On GW we observe a statistically significant improvement (t-test, $\alpha = 0.05$) of more than 4% for both classifier ensembles min and $\mathrm{sum}_\mathrm{map}$ when compared with the MAP accuracy of `Keypoint`. With respect to AP we observe improvements of up to 2% when compared with the best individual member of the ensemble (achieved by mean and $\mathrm{sum}_\mathrm{map}$).

On PAR we observe even larger statistically significant improvements. In particular, the ensemble mean improves the best individual MAP by 13% and 11% with local and global thresholds, respectively. Also the sum-based ensembles sum_α and $\mathrm{sum}_\mathrm{map}$ clearly outperform the best individual member.

Overall, the ensemble mean achieves in two out of four cases the best and and in two cases the second and third best result, while $\mathrm{sum}_\mathrm{map}$ achieves once the best result and three times the second best result. Hence, we conclude that mean and $\mathrm{sum}_\mathrm{map}$ are the best performing ensembles. On the other hand, we observe that the ensemble max is not a well suited strategy in our specific scenario as it achieves the worst result of all ensembles in all four cases.

For the comparison with four state-of-the-art systems based on DTW [5], [26]–[28], we consider the ensemble mean only. In Table VI the MAP and AP of this particular ensemble is compared with the four reference systems. On both datasets and threshold scenarios the proposed system clearly outperforms all DTW-based reference systems. The ensemble achieves improvements of up to 70% (on the PAR dataset using local thresholds).

IV. CONCLUSION AND OUTLOOK

In this paper a procedure to build an ensemble for the task of graph-based keyword spotting is presented. Rather than representing a word by a single graph representation, the proposed ensemble methods make use of several graph representations at a time. In particular, we use four different graph representation applicable to segmented word images.

Basically, a keyword represented as query graph can be retrieved from a set of document graphs by pairwise graph matchings. In the proposed approach of ensemble methods, several query graphs are matched against several document graphs (i.e. one query and one document graph per graph extraction method). Consequently, several graph distances are obtained, which are in turn combined by means of different statistical measurements.

For the experimental evaluation, the proposed ensemble methods are tested on two different benchmark datasets, viz. George Washington and Parzival, using local and global thresholds. The proposed ensemble strategies are first compared with the individual graph-based KWS systems. We observe that all ensemble methods, expect one, clearly outperform the individual methods on both threshold scenarios. Especially, the strategies mean and $\mathrm{sum}_\mathrm{map}$ achieve promising improvements when compared to the individual KWS systems. Last but not least, the overall best ensemble method clearly outperforms four state-of-the-art reference systems based on DTW.

One might argue that ensemble and in particular graph-based approaches are limited by the increased complexity of the matching procedure when compared to statistical approaches. However, recent papers (e.g. [29]–[31]) show that the complete KWS procedure with graphs can be substantially speeded up by filters and other heuristics.

The improvements achieved by ensemble methods motivate several lines of future research. First, new graph representations, which provide different, complementary perspectives on the handwritten words are expected to further improve the diversity of the ensemble and hence the KWS performance. Secondly, different graph matching methods could contribute to obtain more ensemble members. Finally, the combination methods can be extended, for example by including learning-based combination systems.

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] T. Rath and R. Manmatha, “Word image matching using dynamic time warping,” in *Computer Vision and Pattern Recognition*, vol. 2, 2003, pp. II–521–II–527.

[3] J. A. Rodríguez-Serrano and F. Perronnin, “Handwritten word-spotting using hidden Markov models and universal vocabularies,” *Pattern Recognition*, vol. 42, no. 9, pp. 2106–2116, 2009.

TABLE V
INDIVIDUAL CLASSIFIERS VS. ENSEMBLES (USING DIFFERENT COMBINATION STRATEGIES)

		GW				PAR			
	Method	MAP	±	AP	±	MAP	±	AP	±
Individual	`Keypoint`	66.08		55.22		62.04		60.76	
	`Grid`	60.02		46.09		56.50		46.00	
	`Projection`	61.43		49.34		66.23		62.38	
	`Split`	60.23		48.08		59.44		56.25	
Ensemble	min	**70.56**	+4.48	56.82	+1.59	67.90	+1.67	62.33	–0.05
	max	62.58	–3.50	47.94	–7.29	67.57	+1.34	50.59	–11.79
	mean	69.16	+3.08	57.11	+1.89	**79.38**	+13.15	**73.77**	+11.39
	sum_α	68.44	+2.36	55.78	+0.56	74.51	+8.28	68.12	+5.74
	sum_{map}	70.20	+4.12	**57.38**	+2.16	76.80	+10.57	73.56	+11.18

TABLE VI
ENSEMBLE mean VS. FOUR REFERENCE SYSTEMS BASED ON DTW

		GW		PAR	
	Method	MAP	AP	MAP	AP
	mean	**69.16**	**57.11**	**79.38**	**73.77**
DTW	DTW'01 [26]	45.26	33.24	46.78	50.67
	DTW'08 [5]	63.39	41.20	47.52	55.82
	DTW'09 [27]	64.80	43.76	73.49	69.10
	DTW'16 [28]	68.64	56.98	72.38	72.71

[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] 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.

[6] ——, "A model-based sequence similarity with application to handwritten word spotting." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2108–20, 2012.

[7] F. Perronnin and J. A. Rodríguez-Serrano, "Fisher Kernels for Handwritten Word-spotting," in *International Conference on Document Analysis and Recognition*, 2009, pp. 106–110.

[8] D. Conte, P. Foggia, C. Sansone, and M. Vento, "Thirty Years Of Graph Matching In Pattern Recognition," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 18, no. 03, pp. 265–298, 2004.

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

[10] 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 *International Workshop on Document Analysis Systems*, 2014, pp. 207–211.

[11] Q. A. Bui, M. Visani, and R. Mullot, "Unsupervised word spotting using a graph representation based on invariants," in *International Conference on Document Analysis and Recognition*, 2015, pp. 616–620.

[12] P. Riba, J. Llados, and A. Fornes, "Handwritten word spotting by inexact matching of grapheme graphs," in *International Conference on Document Analysis and Recognition*, 2015, pp. 781–785.

[13] 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.

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

[15] L. I. Kuncheva, *Combining Pattern Classifiers*, Hoboken, NJ, USA, 2004.

[16] V. Frinken, A. Fischer, and H. Bunke, "Combining Neural Networks to Improve Performance of Handwritten Keyword Spotting," in *Multiple Classifier Systems*, 2010, pp. 215–224.

[17] G. Louloudis, A. Kesidis, and B. Gatos, "Efficient Word Retrieval Using a Multiple Ranking Combination Scheme," in *International Workshop on Document Analysis Systems*, 2012, pp. 379–383.

[18] H. Wei, G. Gao, and X. Su, "A multiple instances approach to improving keyword spotting on historical Mongolian document images," in *International Conference on Document Analysis and Recognition*, 2015, pp. 121–125.

[19] A. Dumay, R. van der Geest, J. Gerbrands, E. Jansen, and J. Reiber, "Consistent inexact graph matching applied to labelling coronary segments in arteriograms," in *International Conference on Pattern Recognition*, 1992, pp. 439–442.

[20] L. Gregory and J. Kittler, "Using Graph Search Techniques for Contextual Colour Retrieval," in *International Workshop on Structural, Syntactic, and Statistical Pattern Recognition*, 2002, pp. 186–194.

[21] S. Berretti, A. Del Bimbo, and E. Vicario, "Efficient matching and indexing of graph models in content-based retrieval," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 10, pp. 1089–1105, 2001.

[22] S. Fankhauser, K. Riesen, and H. Bunke, "Speeding Up Graph Edit Distance Computation through Fast Bipartite Matching," in *Graph-Based Representations in Pattern Recognition*, 2011, pp. 102–111.

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

[24] 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.

[25] R. Burkard, M. Dell'Amico, and S. Martello, *Assignment Problems*, 2009.

[26] 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.

[27] 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.

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

[29] M. Stauffer, A. Fischer, and K. Riesen, "Speeding-Up Graph-based Keyword Spotting in Historical Handwritten Documents," in *Graph-Based Representations in Pattern Recognition*, 2017.

[30] ——, "Speeding-Up Graph-based Keyword Spotting by Quadtree Segmentations," in *International Conference on Computer Analysis of Images and Patterns*, 2017.

[31] 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.