International Journal of Pattern Recognition and Artificial Intelligence

Annotation-free Word Spotting with Bag-of-Features HMMs

Leonard Rothacker, Fabian Wolf and Gernot A. Fink

Department of Computer Science
TU Dortmund University
Dortmund, 44227, Germany,
Email: {firstname.lastname}@cs.tu-dortmund.de

The *annotation-free* word spotting method that is proposed in this paper makes document images searchable without requiring any labeled training data. Thus, our method supports the exploration of a document collection directly without demanding any manual efforts from the users for the preparation of a training dataset. Our method works in the query-by-example scenario where the user selects an exemplary occurrence of the query word. Afterwards, the entire collection of document images is searched according to visual similarity to the query. The proposed method requires only minimal assumptions about the visual appearance of text. This is achieved by processing document images as a whole without requiring a given segmentation of the images on word level or on line level. Therefore, the method is also *segmentation-free*. Word size variabilities can be handled by representing the sequential structure of text with a statistical sequence model. In order to make the computationally costly application of the sequence model feasible in practice, regions are retrieved according to *approximate* similarity with an efficient model decoding algorithm. Re-ranking these regions according to the visual similarity obtained with the sequence model leads to highly accurate word spotting results. The method is evaluated on five benchmark datasets. In the segmentation-free query-by-example scenario where no annotated training data is available, the method outperforms all other methods that have been evaluated on any of these five benchmarks.

Keywords: Word spotting; Bag-of-features; Hidden Markov models; Historic documents.

1. Introduction

The ability to automatically search documents for occurrences of query words supports the analysis and interpretation of their contents considerably. It is a widely used standard functionality for digital documents containing machine-readable text. Unfortunately, this functionality is not directly available for document images. In order to be processed with automatic search queries in the same way, it would be required to transcribe the document images into machine-readable textual representations first. This can be difficult for non-standardized documents and is, therefore, costly and error-prone in these cases. Non-standardized document collections are highly variable in their visual appearance and cannot be transcribed automatically with off-the-shelf optical character recognition software. Setting up a full transcription recognizer requires huge amounts of annotated training data that is representative for the application domain. Such a training dataset typi-

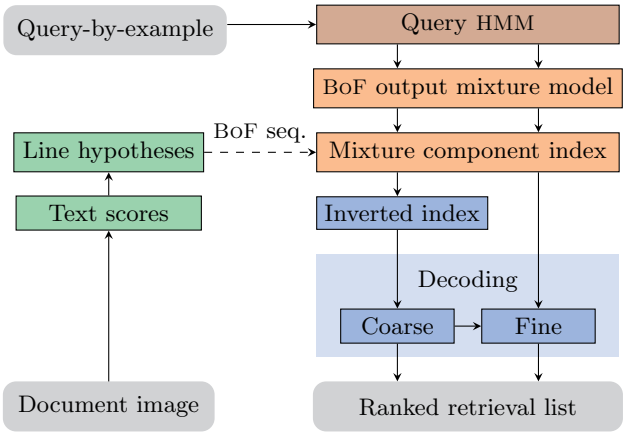


Figure 1: Annotation-free and segmentation-free query-by-example word spotting system.

cally consists of machine-readable transcriptions of document images on line level. For non-standardized and in particular for historic documents this can hardly be obtained automatically but requires considerable manual effort.

In contrast to searching in machine-readable textual representations for occurrences of query words, word spotting searches on document image level based on visual appearance. This makes word spotting a specialized image retrieval technique. Word spotting systems are classified with respect to their supported query modalities, segmentation of the document images and the required amount of annotated training samples.[1,17]

The proposed word spotting method uses the query-by-example modality. This means that the user has to provide an exemplary instance of the query word in a document image. The proposed query model is estimated only from this individual example and no additional annotated training data is used, i. e., the method is *annotation-free*. While this limits the visual variability of the text in the document images that the proposed word spotting method can cope with, users are supported with automatic search functionality directly after acquiring a new collection of document images. In this regard, it is also important that the proposed word spotting method does not require a given segmentation of the document into lines or words, i. e., it is *segmentation-free*.

The most important contribution of the proposed annotation-free and segmentation-free query-by-example method is its direct applicability to a new document collection. No annotated training dataset is required. Neither for model estimation nor for optimizing meta parameters. This sets the application scenario of the proposed method apart from the application scenario of training-based methods, like neural networks, e. g., Ref. 37. Even though annotated training data can be generated synthetically, the synthetic samples must be representative for the application domain in order to achieve top performance.[51] This is particularly relevant

for spotting in historic documents.[18]

Given that the proposed query model has only seen a single example of the query word, the proposed word spotting system yields very high performance. This is achieved by exploiting the properties of document images on a very general level. The use of bag-of-features (BoF)[32] allows for adapting the feature representation to the problem domain in an unsupervised manner. Modeling BoF sequences with a hidden Markov model (HMM) takes the length variability of text into account. A decoding algorithm for word spotting with semi-continuous HMMs[14] allows for efficient retrieval in a coarse-to-fine decoding framework. This is achieved by indexing mixture component probabilities for BoF vector sequences that are extracted from line hypotheses. Fig. 1 shows a schematic visualization. The proposed method is the summary of the dissertation in Ref. 43. The contributions go far beyond the work on word spotting with BoF-HMMs that has been published before: Word spotting with BoF-HMMs in a patch-based decoding framework has been presented in Ref. 45. In contrast to computing similarities for entire patches, the proposed method decodes the most likely occurrence of the query within a patch. Based on the efficient visual-word voting algorithm in Ref. 46, the algorithm is extended to mixture components in semi-continuous HMMs. Extending the work in Ref. 44, we present a study on four different mixture models for word spotting with BoF-HMMs.

The rest of the paper is organized as follows. Sec. 2 presents related works on word spotting. The proposed method for segmentation-free word spotting with BoF-HMMs is discussed in Sec. 3. An experimental evaluation of the proposed method is presented in Sec. 4. We compare our method to results from the literature on five different benchmark datasets in Sec. 4.4. Finally, Sec. 4.6 presents experiments to confirm different architectural design choices and meta parameters.

2. Annotation-free word spotting

Common taxonomy distinguishes word spotting methods based on their query modality, the application of a learning phase and the necessity of an independent segmentation step.[1,17] *Segmentation-based* methods assume that the segmentation of document images into word images can be solved independently of retrieval. In contrast, *segmentation-free* approaches perform segmentation and retrieval jointly. The level of integration varies from solving these two task with dynamic programming, e. g., with HMMs, to selecting segments from a number of competing region hypothesis based on similarity to the query. The proposed method is *segmentation-free*. Another key aspect is whether the learning phase requires annotated training data. Annotations are typically created manually and very costly to obtain. The proposed method is *annotation-free* because its application does not require an annotated training dataset. In the following, methods for extracting relevant document image regions, representing document image regions numerically and computing similarities with respect to the query are discussed. The presentation is focussed on annotation-free word spotting.

2.1. *Document image regions*

For spotting the query word in a document image, plausible document image regions must be obtained. These regions are processed during retrieval and are the basis for creating a ranked retrieval list. Word spotting methods can be grouped in the categories *word-level*, *line-level* and *document-level* according to the regions required by the method.

Word-level approaches require a given segmentation of the document image into words. The main limitation is that errors in the required segmentation will directly lead to errors in the retrieval result, cf. Ref. 17.

Line-level approaches only require a given segmentation of the document image into lines which are represented as sequences of feature vectors. Therefore, line-based word spotting methods are segmentation-free on line-level, e. g., Ref. 15, 37.

Document-level approaches are based on hypotheses that provide competing alternatives. The large majority of irrelevant regions can be filtered out with *non-maximum suppression*, cf. Ref. 49. Document region hypotheses can be categorized in patch hypotheses and word hypotheses. Patch hypotheses are mostly dependent on the query word. The patch geometry, e. g., width and height, is derived from the geometry of the query bounding box, e. g., Ref. 4, 49. The patch positions are either uniformly distributed in the entire document image[4, 49] or within text areas, e. g., Ref. 16. Since the size of the query is given by the query word image, patch-based approaches are often applied in query-by-example scenarios. Word hypotheses are document image regions that are likely to contain words. Words are detected in the document images independently of the query. Detectors are either defined heuristically, e. g., Ref. 24, 22, or estimated from annotated sample data as presented in Ref. 60, 47.

The proposed method combines aspects from word spotting on line-level and on document-level. Retrieval is performed in a patch-based framework. The patch size is based on the size of the query word image. Similar to spotting words in text lines, the most likely occurrences of the query within the most relevant patches is decoded with an HMM.

2.2. *Feature-based retrieval*

Retrieval becomes a nearest neighbor search, if the query is represented such that it can directly be compared with document regions in the same feature space. In the query-by-example scenario, the query word image is used as a template and document image regions are sorted according to similarity to the query, cf. Ref. 17. For this purpose, it is important to choose a suitable similarity measure. Within lower dimensional feature spaces, measures are often based on Euclidean distance, e. g., Ref. 16, 22. Within higher dimensional spaces, Euclidean distance is not discriminative, cf. 31. For high dimensional and histogram-like representations such as *spatial pyramids*[23] or *histograms of oriented gradients*,[12] cosine similarity has been successful, e. g., Ref. 4, 49.

Apart from measuring similarity between holistic document region representations, different approaches have been investigated for computing similarities based on local features. A simple example in this regard is dynamic time warping, cf. Ref. 50. Similarity is based on an optimal alignment of two sequences of feature vectors.[56] The features can be considered as local, since they are extracted frame-wise in writing direction.

A further degree-of-freedom is added when local features are extracted at interest points. Local image features from the document images can be matched with local image features in the query word template. The cumulative distance[24] or average distance[61] of features that have been matched in a cohesive elastic manner can be used as a similarity measure. Spatial consistency is enforced by restricting matches to local neighborhoods around reference points in the document images, e. g., based on regions of interests and guides.[24]

The proposed method addresses the typical writing variabilities by representing document regions with sequences of BoF vectors. The temporal representation combines specific spatial information with the orderless visual word histograms. This leads to a suitable trade-off between specificity and generalization capabilities.

2.3. *Model-based retrieval*

Model-based retrieval uses statistical models, like HMMs, or discriminative models, like *support vector machines* (SVMs) or *convolutional neural networks* (CNNs). Statistical approaches model the distribution of feature vectors that are typical for the query, e. g., Ref. 42, 57. Discriminative approaches separate the feature space into relevant and non-relevant with respect to the query, e. g., Ref. 4, 55. SVMs and HMMs are particularly relevant in the context of the proposed method due to their flexibility with respect to the required amount of annotated training data.[4,41] In contrast, state-of-the-art CNN-based word spotting methods typically use large training datasets with annotated samples from the application domain, cf. e. g., Ref. 37, 55. Training on synthetic data alone did not yield competitive results when the visual appearance of text in the document images is not represented well in the synthesized samples.[18,51]

Distances to the SVM hyperplane are interpreted as similarity scores in order to obtain the ranked retrieval list. In Ref. 34, class information is modeled on word level which limits the user to a predefined lexicon. An annotation-free query-by-example scenario is considered in Ref. 4. For estimating the SVM, non-relevant examples are randomly sampled from the document collection. Based on a single example of the query word, multiple word images that are relevant to the query are given as shifted instances of the query image region. The concept is known as exemplar SVM.[29]

Statistical approaches model the generation of feature vectors using statistical distributions over the document-region feature-space. Thus, retrieval is based on the probability of generating the document-region features with the query model.

For this purpose, the common approach is Bayes' theorem, cf. Ref. 9. The likelihood of the features conditioned on the query, is normalized with the evidence. Since the logarithm of this odds ratio is used in practice, this form of normalization is referred to as log-odds scoring.[8]

In practice, modelling the evidence is a major challenge because it has to represent the semantic structure over the entire feature vector space. In Ref. 57, this is addressed by limiting the queries to a lexicon. The evidence is modeled as the total probability of the likelihoods over all word classes. The quality of a score depends on how well the document region is represented by any of the class models.

The most common statistical method for query-by-string word spotting are HMMs. A widely noticed approach is to model the occurrence of the query word in a text line.[15] Given a document image, the segmented text line images are normalized and represented with sequences of feature vectors. Within the statistical approach, the likelihood is represented by an HMM query model and the evidence is modeled by the so-called filler model.[38] The filler models an arbitrary sequence of characters. Using the filler and the model for the query word, the query model is a compound HMM which represents the occurrence of the query word in a text line. A probabilistic score that is based on an approximation of the query posterior probability given the feature vector sequence of the text line, is obtained by the log-odds of the query model and filler model, see Ref. 15. The quality of the score depends on the recognition result that is computed in the filler model. Consequently, transcription errors will result in smaller differences between the scores of the query and filler model. Empirically, the effect can be confirmed in different HMM-based word spotting approaches that have increased recognition capabilities through integration of n-gram language models[38] and lexicon-based recognition.[57]

Finally, HMMs have also been used for annotation-free query-by-example word spotting scenarios where only the exemplary occurrence of the query word is given. In order to estimate a query word HMM from a single example, a semi-continuous model is used in Ref. 41. The shared *Gaussian mixture model* (GMM) is estimated in an unsupervised manner and only the state-dependent mixture and transition probabilities are estimated at query time. In contrast, continuous HMMs are estimated with large annotated training datasets.[15,37,57] In Ref. 41, a GMM is used for HMM score normalization. Word region hypotheses are ranked according to the log-odds scores of the query model and the so-called universal background model.

Since the background model is unspecific to the semantic structure of the problem domain, regions that are not represented well by the query model are over-estimated. This is essentially the same issue as with filler model normalization.[15,38] The proposed method does not perform score normalization with a background model for this reason.

2.4. *Efficiency*

In practice, word spotting systems have to search large collections of document images. In order to guarantee fast retrieval times, operations that are independent of the query are performed initially. At query time, this indexed information is accessed efficiently. Similarity measures are adjusted to data structures that integrate well with the document region representation e. g., Ref. 25, 49, 40, 57. Since this adjustment often results in approximate similarities, additional re-ranking can improve retrieval results considerably.[4,52] In this scenario, word spotting is performed in a two-stage process. In the first stage, the objective is to quickly obtain short retrieval lists that contain mostly all relevant document image regions. Computationally expensive methods for optimizing the ranking in the resulting retrieval list are applied in the second stage. A trade-off between efficiency and performance can be made by selecting the number of re-ranked regions, cf. e. g., Ref. 4.

The most common data structure for efficient retrieval is the *inverted file structure* (IFS), cf. Ref. 5. The basic idea is to represent document image regions with a codebook of feature codewords. Typically, the codebook is either defined heuristically[25] or obtained automatically through clustering.[52] The inverted index contains an entry for each codeword and stores links to the associated document image regions. Once the query codewords have been obtained, the index allows for fast look-ups. The purpose of the IFS is to retrieve as few document regions including mostly all document regions that are relevant to the query.

The most widely and successfully used approximate similarity measure for word spotting is based on product quantization.[20] It has been applied to segmentation-free word spotting[4,40,49] and is capable of handling large amounts of region representations that are obtained in patch-based approaches. Approximate similarity is based on the similarity of the quantization codewords that are used for representing the query and the document regions.

The proposed method performs retrieval in a two-stage framework. By indexing mixture components in the semi-continuous HMM with an IFS, potentially relevant document image regions are retrieved fast.

3. BoF-HMMs for segmentation-free word spotting

The proposed word spotting system includes methods for generating document region hypotheses, their representation, query modeling and retrieval. The close integration of the components allows for using two different decoding strategies in the same methodological framework. For this purpose, HMMs are semi-continuous. This makes the HMM output model largely independent of the HMM states, allowing for great flexibility with respect to model estimation and model decoding.

The key idea for retrieval with BoF-HMMs is to index representations that are independent of the query. For this purpose, text detection is performed on the document images. Based on text detector scores, *text hypotheses* are computed. Afterwards, text hypotheses are combined in order to define *line hypotheses*, see Sec. 3.1.

For each line hypothesis, a sequence of BoF vectors is extracted, see Sec. 3.2. Using a *mixture model*, BoF vectors are modeled as outputs of the HMM probabilistically. Since the mixture-component probabilities are independent of the query, they are stored in a *mixture component index*, see Sec. 3.3. Different mixture models are considered for modeling the high-dimensional and sparse BoF vectors, see Sec. 3.4. For obtaining the query word HMM from a single sample, it is sufficient to estimate only the state-dependent HMM parameters, see Sec. 3.5. Retrieval is performed in a patch-based framework using two HMM decoding stages, see Sec. 3.6. In the coarse stage, potentially relevant patches are detected with a voting scheme that is inspired by the *generalized Hough transform*.[6] In the fine stage, the most likely occurrence of the query word within each potentially relevant patch is inferred with the *Viterbi algorithm*, cf. Ref. 14. The context of the query word within a patch is represented with a background HMM and whitespace HMMs.

3.1. *Document image regions*

Document region hypotheses are used in order to define the search context for spotting words in document images and estimate whitespace HMMs. Therefore, they are not used in order to segment the document image into word images. Text, line and whitespace hypotheses are used within the retrieval process for segmentation-free word spotting on document level, cf. Sec. 2.1.

Text hypotheses represent text components in document images and are the basis for all region-based operations. Text hypothesis extraction is based on *maximally stable extremal regions*[30] and has been used for obtaining word hypotheses in Ref. 47. Assuming that text areas have higher contrast than background areas, document images are represented with contrast scores. Contrast scores are based on accumulated gradient magnitudes which are computed by SIFT[27] contrast normalization scores. The contrast scores are thresholded at multiple values. The text hypotheses are given by the *extremal regions* in the resulting extremal regions tree.[30] The number of thresholds has to be high enough in order to capture text in low-contrast document image regions. In the following, we use 12 thresholds which are evenly spaced over the interval of minimum and maximum contrast score values in a document image. The experimental evaluation in Ref. 43 showed that more thresholds did not yield any significant improvements.

For generating line hypotheses, each text hypothesis defines a search context in the document image. Horizontally, the search context includes the entire document width. Vertically, the search context encloses the upper and lower bound of the active text hypothesis. In the current search context, the upper bound of all generated line hypotheses is the upper bound of the search context. Given the upper bounds, the lower bounds of all text hypotheses within the search context define the line hypotheses. Line hypotheses are grouped according to line height. They are the basis for mixture component indexing and allow for pruning patches in the patch-based decoding framework during retrieval. Line hypotheses represent

alternatives to each other and do not represent a line segmentation.

Whitespace hypotheses indicate document regions that are mostly located to the left and to the right of words. They are required in order to estimate whitespace HMMs. The most important assumption for extracting whitespace regions is that the text-hypothesis bounding-boxes are unlikely to contain whitespace. In contrast, the document regions to the left and to the right of text hypotheses are more likely to contain whitespace. This is modeled in a per-pixel voting scheme. An accumulator matrix is initialized with zeros and each matrix element corresponds to a pixel in the document image. Afterwards, each text hypothesis votes against its inner bounding box area and for the document image regions to the left and to the right. Bounding boxes are obtained after thresholding the accumulator matrix and performing a connected component analysis. Details on extracting line hypothesis and whitespace hypothesis can be found in Ref. 43.

3.2. *Document region representation*

Document image regions are represented with sequences of BoF vectors, cf. Ref. 32, 45. Regions are either given by the query word image bounding-box or are based on hypotheses, i. e., whitespace hypotheses and line hypotheses. SIFT descriptors[27] are quantized with respect to a visual vocabulary and localized by their center points. Based on the dense grid of quantized image descriptors, the sequence of BoF vectors is obtained by sliding a frame over the document region in writing direction. The frame is moved over all grid columns such that it covers exactly one column at each position. Fig. 2 illustrates the representation of a document image region with a sequence of BoF vectors. The BoF representation at index t in a sequence is denoted as vector $\boldsymbol{x}_t$. The absolute frequency of the visual word with index $v \in \{0, \ldots, V-1\}$ in $\boldsymbol{x}_t$ is a scalar x_{tv}, i. e., $\boldsymbol{x}_t = (x_{t0}, \ldots, x_{t,V-1})^\top$ and V is the size of the visual vocabulary.

The large number of descriptors leads to a large computational effort when computing the visual vocabulary with Lloyd's algorithm, cf. Ref. 14. For this reason, initial centroids are computed with Lloyd's algorithm which is applied on a small set of randomly sampled descriptors. Afterwards, MacQueen's algorithm Ref. 14 clusters the entire set of descriptors.

3.3. *Model integration*

BoF sequences are modeled as observations in the statistical HMM process with a probabilistic mixture model. For this purpose, a mixture model $\Theta = \{(c_k, \Theta_k) \,|\, 0 \leq k < M\}$ is defined by M mixture components with parameters Θ_k and their mixture weights c_k. The weights are prior component probabilities $c_k = p(\mathcal{M} = k \,|\, \Theta)$. M is a meta parameter and $\mathcal{M}$ is a random variable that represents a discrete probability distribution over the event space $\Omega_{\mathcal{M}} = \{0, \ldots, M-1\}$. Eq. 1 defines the mixture model accordingly.

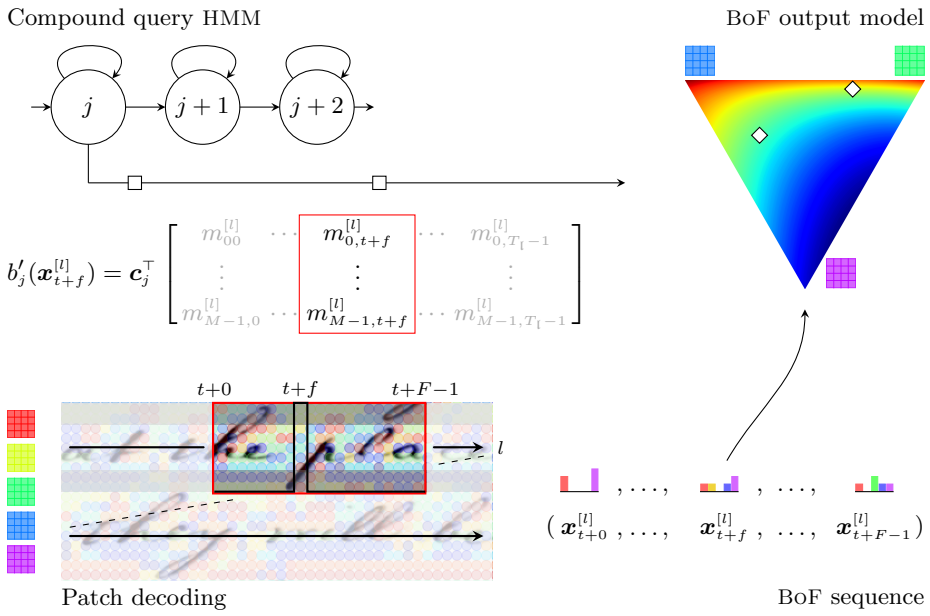


Figure 2: Patch-based decoding. Each patch is represented with a sequence of BoF vectors. The probabilistic BoF mixture model is indicated by a visual word simplex with three visual words. Each point in the simplex corresponds to a specific BoF vector. The probability mass distribution within the simplex is indicated with blue to red colors. The mixture component posteriors $\boldsymbol{m}_t^{[l]}$ for the BoF vectors in each line hypothesis are stored in a look-up table.

$$p(\boldsymbol{x}_t \,|\, \Theta) = \sum_{k=0}^{M-1} p\left(\mathcal{M} = k \,|\, \Theta\right)\, p\left(\boldsymbol{x}_t \,|\, \mathcal{M} = k, \Theta\right) \tag{1}$$

The HMM integration of the BoF output models follows the standard approach for semi-continuous HMMs, cf. Ref. 14. Instead of using the mixture component likelihoods $p(\boldsymbol{x}_t \,|\, \mathcal{M}_t = k, \lambda)$ directly, the likelihoods are replaced with approximations of mixture component posteriors $p(\mathcal{M}_t = k \,|\, \boldsymbol{x}_t, \lambda)$. Under the assumption that the distribution of mixture component priors $p(\mathcal{M} = k \,|\, \Theta)$ is uniform, Eq. 2 defines an approximation of mixture component posteriors.[14]

$$p(\mathcal{M}_t = k \,|\, \boldsymbol{x}_t, \lambda) \approx \frac{p(\boldsymbol{x}_t \,|\, \mathcal{M}_t = k, \lambda)}{\sum_{l=0}^{M-1} p(\boldsymbol{x}_t \,|\, \mathcal{M}_t = l, \lambda)} \tag{2}$$

By using component posteriors, likelihoods are rescaled, thus, normalizing their dynamic range. This has advantages when multiplying many and potentially very small values.[14] Due to the different dynamic ranges of different output models, this normalization also simplifies the integration of alternative models.

In order to integrate the output model in the semi-continuous HMM λ, the random variable $\mathcal{S}_t$ represents a discrete probability distribution over S HMM states at time t. State-dependent mixture component prior probabilities are denoted as $c_{jk} = p(\mathcal{M}_t = k|\mathcal{S}_t = j, \lambda)$. Component posterior probabilities are independent of the HMM states and are denoted as $m_{tk} = p(\mathcal{M}_t = k \,|\, \boldsymbol{x}_t, \lambda)$. Since mixture component posteriors are computed for line hypotheses, it is important to refer to a specific line hypothesis $l \in \Lambda$. Λ is the set of line position indices that have been obtained for a line height in a document image. For this purpose, component posteriors are referred to as $m_{tk}^{[l]}$. The BoF vectors that are extracted from line hypothesis l are referred to as $\boldsymbol{x}_t^{[l]}$ for all $t \in \{0, \ldots, T_l - 1\}$. T_l is the number of BoF vectors in the line hypotheses of a document image. Eq. 3 defines the output probability for BoF vector $\boldsymbol{x}_t^{[l]}$ in state j in terms of c_{jk} and $m_{tk}^{[l]}$.

$$b'_j(\boldsymbol{x}_t^{[l]}) = \sum_{k=0}^{M-1} c_{jk}\, m_{tk}^{[l]} \tag{3}$$

For searching words efficiently, it is essential to avoid computations at query time. In the given scenario, the evaluation of the BoF output model can be precomputed for all line hypotheses. This is a considerable advantage since the output model evaluation takes a large amount of time in the entire HMM decoding process.[14] Look-up tables store component posteriors $\boldsymbol{m}_t^{[l]} = (m_{t0}^{[l]}, \ldots, m_{t,M-1}^{[l]})^\top$ with $m_{tk}^{[l]} > \epsilon_{\text{low}}$ for all line hypotheses $l \in \Lambda$ and all $k \in \Omega_{\mathcal{M}}$. This is feasible due to the use of sparse representations that only store sufficiently large posterior probabilities ($\epsilon_{\text{low}} = 10^{-12}$).

3.4. *Bag-of-Features Output Models*

The typical number of non-zero visual words in a BoF vector is very small in comparison to the typically large number of visual words in the visual vocabulary.[44] Thus, BoF vectors are very high dimensional and extremely sparse. Due to these special characteristics of BoF representations in this word spotting scenario, modeling BoF with a GMM directly is infeasible. Empirically, this has been confirmed for handwritten word recognition with semi-continuous HMMs.[48] Mixture component distributions that allow for modeling BoF directly, will be discussed in the following.

3.4.1. *Von Mises-Fisher*

The *von Mises-Fisher* (vMF) distribution allows for probabilistic modeling of directional data.[7] Its use is inspired by the wide use of cosine similarity for matching BoF representations. The distribution models the generation of BoF vectors on the unit sphere, i. e., $\boldsymbol{x}_t \in \mathbb{R}^V$, $\|\boldsymbol{x}_t\|_2 = 1$, and has properties that are similar to a multivariate Gaussian distribution.[7] The vMF probability density function in Eq. 4 is defined by a mean direction $\boldsymbol{\mu} \in \mathbb{R}^V$, $\|\boldsymbol{\mu}\|_2 = 1$, a concentration parameter $\kappa \in \mathbb{R}_{\geq 0}$ and for dimensionality $V \geq 2$.

$$p(\boldsymbol{x}_t \,|\, \boldsymbol{\mu}, \kappa) = \frac{\kappa^{z-1}}{(2\pi)^z I_{z-1}(\kappa)} e^{\kappa \boldsymbol{\mu}^\top \boldsymbol{x}_t}, \quad z = \frac{V}{2} \tag{4}$$

The normalization factor includes $I_{z-1}(\cdot)$, i. e., the modified Bessel function of the first kind and the higher order $z-1$. $I_{z-1}(\kappa)$ is approximated as suggested in Ref. 13. The model estimation follows the *expectation maximization* (EM) algorithm presented in Ref. 7.

3.4.2. *Multinomial*

Modeling *bag-of-words* (BoW) representations with multinomial distributions is a standard approach for text classification.[5] For this purpose, the main idea is to model the generation of each BoW vector component individually. This way, the model resembles the computation of BoW representations. In analogy, multinomial distributions can be used for modeling BoF, e. g., for image classification.[11]

In order to model the generation of a BoF vector $\boldsymbol{x}_t \in \mathbb{N}^V_{\geq 0}$, $N = \|\boldsymbol{x}_t\|_1$ visual words are independently drawn according to the visual word probabilities $\boldsymbol{p} = (p(\mathcal{V}=0), \ldots, p(\mathcal{V}=V-1))^\top$ where V is the size of the visual vocabulary. For this purpose, $\mathcal{V}$ is a discrete random variable over the event space $\Omega_\mathcal{V} = \{0, \ldots, V-1\}$. Eq. 5 defines the probability mass function of the multinomial model in terms of parameter vector $\boldsymbol{p}$.

$$p(\boldsymbol{x}_t|\boldsymbol{p}) = \frac{\|\boldsymbol{x}_t\|_1!}{\prod_{v=0}^{V-1} x_{tv}!} \prod_{v=0}^{V-1} p(\mathcal{V}=v)^{x_{tv}} \tag{5}$$

3.4.3. *Dirichlet compound multinomial*

Inspired from short text modeling, the *Dirichlet compound multinomial* (DCM) distribution can be considered in order to model sparse BoF vectors, cf. Ref. 28, 13. This is possible by using the conjugate prior of the multinomial distribution, i. e., the Dirichlet distribution, cf. Ref. 9 . The Dirichlet distribution can model the generation of multinomial parameters $\boldsymbol{p}$. Since the Dirichlet distribution is continuous, it is not suitable for modeling discrete data directly.[28] The generation of sparse BoF vectors can be modeled when using the Dirichlet distribution and the multinomial distribution in a compound distribution. By integrating over all visual word probability vectors $\boldsymbol{p}$, the Dirichlet distribution models plausible visual word configurations.

The DCM distribution requires a large number of evaluations of the Gamma function Γ which can be considered computationally expensive. Thus, the application of the DCM is very time consuming if M and V are large.[13] For this reason, an approximation of the DCM is proposed in Ref. 13 that is valid for sparse BoW representations. It is referred to as EDCM distribution because it belongs to the

exponential family. The resulting probability distribution is presented in Eq. 6. A detailed derivation can be found in Ref. 43.

$$p(\boldsymbol{x}_t \,|\, \boldsymbol{\beta}) = \frac{\|\boldsymbol{x}_t\|_1!}{\prod_{v:x_{tv}\geq 1} x_{tv}} \frac{\Gamma(\|\boldsymbol{\beta}\|_1)}{\Gamma(\|\boldsymbol{\beta}\|_1 + \|\boldsymbol{x}_t\|_1)} \prod_{v:x_{tv}\geq 1} \beta_v \tag{6}$$

Due to the sparsity, the characteristics of the BoF vectors in the proposed method are very similar to the characteristics of the BoW vectors considered in Ref. 13. The EDCM mixture model estimation closely follows the EM algorithm described in Ref. 13.

3.4.4. *Visual words*

In contrast to the previously discussed output models, the *visual words* model follows a different approach by modeling the BoF vectors in terms of individual visual word occurrences. For this purpose, the mixture model in Eq. 1 is constrained to have exactly the same number of mixture components as visual words. Thus, mixture components directly refer to visual words. Component parameters Θ_v only encode the visual word index v. The actual observations, i. e., the occurrences of visual words in BoF vectors $\boldsymbol{x}_t$ are modeled probabilistically. Let $\boldsymbol{\mathcal{X}}_t$ be a multivariate random variable that represents independently distributed discrete random variables $(\mathcal{X}_{t0}, \ldots, \mathcal{X}_{t,V-1})^\top$ over the same event space $\Omega_{\mathcal{X}} = \{0, \ldots, N\}$. The events correspond to absolute visual word frequencies x_{tv}. Adapting the mixture model in Eq. 1, $p(\boldsymbol{x}_t \,|\, \Theta)$ can be expressed by marginalizing over visual words.

$$p(\boldsymbol{x}_t \,|\, \Theta) = \sum_{v=0}^{V-1} p(\mathcal{V} = v \,|\, \Theta)\, p(\boldsymbol{x}_t \,|\, \mathcal{V} = v, \Theta) \tag{7}$$

In order to model individual visual word occurrences, the likelihoods for vector $\boldsymbol{x}_t$ are replaced in Eq. 7. For this purpose, $\boldsymbol{x}_t$ defines a distribution over visual words. Based on $\boldsymbol{x}_t$, $p(\mathcal{X}_{tv} > 0 \,|\, \boldsymbol{x}_t)$ is the probability that the absolute frequency of visual word v in BoF $\boldsymbol{x}_t$ is greater than zero. In Eq. 8, the *logical disjunction* is considered in order to model the probability for any visual word configuration in $\boldsymbol{x}_t$ under model Θ.

$$p\left(\bigvee_{\mathcal{X}_{tv}\in\boldsymbol{\mathcal{X}}_t} \mathcal{X}_{tv} > 0 \,\middle|\, \boldsymbol{x}_t, \Theta\right) = \sum_{v=0}^{V-1} p(\mathcal{V} = v \,|\, \Theta)\, p(\mathcal{X}_{tv} > 0 \,|\, \mathcal{V} = v, \boldsymbol{x}_t) \tag{8}$$

The evaluation is based on decomposing the joint probability $p(\mathcal{X}_{tv} > 0, \mathcal{V} = v \,|\, \boldsymbol{x}_t, \Theta)$ in components $p(\mathcal{V} = v \,|\, \Theta)$ and $p(\mathcal{X}_{tv} > 0 \,|\, \mathcal{V} = v, \boldsymbol{x}_t)$. As a result, $p(\mathcal{V} = v \,|\, \Theta)$ are model parameters which are estimated from sample data. Component probabilities $p(\mathcal{X}_{tv} > 0 \,|\, \mathcal{V} = v, \boldsymbol{x}_t)$ are directly computed from the observed BoF vectors $\boldsymbol{x}_t$. For this reason, $\mathcal{X}_{tv}$ is conditioned on $\boldsymbol{x}_t$ but not on Θ. The observed

BoF vector can, therefore, be seen as parameterization of the discrete distributions represented by $\boldsymbol{\mathcal{X}}_t$. Based on $\boldsymbol{x}_t$, probability $p(\mathcal{X}_{tv} > 0 \,|\, \mathcal{V} = v, \boldsymbol{x}_t)$ is given by the relative visual word frequency for visual word v in $\boldsymbol{x}_t$.

$$p(\mathcal{X}_{tv} > 0 \,|\, \mathcal{V}_t = v, \boldsymbol{x}_t) = \frac{x_{tv}}{\|\boldsymbol{x}_t\|_1} \tag{9}$$

Eq. 9 can be considered as a soft visual-word observation model. Instead of modeling a single observation (with probability one), the probability mass is distributed to all visual words that have been observed in frame t. The visual word mixture model can be considered as a pseudo-discrete, hierarchical model with two stages. The first stage generates a visual word index. The second stage generates the binary BoF vector component that corresponds to the visual word that was generated in the first stage. Therefore, the generative process models the generation of a single visual word and not a *bag-of-visual-words*.

Modeling a *disjunction* of visual words instead of a joint probability, leads to very good generalization capabilities for query-by-example word spotting with no annotated training material but the query word image. This is due to the abstraction from the actual BoF vectors. By interpreting BoF vectors as distributions, probabilistic similarity with respect to the model's visual word distribution is computed by cross-correlation.

3.5. *Query modeling*

The query model is a compound HMM that consists of models for the *query word* and the query word context. The query word is provided by-example. The context consists of a *background* model and *whitespace* models. The background model is based on the mixture component prior distribution. The query word model and the whitespace models are estimated from annotated document regions. Model estimation follows the standard approach for HMMs, cf. Ref. 14. This excludes the HMM output model which is independent of the query word due to the semi-continuous property.

For representing the query word image with an HMM, the sequence of $T_\mathfrak{q}$ BoF vectors is extracted. Each BoF vector is represented in terms of the BoF output model with mixture component posteriors. The number of states S of the query word HMM is obtained as a percentage of the number of BoF vectors $T_\mathfrak{q}$. The HMM uses a linear topology, thus, allowing for transitions to the active state and to the next state, cf. Ref. 14. For initializing the query word HMM, the BoF vectors are linearly aligned with the states.[14] For this purpose, the state sequence $\boldsymbol{S} = (s_0, \ldots, s_{T_\mathfrak{q}-1})$ denotes the initial alignment such that $s_t \in \Omega_\mathcal{S}$ and $0 \leq t < T_\mathfrak{q}$. The state at index t is defined in Eq. 10.

$$s_t = \left\lfloor \frac{t}{T_q - 1} \left(S - 1\right) + 0.5 \right\rfloor \tag{10}$$

For each state, initial model parameter estimates are based on the feature vectors that have been aligned with the corresponding state. Essentially, this corresponds to Viterbi training, cf. Ref. 14. The basic idea for Viterbi training is to use the *optimal* output probability $p(\boldsymbol{O}, \boldsymbol{S}^* \,|\, \lambda)$ as optimization criterion instead of the *total* output probability $p(\boldsymbol{O} \,|\, \lambda)$ in the Baum-Welch algorithm. For this purpose, an optimal state-based alignment $\boldsymbol{S}^*$ is computed with the Viterbi algorithm for a training vector sequence $\boldsymbol{O}$. For initialization, a linear alignment is provided according to Eq. 10.

State-dependent mixture component priors are initialized by averaging the mixture component posteriors m_{tk} that have been observed in the corresponding states, see Eq. 11. For this purpose, $\delta : \Omega_s \times \Omega_s \to \{0, 1\}$ serves as a function that indicates whether frame t from the training vector sequence has been aligned with state j that is considered for parameter initialization.

$$\hat{c}_{jk} = \frac{\sum_{t=0}^{T_\mathrm{q}-1} \delta(s_t, j)\, m_{tk}}{\sum_{t=0}^{T_\mathrm{q}-1} \delta(s_t, j)}, \quad \delta(s_t, j) = \begin{cases} 1 & : s_t = j \\ 0 & : s_t \neq j \end{cases} \tag{11}$$

The initialization of the transition probabilities follows the standard approach in the Viterbi training.[14] The initialized model can be refined with Baum-Welch training. However, it is an interesting question if this is suitable with a single example, cf. Sec. 4.6.

The whitespace HMMs are estimated based on whitespace regions that have been obtained for the left-side as well as the right-side context of text. Since whitespace regions have a small width by design, the HMMs consist of a single state each. Otherwise, the model configuration and estimation follows the procedure described for the query word HMM.

The background model represents arbitrary document image content. It is a single-state HMM which models the distribution of mixture components in the document collection, i. e., the background.[41] It is given as distribution of mixture component priors $p(\mathcal{M} = k \,|\, \Theta)$ in the BoF output model Θ.

3.6. *Retrieval*

Retrieval is performed in a patch-based framework. For this purpose, patches are sampled from the document images. Each patch receives a score that indicates similarity to the query. In order to reduce the computational complexity of evaluating a large number of patches with the Viterbi algorithm, see Sec. 3.6.2, a coarse and a fine analysis stage is proposed. In the coarse stage, the BoF-HMM is evaluated with a voting scheme that mimics Viterbi decoding, see Sec. 3.6.3. This allows for a trade-off between retrieval accuracy and efficiency.

3.6.1. *Patch sampling*

Patches are sampled from document images in a regular grid. The patch size is given by the size of the query word image. In order to limit the huge amount of patches, the grid resolution is dynamically adapted to the patch size in horizontal and vertical direction. Thus, more patches are extracted for smaller query words than for larger query words. In order to align the patches with the visual word grid, the patch size is quantized to a multiple of the visual-word grid sampling step. The patch height is quantized with respect to heights of line hypotheses that have been extracted in the document image. Based on the quantized patch size, the patch sampling steps $(\mathfrak{r}_\mathtt{x}, \mathfrak{r}_\mathtt{y})$ are defined. The patch sampling steps are based on scaling the patch size by $\upsilon^{-1} \in \mathbb{Q}$ with $\upsilon^{-1} \leq \frac{1}{4}$. The minimum sampling step in either dimension is the visual-word grid sampling step $\mathfrak{g}$. Since the number of patches is relative to the patch size, a sampling rate of $\upsilon = 8$ has been sufficient for all datasets considered in the experimental evaluation in Sec. 4, cf. Ref. 43.

3.6.2. *Viterbi decoding*

The basic idea for searching document images with the query HMM is to represent patches with sequences of BoF vectors and compute the probability for generating the sequences with the HMM. In order to improve retrieval speed, the evaluation of the BoF output model is precomputed and stored in look-up tables for the line hypotheses with the height that corresponds to the patch height. Thus, within the regular grid of patches only those patches will be processed that lie within line hypotheses that are relevant for the patch height. In the Viterbi algorithm, the output probabilities $b'_j\big(\boldsymbol{x}_t^{[l]}\big) = \boldsymbol{c}_j^\top \boldsymbol{m}_t^{[l]}$ are required. For line hypothesis l, the mixture component posteriors $\boldsymbol{m}_t^{[l]}$ can be obtained from the mixture component index. The state-dependent vector of mixture component priors $\boldsymbol{c}_j$ is given by the model.

A patch representation is defined as a *sub*sequence of a line representation. Provided that line l is represented with vectors $\boldsymbol{x}_t^{[l]}$ with $0 \leq t < T_\mathfrak{l}$, then a patch with F BoF vectors is defined with sub-indices $t+f$ with $0 \leq t \leq T_\mathfrak{l} - F$ where $F \leq T_\mathfrak{l}$ and $0 \leq f < F$. Based on offset t and patch sequence index f, the corresponding mixture component posterior vector $\boldsymbol{m}_{t+f}^{[l]}$ is retrieved from the look-up table as visualized in Fig. 2.

For a given patch size, patches are arranged in a regular $Q \times R$ grid where Q is the number of rows and R is the number of columns. Patches can be addressed with tuple $(q, r) \in \{0, \ldots, Q-1\} \times \{0, \ldots, R-1\}$. Corresponding lines can be addressed based on the patch row indices $l \in \Delta$ with $\Delta = \Lambda \cap \{0, \ldots, Q-1\}$, Λ is the set of relevant line position indices. For a patch column index r, the BoF vector offset t in the line hypothesis is computed with function $g : \{0, \ldots, R-1\} \rightarrow \{0, \ldots, T_\mathfrak{l} - F\}$ based on the patch sampling step $\mathfrak{r}_\mathtt{x}$ and the visual-word grid sampling step $\mathfrak{g}$.

Eq. 12 defines the sequence of F BoF vectors $\boldsymbol{O}_r^{[l]}$ for patch (l, r).

$$\boldsymbol{O}_r^{[l]} = \left(\boldsymbol{x}_{g(r)+0}^{[l]}, \ldots, \boldsymbol{x}_{g(r)+F-1}^{[l]}\right), \quad g(r) = r\,\frac{\mathfrak{r}_\mathfrak{x}}{\mathfrak{g}} \tag{12}$$

The Viterbi algorithm computes $p(\boldsymbol{O}_r^{[l]}, \boldsymbol{S}_r^{[l]*} \mid \lambda)$, i. e., the optimal output probability for BoF vector sequence $\boldsymbol{O}_r^{[l]}$ and the optimal state sequence $\boldsymbol{S}_r^{[l]*}$. For this purpose, the beam search algorithm, cf. Ref. 14, is applied with a negative-logarithmic beam offset of 200 and a floor probability, cf. Ref. 14, of $\epsilon_{\text{floor}} = 10^{-5}$.

Based on the optimal alignment $\boldsymbol{S}_r^{[l]*}$, the optimal output probability $p\big(\boldsymbol{O}_r^{[l]}, \boldsymbol{S}_r^{[l]*} \mid \lambda\big)$ can be expressed in terms of the individual alignments with the states of the models in the compound query HMM. Eq. 13 defines subsequences of BoF vectors. The corresponding subsequences of states are defined in Eq. 13. The notation for a sequence at row index l and column index r, e. g., $\boldsymbol{O}_r^{[l]}$, is extended such that the column index r can be conditioned on the subsequence for the corresponding models of the compound HMM. The subsequence that corresponds to models which are representing the left side context in a patch, i. e., left side background and left side whitespace, is denoted as $\mathfrak{u}$. In analogy, the right side context is denoted as $\mathfrak{v}$. The subsequence that has been aligned with the query word HMM is denoted as $\mathfrak{q}$. The definition of the subsequences is based on the estimated frame offset $\hat{f}_\mathfrak{q}$ for the query word model and the estimated number of frames $\widehat{F}_\mathfrak{q}$ that have been aligned with the query word model within a patch.

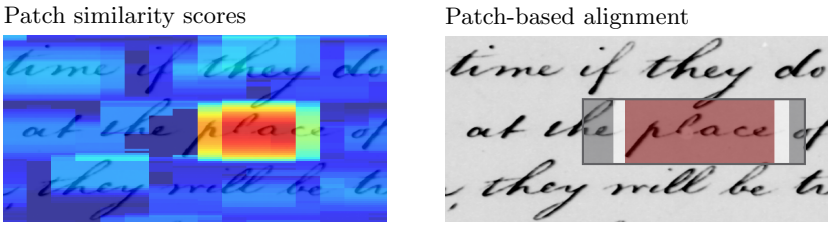
$$\begin{aligned}
\boldsymbol{O}_{r\,|\,\mathfrak{u}}^{[l]} &= \left(\boldsymbol{x}_{g(r)+0}^{[l]}, \ldots, \boldsymbol{x}_{g(r)+\hat{f}_\mathfrak{q}-1}^{[l]}\right), & \boldsymbol{S}_{r\,|\,\mathfrak{u}}^{[l]*} &= \left(s_{g(r)+0}^{[l]*}, \ldots, s_{g(r)+\hat{f}_\mathfrak{q}-1}^{[l]*}\right) \\
\boldsymbol{O}_{r\,|\,\mathfrak{q}}^{[l]} &= \left(\boldsymbol{x}_{g(r)+\hat{f}_\mathfrak{q}}^{[l]}, \ldots, \boldsymbol{x}_{g(r)+\hat{f}_\mathfrak{q}+\widehat{F}_\mathfrak{q}-1}^{[l]}\right), & \boldsymbol{S}_{r\,|\,\mathfrak{q}}^{[l]*} &= \left(s_{g(r)+\hat{f}_\mathfrak{q}}^{[l]*}, \ldots, s_{g(r)+\hat{f}_\mathfrak{q}+\widehat{F}_\mathfrak{q}-1}^{[l]*}\right) \\
\boldsymbol{O}_{r\,|\,\mathfrak{v}}^{[l]} &= \left(\boldsymbol{x}_{g(r)+\hat{f}_\mathfrak{q}+\widehat{F}_\mathfrak{q}}^{[l]}, \ldots, \boldsymbol{x}_{g(r)+F-1}^{[l]}\right), & \boldsymbol{S}_{r\,|\,\mathfrak{v}}^{[l]*} &= \left(s_{g(r)+\hat{f}_\mathfrak{q}+\widehat{F}_\mathfrak{q}}^{[l]*}, \ldots, s_{g(r)+F-1}^{[l]*}\right)
\end{aligned} \tag{13}$$

Due to the independence assumptions in the HMM, $p\big(\boldsymbol{O}_r^{[l]}, \boldsymbol{S}_r^{[l]*} \mid \lambda\big)$ can be expressed in terms of the partial alignments of the BoF vector sequence with the states in the compound query HMM as shown in Eq. 14. The partial output probability for the query word model is obtained in Eq. 15. It is equivalent to Eq. 14. Therefore, the ratio in Eq. 15 does not correspond to a filler-score normalization because the denominator does not represent the entire observation sequence $\boldsymbol{O}_r^{[l]}$.

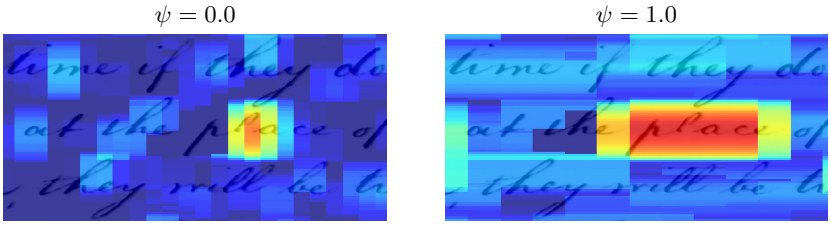
$$p\big(\boldsymbol{O}_r^{[l]}, \boldsymbol{S}_r^{[l]*} \mid \lambda\big) = p\big(\boldsymbol{O}_{r\,|\,\mathfrak{u}}^{[l]}, \boldsymbol{S}_{r\,|\,\mathfrak{u}}^{[l]*} \mid \lambda\big)\, p\big(\boldsymbol{O}_{r\,|\,\mathfrak{q}}^{[l]}, \boldsymbol{S}_{r\,|\,\mathfrak{q}}^{[l]*} \mid \lambda\big)\, p\big(\boldsymbol{O}_{r\,|\,\mathfrak{v}}^{[l]}, \boldsymbol{S}_{r\,|\,\mathfrak{v}}^{[l]*} \mid \lambda\big) \tag{14}$$

$$\Leftrightarrow p\big(\boldsymbol{O}_{r\,|\,\mathfrak{q}}^{[l]}, \boldsymbol{S}_{r\,|\,\mathfrak{q}}^{[l]*} \mid \lambda\big) = \frac{p\big(\boldsymbol{O}_r^{[l]}, \boldsymbol{S}_r^{[l]*} \mid \lambda\big)}{p\big(\boldsymbol{O}_{r\,|\,\mathfrak{u}}^{[l]}, \boldsymbol{S}_{r\,|\,\mathfrak{u}}^{[l]*} \mid \lambda\big)\, p\big(\boldsymbol{O}_{r\,|\,\mathfrak{v}}^{[l]}, \boldsymbol{S}_{r\,|\,\mathfrak{v}}^{[l]*} \mid \lambda\big)} \tag{15}$$

Since the partial output probabilities depend on the number of frames $\widehat{F}_\mathfrak{q}$ that have been aligned with the query word model, the patch scores are obtained after length-normalization, see Eq. 16.



(a) Patch-based retrieval for query `place`



(b) Patch width expansion

Figure 3: Patch sampling rate and width expansion.

$$\nu_{qr} = \begin{cases} \max\left(\sqrt[\widehat{F}_{\mathfrak{q}}]{p\left(\boldsymbol{O}^{[q]}_{r\,|\,\mathfrak{q}}, \boldsymbol{S}^{[q]*}_{r\,|\,\mathfrak{q}} \,|\, \lambda\right)}, \epsilon_{\text{floor}} \right) & : q \in \Lambda \\ \epsilon_{\text{floor}} & : q \notin \Lambda \end{cases} \tag{16}$$

Fig. 3a (left) visualizes patch scores for a section of a document image in the logarithmic domain. Due to the high overlap of neighboring patches, the score distribution is relatively smooth. Consequently, the similarity scores start raising as soon as patches start to overlap with document image regions that are visually similar to the query. The similarity scores reach local maxima for the patches that are centered over document image regions that are most similar to the query. This effect is exploited for retrieving relevant patches with *non-maximum suppression*. Fig. 3a (right) indicates the alignment of the patch with background, whitespace and query word models.

In order to improve the flexibility with respect to occurrences of the query that are larger than the patch size, the patch width $\mathfrak{p}_{\mathtt{w}}$ can be increased according to $\mathfrak{p}_{\mathtt{w}} \leftarrow \mathfrak{p}_{\mathtt{w}} + \psi\mathfrak{p}_{\mathtt{w}}$ with meta parameter $\psi \in \mathbb{R}_{\geq 0}$. The improved word size flexibility comes at the cost of reduced specificity of the scores. An impression of the effect on the patch-similarity scores for two different patch width expansion factors is given in Fig. 3b. The influence on the specificity can be observed for high similarity scores as well as for low similarity scores. The proposed method increases the patch size by 50%, i. e., $\psi = 0.5$. The corresponding visualization can be found in Fig. 3a (left).

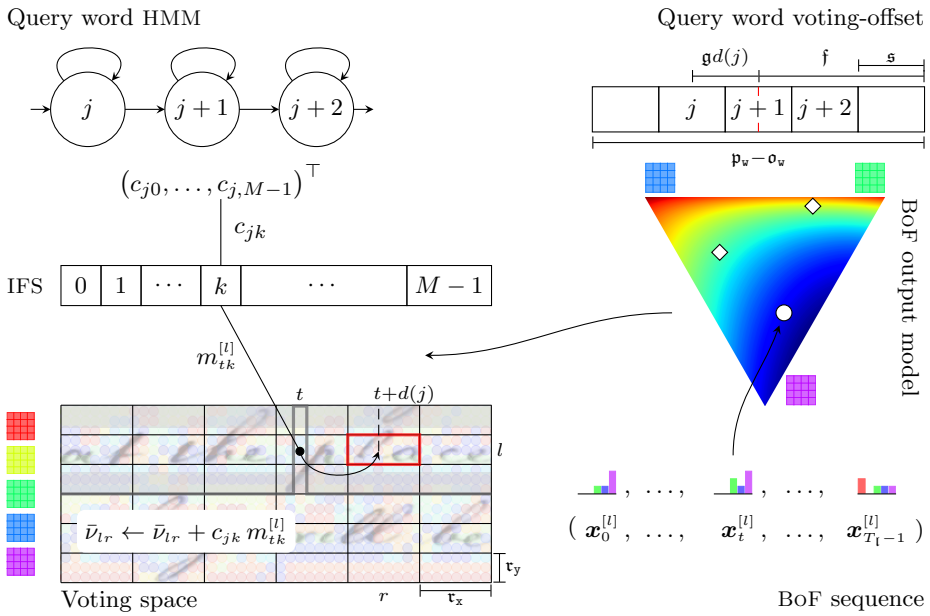


Figure 4: Mixture component voting. The figure indicates the voting process for mixture component posterior $m_{tk}^{[l]}$ of BoF vector $\boldsymbol{x}_t^{[l]}$ and HMM state j. The reference point $\mathfrak{f}$ is located at the center of the section that corresponds to state $j+1$. Based on the spatial location of frame t in line l, the voting mass $c_{jk}\, m_{tk}^{[l]}$ is accumulated in cell (l, r) which is highlighted in red.

3.6.3. *Mixture Component Voting*

Computing similarity scores in a patch-based framework with the Viterbi algorithm is computationally very demanding due to the large number of patches.

Mixture component voting allows for obtaining potentially relevant patches without evaluating each patch individually. The approach is inspired by the application of the *generalized Hough transform*[6] to object detection with SIFT descriptors.[27] For this purpose, local features from the model are matched with local features from the document image. The model defines a reference point in order to obtain a relative displacement vector for each match. The matching features vote for an object hypothesis by increasing an accumulator in the Hough voting space. The voting space is represented by a regular grid of accumulator cells over the image. The cell coordinates are obtained based on the coordinates of the matching features in the document image and their displacement vectors. Fig. 4 visualizes the mixture component voting procedure.

Given the *query word* HMM, mixture components with prior probabilities c_{jk} for all $j \in \Omega_{\mathcal{S}}$ and all $k \in \Omega_{\mathcal{M}}$ are considered as local features. The features are local since they are specific to state j in the state sequence of the query word model. In the same way, mixture components from the document image are represented

with mixture component posteriors $m_{tk}^{[l]}$ for all $l \in \Delta$ with $0 \leq t < T_{\mathfrak{l}}$. These can be considered as local features as well because they are specific to line l and frame t. The key idea is to match all state-dependent prior probabilities with all posteriors in the document image for all components $k \in \Omega_{\mathcal{M}}$. Instead of a binary match indicator, matches are soft, since the voting mass is described by the joint probability for component k given BoF vector $\boldsymbol{x}_t^{[l]}$ and component k given state j in model λ. Matches can be retrieved rapidly by extending mixture component look-up tables with inverted indices. For this purpose the IFS is defined as Υ in Eq. 17. For each mixture component $k \in \Omega_{\mathcal{M}}$, Υ_k stores line-frame indices (l,t) of all frames in all line hypotheses that are represented with a non-zero posterior probability ($\epsilon_{\text{low}} = 10^{-12}$), see Eq. 18. Further, the posteriors are stored along with the indices as weights.

$$\Upsilon = \left\{ \Upsilon_k \,|\, 0 \leq k < M \right\} \tag{17}$$

$$\Upsilon_k = \left\{ (m_{tk}^{[l]}, (l,t)) \,|\, \forall (l,t) \in \Delta \times \{0, \ldots, T_{\mathfrak{l}} - 1\} : m_{tk}^{[l]} > \epsilon_{\text{low}} \right\} \tag{18}$$

Fig. 4 shows a single IFS entry for component posterior $m_{tk}^{[l]}$ of BoF vector $\boldsymbol{x}_t^{[l]}$ which has been extracted from frame t in line l.

For each match, a vector denoting the displacement with respect to a reference point in the query word model is required. The HMM states represent the visual appearance of the query word in horizontal direction. The spatial extent of each state with respect to the patch width can be approximated with $\mathfrak{s}$ for any state, see Eq. 19, where $\mathfrak{p}_{\mathfrak{w}}$ is the patch width and S is the number of HMM states. Thus, the patch is subdivided in S sections of uniform width $\mathfrak{s}$. The reference point $\mathfrak{f}$ is the horizontal patch center.

$$\mathfrak{s} = \frac{\mathfrak{p}_{\mathfrak{w}}}{S}, \quad \mathfrak{f} = \frac{\mathfrak{p}_{\mathfrak{w}}}{2} \tag{19}$$

The displacement in horizontal direction is defined in Eq. 20. Since the reference point is the patch center, the displacement is defined with respect to the section centers. The left bound of the section that corresponds to state j is given by $j\mathfrak{s}$. The horizontal section center is obtained after adding $\frac{\mathfrak{s}}{2}$. The difference with respect to $\mathfrak{f}$ results in the displacement in image coordinates such that a positive displacement is obtained for states representing the beginning of the query word and a negative displacement is obtained for states representing the end of the query word.

$$\begin{aligned} d &: \left\{0, \ldots, S-1\right\} \rightarrow \left\{ -\left\lfloor \frac{T_{\mathfrak{l}}}{2} \right\rfloor, \ldots, \left\lfloor \frac{T_{\mathfrak{l}}}{2} \right\rfloor \right\} \\ d(j) &= \left\lfloor \frac{\mathfrak{f} - \left(j\,\mathfrak{s} + \frac{\mathfrak{s}}{2}\right)}{\mathfrak{g}} \right\rfloor \end{aligned} \tag{20}$$

Function d maps a state index to a horizontal displacement in terms of frame positions by dividing by grid sampling step $\mathfrak{g}$. Thus, the co-domain is bounded

based on the number of frames $T_{\mathfrak{l}}$ in a line. The use of frame indices is convenient for the integration of the IFS and the voting space. Fig. 4 visualizes the voting offset for state j. The distance in image pixels from section center j to the reference point is denoted as $\mathfrak{g}d(j)$.

The Hough voting space over a document image is quantized in voting cells of width and height $(\mathfrak{r}_{\mathbf{x}}, \mathfrak{r}_{\mathbf{y}})$ such that the center-coordinates of the cells are aligned with the center-coordinates of the patches. Since the cell size equals to the patch sampling step, the cells are non-overlapping and there exists exactly one cell per patch. This correspondence allows for obtaining similarity scores for patches based on the voting scores of the cells. In the same way as Viterbi-based patch scores are stored in $Q \times R$ matrix $[\nu_{qr}]$, cf. Eq. 16, votes are accumulated in matrix $[\bar{\nu}_{qr}]$. The accumulator values are initialized to ϵ_{floor}, see Eq. 21. The floor probability $\epsilon_{\text{floor}} = 10^{-5}$ is used in analogy to Eq. 16. Fig. 4 shows the voting space as a regular grid of cells over the document image.

The voting algorithm can be defined for non-zero state-dependent prior probabilities $c_{jk} > \epsilon_{\text{low}}$ for all $j \in \Omega_{\mathcal{S}}$ and all $k \in \Omega_{\mathcal{M}}$ from the query word HMM. All corresponding non-zero mixture posteriors in a document image, i. e., $\forall \left(m_{tk}^{[l]}, (l,t)\right) \in \Upsilon_k$, are retrieved from the IFS. Each element votes for a patch by adding the voting mass $c_{jk}\, m_{tk}^{[l]}$ to the accumulator in the voting space. The corresponding voting-cell index is determined based on line-frame index (l,t) and voting offset $d(j)$, see Eq. 22. Index l is a valid patch row index since $l \in \Delta \subseteq \{0, \ldots, Q-1\}$. Frame index t is mapped to a valid patch column index by function $g^{\star}$, cf. Eq. 23.

$$\bar{\nu}_{qr} = \epsilon_{\text{floor}} \quad \forall (q,r) \in \{0, \ldots, Q-1\} \times \{0, \ldots, R-1\} \tag{21}$$

$$\begin{aligned} &\forall (j,k) \in \Omega_{\mathcal{S}} \times \Omega_{\mathcal{M}} : c_{jk} > \epsilon_{\text{low}},\ \forall \left(m_{tk}^{[l]}, (l,t)\right) \in \Upsilon_k\ . \\ &\qquad \bar{\nu}_{l, g^{\star}(t+d(j))} \leftarrow \bar{\nu}_{l, g^{\star}(t+d(j))} + c_{jk}\, m_{tk}^{[l]} \end{aligned} \tag{22}$$

Function $g^{\star}$ is related to the inverse of function g, cf. Eq. 12. The difference lies in the domain definitions which is important for handling document boundaries. Further, $g^{\star}$ rounds down in order to map a range of frame indices to a single patch index. It has to be noted that an accumulator is only increased if $t + d(j)$ is in the domain of function $g^{\star}$.

$$\begin{aligned} g^{\star} &: \left\{0, \ldots, T_{\mathfrak{l}} - \left\lceil \frac{F}{2} \right\rceil - 1\right\} \rightarrow \left\{0, \ldots, R-1\right\} \\ g^{\star}(t) &= \left\lfloor t\, \frac{\mathfrak{g}}{\mathfrak{r}_{\mathbf{x}}} \right\rfloor \end{aligned} \tag{23}$$

The mixture component algorithm can be interpreted as a coarse evaluation of output probabilities $b'_j\left(\boldsymbol{x}_t^{[l]}\right)$. For each state and each BoF vector, the output probabilities are computed component-wise and added to the cell that corresponds to line $l \in \Delta$ and frame $t+d(j) \in \{0, \ldots, T_{\mathfrak{l}}-1\}$. Assuming that the query word HMM has just a single state, the voting scores correspond to the HMM output probabilities.

For multiple states, the sums are extended to output probabilities for frames in the horizontal neighborhood that vote for the same cell. The voting offset reflects the model structure due to its dependence on state j. The output probability for any BoF vector in a cell, i. e., vectors that are consistent with the sequential model structure is considered. Similarity scores $[\bar{\nu}_{qr}]$ are, therefore, sensitive to individual, high output probabilities. This leads to high recall at the cost of a high false positive rate.

Fig. 4 indicates the voting process for state-dependent component prior c_{jk} and component posterior $m_{tk}^{[l]}$ which is obtained through the IFS. The voting mass is accumulated in cell (l, r) that contains frame $t + d(j)$.

3.6.4. *Two-stage integration*

The two-stage integration is achieved by obtaining potentially relevant patches with mixture component voting and re-ranking these patches with the Viterbi algorithm. Since mixture component voting is sensitive to document image regions that are partially similar to the query word, an extension to patch-based *non-maximum suppression* (NMS) is proposed. In order to compromise between the number of patches and re-ranking mostly all relevant patches, similarity scores $[\bar{\nu}_{qr}]$ are smoothed with a discrete anisotropic Gaussian filter before applying NMS. The size of the Gaussian filter corresponds to the patch size and the size of the NMS filter allows up to 50% patch overlap. Strong local optima in close proximity to each other can be preserved this way.

Further improvements can be achieved if more than just a single locally optimal patch is re-ranked with the Viterbi algorithm. For this purpose, the locally optimal patches are indicated as ones in a binary $Q \times R$ matrix. Using a binary dilation operation, the ones can be extended into the local neighborhood according to a 3×3 structuring element, i. e., the re-ranking mask. Afterwards, NMS is applied to the re-ranked patches such that the retrieved patches do not overlap.

4. Evaluation

Segmentation-free word spotting with BoF-HMMs is evaluated on five publicly available benchmarks, see Sec. 4.2. The corresponding performance measures are introduced in Sec. 4.1. In Sec. 4.4, we compare our method to results from the literature and to a baseline system. The baseline is presented in Sec. 4.3. An important result is that the proposed method is very robust with respect to its meta parameters except for the size of the SIFT descriptors. For this reason, an automatic descriptor size estimation is presented in Sec. 4.5. This makes the proposed method directly applicable to a newly acquired dataset. Finally, a discussion on architectural design choices and meta parameters is presented in Sec. 4.6. In this regard, two out of the five benchmarks are used for meta parameter optimization. No parameter optimization has been performed for the other benchmarks. The SIFT descriptor size is estimated automatically for these benchmarks.

4.1. *Performance measures*

The quantitative analysis of a word spotting method typically measures if all occurrences of the query word are present in the retrieval list as well as the order of the relevant detections in the list. The most common measure which incorporates both of these criteria into a single value is the average precision.[5] In order to measure the performance, the relevance of these regions must be determined. Based on the *intersection over union* (IoU) an element is considered as relevant if the detected document image region overlaps with a relevant bounding box from the dataset annotations by more than a given threshold, here 50%. A bounding box annotation is relevant to the query if it is labeled with the query word.

The relevance of the detected document image region at retrieval list index k can be expressed by function $\varphi(k)$. Eq. 24 defines *average precision* and *mean average precision* over a larger set of queries Q with R denoting the total number of relevant regions for the query in the dataset.

$$\mathrm{AP} = \frac{1}{R} \sum_{k=0}^{K-1} \pi(k+1)\,\varphi(k), \quad \mathrm{mAP} = \frac{1}{Q} \sum_{q=0}^{Q-1} \mathrm{AP}_q \tag{24}$$

In the following, mAP will be the main performance measure. Some methods from the literature use a variation of mAP, the *mean average interpolated precision*, which is based on an approximation of the area under the interpolated precision-recall curve, cf. Ref. 36. The corresponding results are reported accordingly.

Since the mAP is an average case performance measure, it is unclear whether (small) differences in the average case are due to large differences in the performance of few queries or if the differences can be observed for the majority of the queries. Significance tests allow for a statistical analysis of the differences which supports the interpretation of the results. Therefore, we conduct permutation tests following the procedure described in Ref. 55. Within the following tables the best result is indicated with a bold font. Results that differ from the best result significantly (significance level 5%), are indicated with an italic font. It should be noted that no significance test can be performed with the results from the literature because the average precisions per query are unavailable.

4.2. *Datasets*

Word spotting benchmarks are defined by a set of document images, annotated document regions including occurrences of the query words and an evaluation protocol. The five benchmark datasets have been selected according to their relevance with respect to the scenario where a new dataset is explored with query-by-example word spotting. Thus, we do not use any annotated training data.

4.2.1. *George Washington letters*

The George Washington dataset is the most widely used benchmark dataset for evaluating word spotting methods, cf. Ref. 17. The document images originate from the *George Washington papers* collection at the *Library of Congress*, Washington DC, USA.[59] The document images in the word spotting benchmark come from *Letterbook 1* in *Series 2: Letterbooks*. The letterbooks contain copies of Washington's mail and have been written by George Washington and his secretaries in the 18th century. The writing style in the document images from the benchmark is very homogeneous since it has been re-copied at a later time.[59] Twenty document images along with 4,860 bounding box annotations on word level are available at the *University of Massachusetts*[a]. The protocol that will be used for evaluating segmentation-free query-by-example word spotting without annotated training material has been described in Ref. 49. The benchmark will be denoted as GW20. It is used for parameter optimization in Sec. 4.6.

4.2.2. *Jeremy Bentham manuscripts*

The Bentham manuscripts have been used in the word spotting competitions[35,39] at the *Int. Conference on Frontiers in Handwriting Recognition (ICFHR) 2014* and the *Int. Conference on Document Analysis and Recognition (ICDAR) 2015*. The document images originate from the *University College London*, UK.[26] The manuscripts contain works on law and moral philosophy and have been analyzed in the context of the project *Transcribe Bentham* where over 19,000 document images have been transcribed.[10] The writing style varies considerably. However, the writing style does not change from page to page.

The segmentation-free query-by-example benchmark that was defined for the 2014 competition[b] contains 50 document images and 290 query word images.[35] The segmentation-free query-by-example benchmark that was defined for the 2015 competition[c] contains 70 document images and 1,421 query word images.[39] The Bentham 2014 benchmark will be denoted as BT50 and is used for parameter optimization in Sec. 4.6. The Bentham 2015 benchmark will be denoted as BT70.

It is important to note that the results which are reported in the competitions are based on non-standard relevance criteria in the segmentation-free scenario in order to emphasize the detection qualities. However, in Ref. 61 the corresponding results have been re-evaluated with the standard IoU measure (50% IoU threshold). The following evaluations will mainly use the standard evaluation protocol from Ref. 61. The use of the original relevance criterion will be indicated by (*) in Tab. 2.

[a] http://ciir.cs.umass.edu/downloads/old/
[b] http://vc.ee.duth.gr/H-KWS2014/
[c] http://transcriptorium.eu/~icdar15kws/

4.2.3. *Botany and Konzilsprotokolle*

The Konzilsprotokolle and Botany benchmarks have been used in the word spotting competition[36] at the *Int. Conference on Frontiers in Handwriting Recognition (ICFHR) 2016*. Both benchmarks have been prepared in the European project *READ*.

The *Konzilsprotokolle* collection is archived at the *University of Greifswald*, Germany and contains around 18,000 documents. The document images contain notes of formal meetings that have been written in the 18th century. In contrast to all other datasets presented in Sec. 4.2, the script is German *Kurrent* instead of English *Latin*. The writing style in the document images of the benchmark can be considered as homogeneous.

The *Botany in British India* collection is hosted at the *British Library* , London, UK. The collection contains manuscripts on various botanical topics that have been written in the 19th century. The document images in the benchmark contain substantial writing style variations. Thus, the dataset characteristics are *unsuitable* for the targeted scenario where no annotated training data is available. The benchmark is included in order to demonstrate the limitations of the proposed method.

The main objective of the 2016 competition[d] is the evaluation of word spotting methods on different scripts and an analysis with respect to the required amount of training data.[36] In analogy to all other benchmarks considered, the results for the baseline method and for the BoF-HMM have been obtained without using any annotated training data.

The Konzilsprotokolle will be denoted as KP20. It contains 200 query word images and 20 document images. The Botany dataset will be be denoted as BO20. It contains 150 query word images and 20 document images. On both benchmarks, the performance measure is a variant of mAP with maximum interpolation.

4.3. *Baseline*

The baseline is based on a spatial pyramid and uses a temporal cell structure that resembles the temporal modeling structure of the BoF-HMM. For this purpose, the baseline uses the same descriptors, the same visual vocabulary, the same patch sizes and the same patch sampling steps which are used for segmentation-free word spotting with the BoF-HMM. The query word image and all patches are represented with this temporal spatial-pyramid adaptation. Similarity of the patch representations with respect to the query word representation is computed with cosine similarity. The patch score matrix is smoothed with an anisotropic Gaussian that corresponds to the size of the query word image, cf. Ref. 49. Locally most similar patches are retrieved with NMS such that the retrieved patches do not overlap.

While the patch-based framework of the baseline corresponds to the patch-based framework that is used for word spotting with BoF-HMMs, an important question

[d] `https://www.prhlt.upv.es/contests/icfhr2016-kws`

Table 1: Query-by-example results summary (mAP [%])

Method	GW20	BT50	BT70	KP20	BO20
Baseline	*70.2*	*45.6*	*30.4*	*72.9*	44.6
BoF-HMM	**75.1**	**56.4**	**39.2**	**79.1**	**47.5**
Spatial pyramid indexing[49]	61.4	–	–	–	–
Exemplar SVM[4]	59.1	–	–	–	–
Scale-space pyramid[40]	56.0	–	–	–	–
Local feature matching[61]	–	51.7	32.6	–	–
Random projections[22]	50.1	42.3	–	61.8	37.5
Inkball models[19]	–	40.9	–	–	–
Cohesive elastic matching[33]	–	39.7	–	–	–
Cohesive elastic matching[24]	–	22.1	–	–	–
Spatial pyramid matching[54]	–	–	29.3	–	–
Spatial pyramid matching[2]	–	–	11.6	–	–

is how to adapt the spatial pyramid to the characteristics of text. An important result from Ref. 3 is that a high temporal resolution, i. e., a large number of cells in horizontal direction, is important for high word spotting performance. However, this comes at the cost of high dimensional patch representations. In Ref. 49, this has been addressed with product quantization. In order to limit the number of cells, the baseline method does not use a pyramidal structure, but four consecutive, equally sized cells on a single level. The design is inspired by the BoF-HMM that also does not use a pyramidal structure but represents the temporal structure with states.

Another spatial pyramid extension which has been applied to word spotting is power normalization.[2,3] Provided that all vector components are non-negative, the key idea is to reduce the influence of large vector components with a power normalization exponent α with $0 < \alpha < 1$. Following the results in Ref. 3, the baseline method uses a power normalization exponent of $\alpha = 0.35$. An evaluation of these baseline parameters can be found in Ref. 43.

4.4. *Comparison to results from the literature*

The results for segmentation-free word spotting with BoF-HMMs are presented on five benchmark datasets, cf. Sec. 4.2. The meta parameter optimization has been performed on the GW20 and on the BT50 datasets, see Sec. 4.6. With respect to the considered scenario, it is assumed that no validation sets are available for the other datasets. Hence, the results for all benchmarks have been obtained with the same BoF-HMM meta parameters, except for the descriptor size which is individually estimated on each dataset according to Sec. 4.5.

Tab. 1 shows the quantitative comparison to related methods from the literature in the *annotation-free* scenario. The results show that the BoF-HMM outperforms all other methods in terms of word spotting performance by a large margin. None

Table 2: Comparison to query-by-example results that use annotated training data (number of labels and mAP [%])

Method	BT50		KP20		BO20	
	Labels	mAP (*)	Labels	mAP	Labels	mAP
Baseline	1	41.7	1	72.9	1	44.6
BoF-HMM	1	54.9	1	79.1	1	47.5
PHOCNet[47]	–	–	16920	91.1	21982	74.5
PHOCNet[36,53]	–	–	1850	52.2	1685	15.9
GMM-HMM[58]	8019 (lines)	71.5	–	–	–	–
CRNN[37]	8337 (lines)	87.3	–	–	–	–

Labels referes to the number of word-level annotations, unless noted otherwise.
(*) 70% intersection-over-annotation.

of the methods in Tab. 1 uses annotated training material.

Due to the small word size variability in the GW20 dataset, all related methods but one are built on patch-based frameworks with a fixed patch geometry. The comparison between the baseline and the BoF-HMM shows that patch decoding with the Viterbi algorithm offers advantages over a static patch retrieval framework. The difference is *significant*.

As discussed in Sec. 4.2, the BT50 and BT70 datasets can be considered especially challenging due to the writing style variabilities. The BoF-HMM has to generalize across writing styles based on a single example of the query. When no annotated training data is used, the BoF-HMM outperforms all related methods on the BT50 and on the BT70 benchmarks by a large margin, see Tab. 1. Besides the BoF-HMM, only the *local feature matching* method[61] achieves a mAP higher than 50% on BT50. This method is designed for handling larger word size variabilities which fits with the characteristics of the Bentham manuscripts. The results on BT50 and BT70 demonstrate the word spotting capabilities of the BoF-HMM in a very challenging word spotting scenario. The significant improvements over the baseline of 10.8% (absolute) on BT50 and 8.8% (absolute) on BT70 can be seen as major advancements. The results emphasize the importance of handling word size variabilities with a sequence model.

With respect to the KP20 and BO20 datasets, the results are consistent with the other benchmarks, see Tab. 1. The BoF-HMM outperforms the *random projections*[22] method as well as the baseline results. The difference between the BoF-HMM and the baseline on BO20 is not significant which can be explained with the large variability on BO20.

A comparison with methods from the literature that use annotated training data is shown in Tab. 2. For this purpose, the number of training annotations is reported along with the mAP. Tab. 2 shows that the mAP improvements achieved by the training-based methods come at the cost of large training corpora with thousands of annotated samples. If the visual variability in the document images is limited, like

on KP20, the word spotting performance achieved by the BoF-HMM compares very favorably. It should be noted that the results for the segmentation-free application of the PHOCNET[53] are reported in Ref. 36. The difference to the PHOCNET application in Ref. 47 is the generation of region hypotheses and the different training dataset sizes. The GMM-HMM[58] and the recurrent neural network (CRNN)[37] rely on an automatic line segmentation and training annotations on line level. The training-based methods in Tab. 2 mostly outperform the BoF-HMM. However, the BoF-HMM can support the creation of an annotated training dataset which is required for these methods.

4.5. *Descriptor size estimation*

Obtaining an optimal SIFT descriptor size requires an annotated validation set, see Sec. 4.6. If a collection of document images is explored with automatic methods for the first time, this annotated validation set is usually not available. In the targeted scenario, the application of BoF-HMMs requires an automatic estimation.

The descriptor size is related to the typical height of the text core area. The height estimate is based on horizontal projection-profiles, cf. Ref. 21. A projection-profile is computed for each gray-scale document image. Provided that the pen-stroke is represented with low image intensity values, text lines will be represented as valleys and document background regions will be represented as elevations in the projection-profile. A simple approach to text line detection is to threshold the projection-profile at a lower percentile q_{profile} of the distribution of projection-profile values. The height estimate for each detection is given by the run-length of projection-profile values that fall below the threshold obtained at percentile q_{profile}. It is important to note that the estimated heights strongly depend on the chosen percentile. For this reason, the global height estimate is obtained at an upper percentile $q_{\text{height}} = 100 - q_{\text{profile}}$ of the estimated text height distribution of the height estimates across all documents. Choosing percentile q_{height} in dependence of q_{profile} leads to a larger global height estimate when the height distribution is dominated by smaller heights and a smaller global height estimate when the height distribution is dominated by larger heights. Text height estimations are determined for three different percentiles $q_{\text{profile}} \in \{20, 25, 30\}$. The descriptor size estimate is obtained by rounding the average of the three height estimates to any of the descriptor sizes $\{16, 24, 32, \dots\}$.

4.6. *Optimization*

In the following, the objective is to analyze the effect of different parameterizations and architectural design choices to the performance of the BoF-HMM. For this purpose, experiments will be performed on the validation benchmarks GW20 and BT50. The effect of an adjustment is measured with a permutation test. Except for the SIFT descriptor size, the results are consistent on both benchmarks.

Table 3: BoF representation evaluation (mAP [%])

Grid sampling step	Vocabulary size	GW20	BT50
3	4096	**75.1**	**56.4**
5	4096	*73.0*	*50.9*
3	512	*68.0*	52.9
3	1024	*71.0*	55.3
3	2048	*73.5*	57.0
3	6144	75.5	55.2
3	8192	75.5	55.0

Table 4: BoF descriptor size evaluation (mAP [%])

SIFT descriptor size	GW20	BT50
16	*64.5*	*46.1*
24	*71.0*	54.1
32	*73.7*	**56.4**
40	75.0	53.2
48	**75.1**	*49.3*
56	*73.9*	*44.4*

4.6.1. *Bag-of-features representations*

Three meta-parameters are important for defining the BoF sequences which are used in order to represent document image regions. The *grid sampling step* specifies the distance of the SIFT descriptor center points in horizontal and vertical direction. The *vocabulary size* defines the number of visual words in the codebook. Both parameters are evaluated in Tab. 3. The SIFT *descriptor size* specifies the edge length of the square descriptor area. The influence of different descriptor sizes is presented in Tab. 4.

Our experiments show that the descriptor sampling step in the dense grid has a large influence on the performance. The performance difference between sampling steps of 3 and 5 pixels is significant. However, it has to be noted that a higher grid resolution comes at the cost of reduced computational efficiency. While a grid step of 5 pixels results in an average of 651×400 grid rows and grid columns on the GW20 dataset, a grid step of 3 pixels leads to an average grid resolution of 1084×667 over the 20 document images. This grid resolution is already demanding with respect to computational efficiency and memory efficiency. An even smaller grid step can be considered as infeasible for these reasons.

Less critical with respect to the performance is the vocabulary size. The experiments on both benchmarks show that high performance is achieved with large vocabulary sizes where the mAP converges at 4096 visual words. Larger vocabularies do not lead to significant improvements. Regarding the characteristics of the

Table 5: Output model comparison (mAP [%])

Output mixture model	GW20	BT50
Visual words	**75.1**	**56.4**
vMF	*71.6*	*50.2*
EDCM	*70.1*	*49.9*
Multinomial	*68.1*	*49.1*

datasets, this can be explained with the limited visual variability of the text in the document images. In this targeted scenario, a visual vocabulary with 4096 visual words can be considered as a robust parameterization.

The size of the SIFT descriptors is the most sensitive and, therefore, also most important meta parameter for word spotting with BoF-HMMs. Since SIFT descriptors represent the visual image features, the descriptor size strongly depends on the dataset. Tab. 4 shows that the locally optimal result for GW20 is achieved with a descriptor size of 48 pixels while the locally optimal result for the BT50 benchmark is achieved with a descriptor size of 32. When the performance of these two descriptor sizes is compared on the GW20 benchmark only, the best performance of 75.1% mAP is significantly better than the 73.7% mAP achieved with the value 32. The effect can be observed in analogy on the BT50 benchmark.

This sets the descriptor size apart from all other meta parameters. In practice, finding an optimal SIFT descriptor size requires a validation set of annotated samples that is representative for the corresponding document collection. In the targeted scenario where historians start the exploration of a document collection, such a validation set will typically not be available. However, it can be shown that the estimates obtained according to Sec. 4.5 correspond to the optimal descriptor sizes derived from the validation sets, for all considered benchmarks except for GW20.[43]

4.6.2. *Output model comparison*

BoF output models are required for modeling the generation of BoF vectors in the statistical HMM process. In this regard, four approaches have been presented in Sec. 3.4. The *von-Mises-Fisher* (vMF) mixture model, the multinomial mixture model as well as the *exponential Dirichlet compound multinomial* (EDCM) mixture model are estimated from BoF vectors in an unsupervised manner. For this purpose, BoF vectors are extracted from all line hypotheses in all document images of the corresponding datasets. The visual-word mixture model is directly given by the visual vocabulary and does not require an additional estimation step besides clustering SIFT descriptors.

Tab. 5 shows a comparison of the models on the validation benchmarks. The experiments show that the visual-word mixture model significantly outperforms the other models. Thus, the best trade-off between generalization capabilities and

Table 6: Query word HMM meta parameters (mAP [%])

States $\lfloor \cdot T_\mathrm{q} \rfloor$	Iterations	GW20	BT50
0.3	0	*72.1*	*47.9*
0.5	0	74.5	54.1
0.7	0	**75.1**	**56.4**
0.9	0	*72.8*	52.6
0.7	3	74.6	53.6
0.7	5	*73.8*	*51.0*

specificity is achieved with the visual-word mixture model. The consideration of the vMF mixture model is inspired from the wide use of cosine similarity for word spotting. However, the model is very sensitive to the selection of meta parameters, see Ref. 43. An interesting observation is the performance difference of the EDCM model in comparison to the visual-word mixture model. This can be explained with the better generalization capabilities of the visual-word model. An EDCM mixture component represents the generation of the entire BoF vector. In contrast, the generation of just a single visual word is represented by the visual-word mixture model. It is more likely to find individual occurrences of visual words in the document images (disjunction) than configurations of multiple visual words in the document images (conjunction).

4.6.3. *Query modeling*

The most important meta parameters for the query word HMM include the number of HMM states and the number of Baum-Welch training iterations for estimating the model. Since the query word HMM represents just a single document image region in the query-by-example scenario, the training process reduces to the estimation of transition probabilities and mixture component weights. The output model is not considered in this regard but estimated in an unsupervised manner. The number of states is given as a percentage of the length T_q of the BoF vector sequence that is extracted from the query word region. The model is initialized based on a linear alignment of these BoF vectors with the states. Each HMM state represents the visual appearance of a section of the query word image. Thus, choosing the (relative) number of states is a trade-off between the specificity and the generalization capabilities of the model. In this regard, the number of states can be seen as a length constraint for the document image regions that can be retrieved with the query word HMM. Due to the linear HMM topology, the number of BoF vectors in a document image region must be greater or equal to the number of states. Allowing for more flexibility in the alignment with a *Bakis* topology did not change the results significantly, cf. Ref. 43. Tab. 6 shows the effect of the relative number of states and the number of training iterations on the validation benchmarks. The locally optimal scaling factor for the number of states is 0.7.

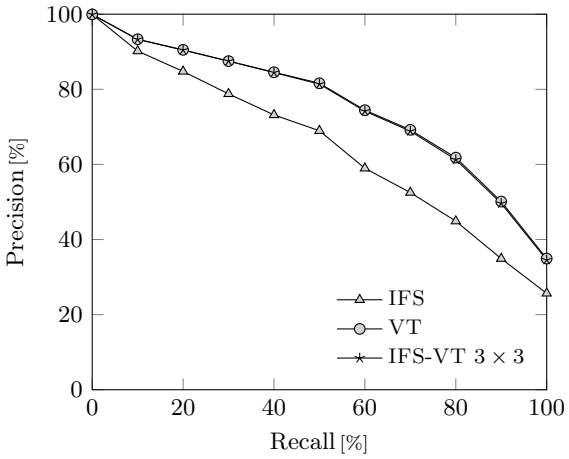


Figure 5: Two-stage decoding on GW20.

Furthermore, an interesting behavior can be observed for the number of training iterations in Tab. 6. In comparison to using the model directly after the initialization, the measured performance is reduced after three training iterations and significantly worse after five iterations on both benchmarks. This can be explained with overfitting of the model due to the lack of an annotated training dataset. When optimizing the total output probability, the Baum-Welch algorithm focusses on the mixture components, here visual words, that most of the BoF vectors have in common according to their probabilistic alignment with the states. However, with just a single sequence there is no guarantee that these are the components that are relevant for spotting the query word.

4.6.4. *Retrieval efficiency*

Efficiency is achieved by decoding in two stages. Mixture component voting retrieves potentially relevant regions at high *recall* and with high *computational efficiency*. Viterbi decoding allows for re-ranking and refining these regions with high *average precision* at the *cost* of high *computational complexity*.

Fig. 5 shows *interpolated precision-recall* curves on the GW20 benchmark: for the first stage only (IFS), after re-ranking with a 3×3 mask (IFS-VT 3×3) and with the second stage only (VT). It can be seen that the precisions for the two-stage approach are almost identical to the precisions obtained for the second stage only. This demonstrates that hardly any accuracy is lost with re-ranking in comparison to a full search.

In order to allow for an assessment of the applicability in practice, Tab. 7 shows a quantitative comparison of the retrieval efficiency measured in average milliseconds per query and document. Mixture component voting is implemented in *C++*, the

Table 7: Retrieval time on GW20 (time [ms] and mAP [%])

Method	query/page	mAP
Baseline	> 60000	*70.2*
BoF-HMM (IFS)	1406	*64.1*
BoF-HMM (IFS-VT 1×1)	1859	74.1
BoF-HMM (IFS-VT 3×3)	2321	**75.1**
BoF-HMM (VT)	19268	75.4
Spatial pyramid ind.[49]	3	61.4
Exemplar SVM[4]	86	59.1
Scale-space pyramid[40]	115	56.0
Random projections[22]	80	50.1

Viterbi algorithm is implemented in the HMM toolkit *ESMERALDA* in *C* and the application of both components is implemented in *Python*. The processor is an *Intel(R) Xeon(R) CPU E5-2650 v4 @ 2.20GHz* with 12 physical cores.

The average retrieval times show that the high word spotting performance of the BoF-HMM comes at the cost of computational efficiency. In order to show results for a scenario with a standard hardware setup, the retrieval times refer to a configuration where line representations and inverted indices are not cached but computed on demand. Retrieval times refer to processing within a single thread in a single process. The parallelization capabilities of the CPU are not taken advantage of. Therefore, Tab. 7 shows worst case results.

If only the parallelization capabilities are taken into consideration, e. g., on the given CPU 10 pages can be processed in parallel without affecting the individual retrieval times, average retrieval times of around 140 ms in the first stage only (IFS), 190 ms after re-ranking (IFS-VT 1×1) and 230 ms after re-ranking with a 3×3 mask (IFS-VT 3×3) per query and page are feasible in practice. The result for applying the second stage only (VT) is added as a reference. The results for the BoF-HMM include the time for computing line representations and indices. It has to be noted that all of the related approaches would benefit from parallelization.

In contrast to the retrieval times, the query estimation time does not scale with the number of documents. On the GW20 benchmark, the average time for obtaining the query model is 202 milliseconds. Thus, overall, the time can be neglected.

5. Conclusion

Annotation-free word spotting with BoF-HMMs makes document images searchable with minimum manual effort. The proposed method allows for preparing the collection for word spotting fully automatically. The only meta parameter that is sensitive to the visual characteristics of the document collection is the descriptor size. An automatic size estimate is based on an estimate of the typical text core height. The following architectural design choices for segmentation-free word spotting with BoF-

HMMs have been confirmed: The *visual-word mixture model* outperforms three other mixture models that are relevant for representing BoF vectors. Mixture component voting makes the application of the computationally expensive Viterbi decoding feasible in practice. Word spotting performance is affected only marginally. The combination of the different region detection approaches with Viterbi decoding addresses word size variabilities effectively. The computational effort is limited with a new voting-based decoding algorithm for semi-continuous HMMs.

We have proposed a method for segmentation-free word spotting that largely outperforms the state-of-the-art in the query-by-example scenario where no annotated training data is available. BoF-HMMs can support the creation of an annotated training dataset in order to further improve the results with neural networks. The method can directly be applied on a new dataset even if no training annotations are available.

Acknowledgements

This work has partially been supported by the German Research Foundation (DFG) within project Fi799/9-1. The authors would like to thank Marçal Rusiñol who supported this work in the early stages.

Bibliography

1. R. Ahmed, W. G. Al-Khatib and S. A. Mahmoud, A survey on handwritten documents word spotting, *Int. Journal Multimedia Information Retrieval* **6**(1) (2017) 31–47.
2. D. Aldavert, M. Rusiñol, R. Toledo and J. Lladós, Integrating visual and textual cues for query-by-string word spotting, in *Proc. of the Int. Conf. on Document Analysis and Recognition* (2013) pp. 511–515.
3. D. Aldavert, M. Rusiñol, R. Toledo and J. Lladós, A study of bag-of-visual-words representations for handwritten keyword spotting, *Int. Journal on Document Analysis and Recognition* **18** (September 2015) 223–234.
4. J. Almazán, A. Gordo, A. Fornés and E. Valveny, Segmentation-free word spotting with exemplar SVMs, *Pattern Recognition* **47**(12) (2014) 3967 – 3978.
5. R. A. Baeza-Yates and B. A. Ribeiro-Neto, *Modern Information Retrieval - the concepts and technology behind search, Second edition* (Pearson Education Ltd., Harlow, England, 2011).
6. D. Ballard, Generalizing the hough transform to detect arbitrary shapes, *Pattern Recognition* **13**(2) (1981) 111 – 122.
7. A. Banerjee, I. S. Dhillon, J. Ghosh and S. Sra, Clustering on the unit hypersphere using von Mises-Fisher distributions, *Journal of Machine Learning Research* **6** (December 2005) 1345–1382.
8. C. Barrett, R. Hughey and K. Karplus, Scoring hidden Markov models, *Comput Appl Biosci* **13**(2) (1997) 191–199.
9. C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)* (Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006).
10. T. Causer, K. Grint, A.-M. Sichani and M. Terras, Making such bargain: Transcribe bentham and the quality and cost-effectiveness of crowdsourced transcription, *Digital Scholarship in the Humanities* **33**(3) (2018) 467–487.

11. G. Csurka, C. R. Dance, L. Fan, J. Willamowski and C. Bray, Visual categorization with bags of keypoints, in *Workshop on Statistical Learning in Computer Vision* (2004) pp. 1–22.
12. N. Dalal and B. Triggs, Histograms of oriented gradients for human detection, in *Proc. IEEE Comp. Soc. Conf. on Computer Vision and Pattern Recognition*, Vol. 1 (June 2005) pp. 886–893.
13. C. Elkan, Clustering documents with an exponential-family approximation of the Dirichlet compound multinomial distribution, in *Proc. of the Int. Conf. on Machine Learning*, *ICML '06* (ACM, New York, NY, USA, 2006) pp. 289–296.
14. G. A. Fink, *Markov Models for Pattern Recognition, From Theory to Applications*, 2 edn. (Springer, Berlin, Heidelberg, 2014).
15. A. Fischer, A. Keller, V. Frinken and H. Bunke, Lexicon-free handwritten word spotting using character HMMs, *Pattern Recognition Letters* **33**(7) (2012) 934–942.
16. B. Gatos and I. Pratikakis, Segmentation-free word spotting in historical printed documents, in *Proc. of the Int. Conf. on Document Analysis and Recognition* (2009)
17. A. P. Giotis, G. Sfikas, B. Gatos and C. Nikou, A survey of document image word spotting techniques, *Pattern Recognition* **68** (2017) 310 – 332.
18. N. Gurjar, S. Sudholt and G. A. Fink, Learning deep representations for word spotting under weak supervision, in *Proc. of the Int. Workshop on Document Analysis Systems* (April 2018) pp. 7–12.
19. N. R. Howe, Part-structured inkball models for one-shot handwritten word spotting, in *Proc. of the Int. Conf. on Document Analysis and Recognition* (August 2013) pp. 582–586.
20. H. Jegou, M. Douze and C. Schmid, Product quantization for nearest neighbor search, *IEEE Trans. on Pattern Analysis and Machine Intelligence* **33** (January 2011) 117–128.
21. K. Kise, Page segmentation techniques in document analysis, in D. Doermann and K. Tombre (eds.), *Handbook of Document Image Processing and Recognition* (Springer-Verlag, London, 2014) pp. 135–175.
22. A. Kovalchuk, L. Wolf and N. Dershowitz, A simple and fast word spotting method, in *Proc. of the Int. Conf. on Frontiers in Handwriting Recognition* (September 2014) pp. 3–8.
23. S. Lazebnik, C. Schmid and J. Ponce, Beyond bags of features: spatial pyramid matching for recognizing natural scene categories, in *Proc. IEEE Comp. Soc. Conf. on Computer Vision and Pattern Recognition*, Vol. 2 (2006) pp. 2169–2178.
24. Y. Leydier, A. Ouji, F. LeBourgeois and H. Emptoz, Towards an omnilingual word retrieval system for ancient manuscripts, *Pattern Recognition* **42**(9) (2009).
25. J. Lladós and G. Sanchez, Indexing historical documents by word shape signatures, in *Proc. of the Int. Conf. on Document Analysis and Recognition* (September 2007) pp. 362–366.
26. D. G. Long and A. T. Milne, *The manuscripts of Jeremy Bentham : a chronological index to the collection in the Library of University College London* (Library and Bentham Committee, University College London, 1981).
27. D. G. Lowe, Distinctive image features from scale-invariant keypoints, *Int. Journal of Computer Vision* **60** (2004).
28. R. E. Madsen, D. Kauchak and C. Elkan, Modeling word burstiness using the Dirichlet distribution, in *Proc. of the Int. Conf. on Machine Learning*, *ICML '05* (ACM, New York, NY, USA, 2005) pp. 545–552.
29. T. Malisiewicz, A. Gupta and A. A. Efros, Ensemble of exemplar-SVMs for object detection and beyond, in *Proc. of the Int. Conf. on Computer Vision* (November 2011)

pp. 89–96.
30. J. Matas, O. Chum, M. Urban and T. Pajdla, Robust wide-baseline stereo from maximally stable extremal regions, *Image and Vision Computing* **22**(10) (2004) 761–767.
31. D. Nister and H. Stewenius, Scalable recognition with a vocabulary tree, in *Proc. IEEE Comp. Soc. Conf. on Computer Vision and Pattern Recognition*, Vol. 2 (2006) pp. 2161–2168.
32. S. O'Hara and B. A. Draper, Introduction to the bag of features paradigm for image classification and retrieval, *ArXiv e-prints* (January 2011).
33. W. Pantke, M. Dennhardt, D. Fecker, V. Märgner and T. Fingscheidt, An historical handwritten arabic dataset for segmentation-free word spotting - hadara80p, in *Proc. of the Int. Conf. on Frontiers in Handwriting Recognition* (September 2014) pp. 15–20.
34. F. Perronnin and J. A. Rodríguez-Serrano, Fisher kernels for handwritten word-spotting, in *Proc. of the Int. Conf. on Document Analysis and Recognition* (July 2009) pp. 106–110.
35. I. Pratikakis, K. Zagoris, B. Gatos, G. Louloudis and N. Stamatopoulos, ICFHR 2014 competition on handwritten keyword spotting (H-KWS 2014), in *Proc. of the Int. Conf. on Frontiers in Handwriting Recognition* (2014) pp. 814–819.
36. I. Pratikakis, K. Zagoris, B. Gatos, J. Puigcerver, A. Toselli and E. Vidal, ICFHR 2016 handwritten keyword spotting competition (H-KWS 2016), in *Proc. of the Int. Conf. on Frontiers in Handwriting Recognition* (2016) pp. 613–618.
37. J. Puigcerver, A probabilistic formulation of keyword spotting, dissertation, Universitat Politècnica de València (2018).
38. J. Puigcerver, A. H. Toselli and E. Vidal, Probabilistic interpretation and improvements to the HMM-filler for handwritten keyword spotting, in *Proc. of the Int. Conf. on Document Analysis and Recognition* (August 2015) pp. 731–735.
39. J. Puigcerver, A. Toselli and E. Vidal, ICDAR 2015 competition on keyword spotting for handwritten documents, in *Proc. of the Int. Conf. on Document Analysis and Recognition* (2015) pp. 1176–1180.
40. I. Rabaev, K. Kedem and J. El-Sana, Keyword retrieval using scale-space pyramid, in *Proc. of the Int. Workshop on Document Analysis Systems* (April 2016) pp. 144–149.
41. J. Rodríguez-Serrano and F. Perronnin, Handwritten word-spotting using hidden Markov models and universal vocabularies, *Pattern Recognition* **42**(9) (2009).
42. J. Rodríguez-Serrano and F. Perronnin, A model-based sequence similarity with application to handwritten word spotting, *IEEE Trans. on Pattern Analysis and Machine Intelligence* **34**(11) (2012) 2108–2120.
43. L. Rothacker, Segmentation-free word spotting with bag-of-features hidden Markov models, dissertation, TU Dortmund University (2019).
44. L. Rothacker and G. A. Fink, Robust output modeling in bag-of-features HMMs for handwriting recognition, in *Proc. of the Int. Conf. on Frontiers in Handwriting Recognition* (October 2016) pp. 199–204.
45. L. Rothacker, M. Rusiñol and G. A. Fink, Bag-of-features HMMs for segmentation-free word spotting in handwritten documents, in *Proc. of the Int. Conf. on Document Analysis and Recognition* (August 2013) pp. 1305–1309.
46. L. Rothacker, M. Rusiñol, J. Lladós and G. A. Fink, A two-stage approach to segmentation-free query-by-example word spotting, *manuscript cultures* (7) (2014) 47–57.
47. L. Rothacker, S. Sudholt, E. Rusakov, M. Kasperidus and G. A. Fink, Word hypotheses for segmentation-free word spotting in historic document images, in *Proc. of the Int. Conf. on Document Analysis and Recognition* (November 2017) pp. 1174–1179.
48. L. Rothacker, S. Vajda and G. A. Fink, Bag-of-features representations for offline handwriting recognition applied to Arabic script, in *Proc. of the Int. Conf. on Fron-*

tiers in Handwriting Recognition (September 2012) pp. 149–154.
49. M. Rusiñol, D. Aldavert, R. Toledo and J. Lladós, Efficient segmentation-free keyword spotting in historical document collections, *Pattern Recognition* **48**(2) (2015) 545 – 555.
50. H. Sakoe and S. Chiba, Dynamic programming algorithm optimization for spoken word recognition, *IEEE Transactions on Acoustics, Speech, and Signal Processing* **26** (February 1978) 43–49.
51. A. Sharma and K. P. Sankar, Adapting off-the-shelf CNNs for word spotting & recognition, in *Proc. of the Int. Conf. on Document Analysis and Recognition* (August 2015) pp. 986–990.
52. R. Shekhar and C. Jawahar, Word image retrieval using bag of visual words, in *Proc. of the Int. Workshop on Document Analysis Systems* (March 2012) pp. 297–301.
53. S. Sudholt and G. A. Fink, PHOCNet: A deep convolutional neural network for word spotting in handwritten documents, in *Proc. of the Int. Conf. on Frontiers in Handwriting Recognition* (October 2016) pp. 277–282.
54. S. Sudholt and G. A. Fink, A modified isomap approach to manifold learning in word spotting, in *Proc. of the German Conf. on Pattern Recognition* (2015) pp. 529–539.
55. S. Sudholt and G. A. Fink, Attribute CNNs for word spotting in handwritten documents, *Int. Journal on Document Analysis and Recognition* **21** (September 2018) 199–218.
56. K. Terasawa and Y. Tanaka, Slit style hog feature for document image word spotting, in *Proc. of the Int. Conf. on Document Analysis and Recognition* (July 2009) pp. 116–120.
57. A. H. Toselli, E. Vidal, V. Romero and V. Frinken, HMM word graph based keyword spotting in handwritten document images, *Information Sciences* **370-371** (2016) 497 – 518.
58. E. Vidal, A. H. Toselli and J. Puigcerver, High performance query-by-example keyword spotting using query-by-string techniques, in *Proc. of the Int. Conf. on Document Analysis and Recognition* (August 2015) pp. 741–745.
59. G. Washington, George washington papers, in *Letterbook 1, Aug. 11, 1754 - Dec. 25, 1755, Series 2,*(Letterbooks 1754 to 1799) (Library of Congress, Washington, DC, 1754)
60. T. Wilkinson, J. Lindström and A. Brun, Neural ctrl-f: Segmentation-free query-by-string word spotting in handwritten manuscript collections, in *Proc. of the Int. Conf. on Computer Vision* (October 2017) pp. 4443–4452.
61. K. Zagoris, I. Pratikakis and B. Gatos, Unsupervised word spotting in historical handwritten document images using document-oriented local features, *IEEE Trans. on Image Processing* **26** (August 2017) 4032–4041.

Biographical Sketch and Photo

Leonard Rothacker received the master's degree (Diploma) in computer science from TU Dortmund University, Dortmund, Germany, in 2011 and the Ph.D. degree (Dr.-Ing.) also in computer science from TU Dortmund University, Dortmund, Germany, in 2019. From 2011 to 2018 he has been a member of the *Pattern Recognition in Embedded Systems* group at TU Dormund University headed by Prof. Gernot A. Fink. He is currently working at Lufthansa Industry Solutions AS GmbH, Hamburg, Germany. His research interests lie in the fields of pattern recognition, computer vision and document analysis.

Gernot A. Fink received the master's degree (Diploma) in computer science from the University of Erlangen-Nuremberg, Erlangen, Germany, in 1991 and the Ph.D. degree (Dr.-Ing.) also in computer science from Bielefeld University, Bielefeld, Germany, in 1995. In 2002 he received the *venia legendi* (Habilitation) in Applied Computer Science from Bielefeld University.
From 1991 to 2005 he was with the Applied Computer Science Group at the Faculty of Technology of Bielefeld University. Since 2005 he is professor at the Department of Computer Science of TU Dortmund University, Dortmund, Germany, where he heads the *Pattern Recognition in Embedded Systems* group.
His research interests lie in the development and application of pattern recognition methods in the fields of computer vision, human activity reconition, and document image analysis with a focus on semi-supervised learning and deep neural networks.

Fabian Wolf received the master's degree in Automation and Robotics from TU Dortmund University, Dortmund, Germany in 2018. Since 2018 he has been a member of the *Pattern Recognition in Embedded Systems* group at the TU Dormund University headed by Prof. Gernot A. Fink. His research interests lie in the fields of pattern recognition, computer vision and document image analysis. In his work, he focuses on shallow and deep learning methods for word spotting, with a special interest in semi-supervised, self-supervised and transfer learning techniques.