# QUERY-BY-EXAMPLE WORD SPOTTING USING MULTISCALE FEATURES AND CLASSIFICATION IN THE SPACE OF REPRESENTATION DIFFERENCES

*Mohamed Mhiri, Mohamed Cheriet, Christian Desrosiers*

École de technologie supérieure (ÉTS)
Montreal, Canada

## ABSTRACT

Word spotting in document images is a challenging problem, due to the large intra-class variability in handwritten shapes and the lack of labeled data. To tackle these challenges, this paper proposes an efficient multiscale representation for word images, which is learned in an unsupervised manner using the spherical k-means algorithm. A pooling function is applied in a spatial grid to obtain a fixed-length vector of features, robust to small shifts in the image. Scale variability in handwritten data is also considered by using patches of various sizes in the encoding process. Another important contribution of this work is to model the training-based word spotting task as a classification problem in the space of representation differences, thereby allowing the learned model to find matches for word classes that were not seen in training. The proposed system is evaluated on the well-known George Washington (GW) dataset. Experimental results show that our system outperforms state-of-the-art word spotting approaches in both training-free and training-based scenarios.

***Index Terms***— word spotting, handwriting representation, representation learning, image retrieval.

## 1. INTRODUCTION

With the increasing number of digitized historical documents, the development of automated methods to extract and analyze handwritten text is attracting a growing amount of attention. However, due to the large variability of handwriting data and the frequent degradation of historical documents, this task remains quite challenging. In this paper, we tackle a key problem of text understanding, called *query-by-example word spotting*, the goal of which is to find all occurrences of a query word image in a dataset of digitized documents.

To process handwritten data efficiently, it has to be encoded using an effective representation [1]. Over the years, many handwriting representations have been proposed in the literature [2, 3, 4]. Most of these representations concentrate on the critical step of feature encoding, which can be divided into two broad categories: handcrafted features and learned features [5, 6]. Handcrafted features are widely used since they are simple to implement and can be interpreted easily. However, recent studies have shown the advantages of learned features, which can encode richer information and better adapt to the target task than handcrafted features [7, 8]. Generally, features learning techniques can be categorized as *supervised* methods, which use labeled data to learn high-level features, or *unsupervised* methods, that are less human dependent and can learn the features without labeled data [9, 10].

In the work of Rusinol et al. [2], a word image is divided into a grid of equally sized overlapping patches. Then, each patch is described using the bag-of-visual-words model over extracted SIFT descriptors [11]. This representation is refined using the Latent Semantic Indexing technique [12]. Similarly, the work of Almazan et al. [4] uses a histogram-based representation, where a word image is divided into equally sized patches, each one represented by Histogram of Oriented Gradients (HOG) features. A more recent work, [13] used a Gaussian mixture model (GMM) trained on set of SIFT features to compute a Fisher vector (FV).

In this work, we make two original contributions to the query-by-example word spotting problem:

1. A novel unsupervised handwriting representation is presented, which shows significant advantages in terms of accuracy and computational efficiency over existing approaches for the query-by-example word spotting problem. By using multiscale patches from the input word image, and via a spatial pooling strategy, this representation is made robust to the translation and scale variability of handwriting data;

2. We propose an efficient method for training-based word spotting, in which the task is considered as a classification problem in the space of representation differences. This method offers a simple way to extend the learned representation to classes that have not been used during training.

The following section describes the proposed handwriting representation, which combines steps for learning, encoding and pooling the representation's features. We then apply

Thanks the NSERC of Canada for their financial support

our representation to the tasks of training-free and training-based query-by-example word spotting, and show its advantages over state-of-the-art approaches for these tasks.

## 2. METHODOLOGY

### 2.1. Proposed handwriting representation

The proposed handwriting representation is composed of three steps, focusing on feature representation learning, feature encoding and spatial pooling. First, a set of patches is extracted randomly from the digitized documents. These patches are then processed by the spherical k-means algorithm [14] to learn a feature representation (i.e., a *codebook* or *dictionary*). In the second stage, this representation is used to map input word images in the space of learned features. Finally, after encoding a given word image, a pooling function is applied in a spatial grid, where features in each grid region are aggregated independently. This spatial pooling step is used to make the representation robust to small shifts in the word image.

To take into account the scale variability in handwriting data, this three-step process is repeated at multiple scales, each one corresponding to a different patch size in the word images. The pooled features obtained at each scale are then concatenated to form a single feature vector. The overall process is summarized in Fig. 1. The following sub-sections present each step in greater details.

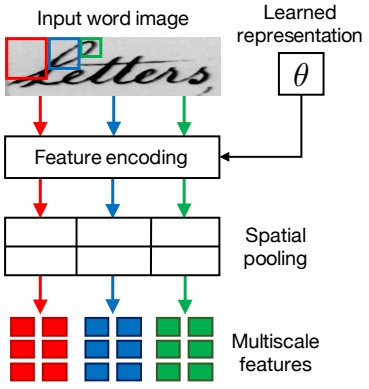


**Fig. 1**: The proposed multiscale representation for word images, where patches of different sizes are encoded using a learned representation $\theta$ and aggregated within grid regions ($3 \times 2$ grid in this example), via a pooling function.

#### 2.1.1. Representation learning

To learn the feature representation, we start by randomly extracting $T$ patches of $M \times M$ pixels from the digitized documents. These patches are then vectorized and pre-processed using Principal Component Analysis (PCA) [7, 8] to remove linear correlations between pixels. Denote as $X \in \mathbb{R}^{M^2 \times T}$ the matrix of pre-processed patches and $x_i \in \mathbb{R}^{M^2}$ the $i$-th patch vector of $X$. Following this, the spherical k-means algorithm is applied on $X$ to group these patches into $K$ clusters. The cluster centroids $\theta_k$, $k = 1, \ldots, K$, are then used to define the feature representation $\theta$. Starting with a random set of centroids, the spherical k-means algorithm performs the following steps:

1. For each patch $x_i$, assign $x_i$ to the centroid with smallest angular distance:
$$z_{ki} := \begin{cases} \theta_k^T x_i, & \text{if } k = \operatorname{argmax}_j |\theta_j^T x_i| \\ 0, & \text{otherwise,} \end{cases} \quad (1)$$
where $Z$ is the assignment matrix.

2. Recompute $\theta$ by projecting $Z$ on $X$:
$$\theta := XZ^T; \quad (2)$$

3. Unit normalize $\theta$:
$$\theta_k := \theta_k / \|\theta_k\|_2, \; k = 1, \ldots, K; \quad (3)$$

While standard k-means considers the Euclidean distance, this algorithm uses angular distance to map patches to their nearest centroid. The main advantage of this distance measure is that it is more informative than Euclidean distance in high dimensional spaces [14], such as in our application.

#### 2.1.2. Feature encoding

To encode a word image, a sliding window of same size as the patches (i.e., $M \times M$) is applied to the image. Let $X \in \mathbb{R}^{M^2 \times N}$ be the matrix of patches obtained from this process followed by PCA. The feature representation $\theta$, learned in the previous step, is used to map $X$ to its corresponding feature matrix $Y \in \mathbb{R}^{K \times N}$. As mapping function, we use the popular soft-threshold operator [9, 10, 15]

$$y_{ki} = \max\left(0, \theta_k^T x_i - \alpha\right), \quad (4)$$

where $\alpha$ is a user-defined parameter. It can be shown that features obtained via this operator minimize the following sparse least-angle regression problem:

$$\underset{Y}{\operatorname{argmin}} \; \frac{1}{2} \sum_{i=1}^{N} \|y_i - \theta^T x_i\|_2^2 + \alpha \sum_{i=1}^{N} \|y_i\|_1, \quad (5)$$

where $\|\bullet\|_1$ is the $L_1$ norm. Hence, parameter $\alpha$ controls the sparseness of encoded patches in the feature space.

#### 2.1.3. Spatial pooling

Capturing spatial information within the representation of word images is essential to make it robust to small shifts in these images. This is specially true for handwriting data, which may contain a high spatial variance. In this work,

we address this problem via a feature pooling strategy, by which the image is divided into a grid of $W \times H$ regions, and local features of each region are aggregated independently. While other pooling functions could be considered (e.g., average pooling), we used the max pooling function, which returns for each grid region the maximum value of features observed in that region. The final word image representation is then obtained as the concatenation of all these local region representations. Finally, this global representation is $L_2$-normalized to reduce the effect of noise in word images.

#### 2.1.4. Multiscale representation

The proposed representation has several interesting properties. Thus, it can accommodate word images of all sizes by generating a fixed-length vector of features. Using pooling, it also captures spatial information of features within a word image and offers robustness to local variability in the handwriting.

This representation can however be improved by considering features extracted at multiple scales. Toward this goal, we apply the feature representation, encoding and pooling process, described above, using $S$ patch sizes $M_s$, $s = 1, \ldots, S$. Let $Y_s$ be the features obtained for scale $s$, the final features are obtained by concatenating the features computed at all scales: $Y^T = [Y_1^T \ldots Y_S^T]$.

### 2.2. Word spotting

The word spotting problem is solved in two different scenarios: a *training-free* scenario where no training data are provided, and a *training-based* one, for which labeled occurrences of word images are available for training. The following subsections explain how the proposed features are used for each of these two scenarios.

#### 2.2.1. Training-free scenario

In this case, word spotting is performed via a simple matching approach based on Euclidean distance. For a given query word image, the Euclidean distance is calculated between the feature vector of this image and that of each word image in the document dataset. Note that in this case, the feature dimensionality $K$ is small, therefore the Euclidean distance is suitable.

These distances are then sorted in ascending order to produce a ranked list of matches. Let $Q$ the total number of test queries, word spotting performance is measured using the Mean Average Precision (MAP):

$$\text{MAP} = \frac{1}{Q} \sum_{q=1}^{Q} \frac{1}{|R_q|} \sum_{k=1}^{|R_q|} \text{prec}_k(L_q; R_q), \qquad (6)$$

where $R_q$ is the set of relevant regions for query $q$, $L_q$ is the ranked list of regions obtained for the same query, and $\text{prec}_k$ is the precision at cut-off $k$ (i.e., considering only the first $k$ items in the list).

#### 2.2.2. Training-based scenario

When training examples are available, we model the word spotting task as a binary classification problem in the space of representation differences. Instead of predicting the label of an input word image, our model considers pairs of word images $(i, j)$ and determines whether these two word images are from the same class (i.e., they correspond to the same handwritten word). Although the features of both word images could be given as input to the classifier (e.g., as in Siamese neural networks [16]), the proposed model uses the vector of absolute feature differences as input: $y_k^{(i,j)} = |y_{ki} - y_{kj}|$, $k = 1, \ldots, K$. This strategy has the advantage of limiting the number of model parameters, and considering explicitly symmetry (i.e., $y^{(i,j)} = y^{(j,i)}$).

However, a problem with the proposed strategy comes from class imbalance: most pairs of word images represent different word classes. To solve this issue, we select for each word image $i$ the $D$ word images $j$ most similar to $i$, based on Euclidean distance. These pairs of similar word images $(i, j)$ are then used to train the model. Parameter $D$ is typically selected such that, on average, half the training examples correspond to the same class. In this work, this parameter was set empirically to $D = 10$.

While other types of classifiers could also be considered, in this work, we used a linear Support Vector Machine (SVM) for its small number of meta-parameters requiring tuning, its low computational complexity, and its good generalization ability. Note that linear SVMs are commonly used for hard image recognition tasks [17]. During testing, for a given query word image in the test set, we first filter the list of possible matches by selecting only those classified as same class by the SVM. Remaining matches are then ranked by Euclidean distance, as in the training-free scenario. Thus, the only difference between the proposed training-free and training-based approaches is the pre-filtering step using the SVM.

## 3. EXPERIMENTS

### 3.1. Method parameters

The proposed method requires the tuning of the following meta-parameters: the pooling grid size $W \times H$, the number of encoding features $K$ (i.e., centroids in the clustering algorithm), the patch size $M$, and the soft-threshold parameter $\alpha$ to control feature sparseness. Based on prior experiments, we selected a $2 \times 4$ pooling grid and $K = 256$ features per grid region, giving a total of $2 \times 4 \times 256 = 2048$ features per word image. To have a multiscale representation, these features were obtained separately for three different patch sizes (i.e., $M_1 = 16$, $M_2 = 22$ and $M_3 = 28$), and combined in a

single vector of $3 \times 2048 = 6144$ features. Finally, in order to have a uniform sparseness across different patch sizes, we set $\alpha$ to be proportional to $M$: $\alpha = \beta \cdot M$, with $\beta \in [0, 1]$.

### 3.2. Word spotting

We evaluated our word spotting approach using the George Washington dataset [2], which contains 20 pages of historical documents and includes 4860 word annotations made by two different experts. As in [18] and [13], we used a 4-fold cross-validation methodology, where the 4860 examples are split into 4 even-sized subsets, each one removed in turn from the training set and used as test set to measure the MAP. Performance is reported as the mean MAP computed across these 4 folds. Note that only queries with at least two occurrences were considered.

**Table 1**: Word spotting performance (MAP) of the proposed training-free and training-based methods, for various values of sparseness parameter $\beta$.

| METHOD | $\beta=0$ | $\beta=0.1$ | $\beta=0.2$ | $\beta=0.3$ |
|---|---|---|---|---|
| **Training-free** (Euclidean distance ranking) | 61.9% | 64.4% | **66.1%** | 62.7% |
| **Training-based** (SVM pre-filtering + Euclidean distance ranking) | 77.3% | 78.7% | **79.6%** | 77.2% |

Table 1 gives the performance (MAP) of our method in the training-free and training-based word spotting scenarios, for different values of sparseness parameter $\beta$. Results show that using training data to pre-filter poor candidates can significantly improve performance. Thus, our training-based model obtained a highest MAP of 79.6% compared to 66.1% for the training-free model. Moreover, we observe that parameter $\beta$ has a critical impact on performance, and that a maximum MAP is reached for $\beta = 0.2$ in both training-free and training-based methods. This illustrates the benefit of using sparse regularization to dynamically select the most useful features in the representation.

**Table 2**: Word spotting performance (MAP) of the proposed *training-free* method, compared to the state-of-the-art approaches for this task.

| METHOD | PERFORMANCE (MAP) |
|---|---|
| Vinciarelli features [19] | 50.0% |
| HOG features + exemplar SVM ranking [20] | 59.1% |
| Fisher vector [13] | 62.7% |
| **Our training-free method** (Euclidean distance ranking) | **66.1%** |

In Tables 2 and 3, we compare the proposed methods with state-of-the-art approaches for training-free and training-based word spotting. In the training-free scenario, we observe that our method outperforms popular handwriting representations based on Vinciarelli features [19], HOG features [20] and Fisher vectors [13] by a significant margin. Our method, which uses spherical k-means for representation learning and Euclidean distance for ranking, obtained a MAP value of 66.1%, compared to 62.7% for the second best approach (Fisher vector). Likewise, our training-based method, combining a representation-difference SVM classifier for pre-filtering matches and Euclidean distance for ranking filtered matches, obtains a significantly higher MAP than state-of-the art approaches based on Semi-continuous Hidden Markov Model (SC-HMM) [19] and character modelling [21]: 79.6%, versus 67.0% and 53.0% for these approaches. Note that both these approaches use labeled word directly to learn a representation, whereas our method defines the matching task as a binary classification problem in the space of representation-differences.

**Table 3**: Word spotting performance (MAP) of the proposed *training-based* method, compared to the state-of-the-art approaches for this task.

| METHOD | PERFORMANCE (MAP) |
|---|---|
| Semi-continuous HMM [19] | 53.0% |
| Liang et al. [21] | 67.0% |
| **Our training-based method** (SVM pre-filtering + Euclidean distance ranking) | **79.6%** |

## 4. CONCLUSION

This paper proposed two methods for the task of word spotting in training-free and training-based scenarios. These methods use an efficient representation for word images containing handwritten data, which is learned in an unsupervised manner via the spherical k-means algorithm. This representation is made robust to small shifts in the image using spatial pooling. Moreover, scale variability in handwriting data is addressed via a multiscale approach, where patches of different sizes are used to build the representation. In the training-based scenario, labeled word images are used to learn a classification model, predicting whether two word images belong to the same class. An important advantage of this strategy is that it can be used to find occurrences of a word image, even though the class of this word was not seen during training. Experiments on the George Washington (GW) dataset show the proposed methods to outperform state-of-the-art word-spotting approaches, both in the training-free and training-based scenario.

## 5. REFERENCES


[1] Y. Chherawala, P. P. Roy, and M. Cheriet, "Feature design for offline arabic handwriting recognition: Handcrafted vs automated," *International Conference on Document Analysis and Recognition*, 2013.

[2] M. Rusinol, D. Aldavert, R. Toledo, and J. Llados, "Browsing heterogeneous document collections by a segmentation-free word spotting method," *International Conference on Document Analysis and Recognition*, 2011.

[3] H. Greenspan P. Keaton and R. Goodman, "Keyword spotting for cursive document retrieval," *Document Image Analysis*, 1997.

[4] Jon Almazn, Albert Gordo, Alicia Forns, and Ernest Valveny, "Segmentation-free word spotting with exemplar svms," *Pattern Recognition*, 2014.

[5] A. Graves, M. Liwicki, S. Fernndez, R. Bertolami, H. Bunke, and J. Schmidhuber, "A novel connectionist system for unconstrained handwriting recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2009.

[6] F. Slimane, O. Zayene, S. Kanoun, A. M. Alimi, J. Hennebert, and R. Ingold, "New features for complex arabic fonts in cascading recognition system," *International Conference on Pattern Recognition*, 2012.

[7] Yoshua Bengio, "Learning deep architectures for AI," *Foundations and Trends in Machine Learning*, 2009.

[8] Yoshua Bengio, Aaron C. Courville, and Pascal Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2013.

[9] Adam Coates, Andrew Y. Ng, and Honglak Lee, "An analysis of single-layer networks in unsupervised feature learning," *International Conference on Artificial Intelligence and Statistics*, 2011.

[10] Adam Coates and Andrew Y. Ng, "Selecting receptive fields in deep networks," *Advances in Neural Information Processing Systems*, 2011.

[11] Gabriella Csurka, Christopher R. Dance, Lixin Fan, Jutta Willamowski, and Cdric Bray, "Visual categorization with bags of keypoints," *In Workshop on Statistical Learning in Computer Vision, ECCV*, 2004.

[12] Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman, "Indexing by latent semantic analysis," *Journal of the American society for information science*, 1990.

[13] Jon Almazán, Albert Gordo, Alicia Fornés, and Ernest Valveny, "Word spotting and recognition with embedded attributes," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2014.

[14] Kurt Hornik, Ingo Feinerer, Martin Kober, and Christian Buchta, "Spherical k-means clustering," *Journal of Statistical Software*, 2012.

[15] Alex Krizhevsky and Geoffrey Hinton, "Learning multiple layers of features from tiny images," *Technical report, University of Toronto*, 2009.

[16] Sergey Zagoruyko and Nikos Komodakis, "Learning to compare image patches via convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 4353–4361.

[17] Karen Simonyan and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[18] V. Frinken, A. Fischer, R. Manmatha, and H. Bunke, "A novel word spotting method based on recurrent neural networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012.

[19] Jose Rodriguez-Serrano and Florent Perronnin, "A model-based sequence similarity with application to handwritten word spotting," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2012.

[20] Jon Almazán, Albert Gordo, Alicia Fornés, and Ernest Valveny, "Segmentation-free word spotting with exemplar svms," *Pattern Recognition*, 2014.

[21] Y. Liang, M.C. Fairhurst, and R.M. Guest, "A synthesised word approach to word retrieval in handwritten documents," *Pattern Recognition*, 2012.