Learning Local Image Descriptors for Word Spotting

Sebastian Sudholt, Leonard Rothacker, Gernot A. Fink
Department of Computer Science
Technische Universität Dortmund University
44221 Dortmund, Germany
Email: {sebastian.sudholt, leonard.rothacker, gernot.fink}@tu-dortmund.de

***Abstract*—The Bag-of-Features paradigm has enjoyed great success in computer vision as well as document image analysis applications. By far the most common approach here is to power the Bag-of-Features pipeline with SIFT descriptors which are then clustered into a visual vocabulary using Lloyd's algorithm. In contrast to using handcrafted descriptors, many researches have started to use descriptors that have been learned from data. While descriptor learning is common in other computer vision tasks, there has been little work on learning descriptors for document analysis purposes.**

In this work we propose a descriptor learning pipeline designed for word spotting. Evaluation results on the well known George Washington database demonstrate that word-spotting results can effectively be improved by learning specialized local image descriptors.

I. Introduction

The automatic transcription of handwritten documents is a challenging task, which, opposed to printed character recognition, is still considered an unsolved problem. Often times, the documents at hand are from ancient times and exhibit severe degradations such as fading ink or noise. In addition to that, the variability for identical characters is much higher than for printed characters. This makes it hard for standard OCR methods to obtain good results on these kinds of documents.

Over the years, different approaches have been proposed to overcome the aforementioned limitations. *Keyword spotting* or simply *word spotting* has been a succesful approach to indexing document images. Instead of classifying word images, word spotting seeks to retrieve all regions from a given document or document collection that are similar to the supplied query [1]. The most common approach to word spotting is *query-by-example*. Here, the similarity between different word images is computed from their visual appearance. The models used for this task include Recurrent Neural Networks [2], Spatial Pyramids [3] and Hidden Markov Models [4]. The advantage of the latter two lies in the fact, that they only need a single word image in order to train the model. This is a desirable feature as word spotting is designed to automatically index a document collection and intends to spare human indexing [1]. Requiring less annotated word images is thus a major advantage and leads to less manual work.

The backbone for recent word spotting methods has been the use of local image features with the SIFT descriptor being the most prominent ([3], [4], [5], [6]). The standard procedure here is to compute local SIFT features over a dense grid and to generate a visual vocabulary from it by applying the generalized Lloyd clustering algorithm. While good results were reported for such pipelines, the results still leave room for improvement. Quantization is known to induce errors that especially show when descriptors are close to the border of two or more clusters. This limitation has been attempted to be overcome by soft assigning descriptors to the respective clusters [7]. The improvement over hard quantization was only marginal. Furthermore, soft assignment does not tackle the quantization error problem at its root. It would be much more desirable to obtain descriptors that lead to a more discriminative representation.

While learning such descriptors has been done succesfully in other computer vision applications, there has been, to the knowledge of the authors, no work concerning descriptor learning in word spotting. In this paper, we propose a descriptor learning pipeline that is tailor-made for query-by-example word spotting applications. The rest of the paper is organized as follows: In section II an overview of related work concerning descriptor learning is presented. Section III explains the proposed pipeline in detail. In section IV the results of the conducted experiments are shown while section V concludes the paper.

II. Related Work

Recent descriptor learning approaches have made use of distance learning techniques. Here, the goal is to learn a metric under which a given classification problem can be solved more easily. Different kinds of distance learning approaches have been proposed that make use of either deep architectures [8], SVMs [9] or linear projections [10]. Hua et al. [11] use Linear Discriminant Embedding to project SIFT descriptors into a subspace where matching descriptors lie together closely while non-matching descriptors are pushed apart. Two descriptors are considered matching, if they should lie closely together in the projected subspace while non-matching descriptors should lie apart. The main drawback here is that all descriptors need to be manually labeled as matching or non-matching. This is especially infeasable when descriptors are calculated over a dense grid in which case there can easily be millions of descriptors for a given image.

In order to conquer these limitations, Philbin et al. [12] proposed a method that can determine matching and non-matching descriptors in an unsupervised fashion. The first step here is to extract local image descriptors from a set of images. Afterwards, pairs of images $\mathbf{A}$ and $\mathbf{B}$ are randomly drawn from the set and for each descriptor from $\mathbf{A}$ its descriptor space nearest neighbor from $\mathbf{B}$ is calculated. From these correspondences, a homography is calculated between the two images. Descriptor pairs, that are consistent with this transformation, are considered matches. Additionally, the class

of non-matching descriptors is further split up into two classes: descriptor pairs that are inconsistent with the previsouly calculated homography transformation and descriptor pairs that are picked at random. The goal now is the same as in [11]: find a projection that pushes together matching descriptor while keeping descriptor pairs from the two non-match classes apart. The results show that splitting the class of non-matching descriptor pairs into two different sub-classes has an added benefit on the performance of the descriptor learning system.

III. Method

Our method for learning descriptors in word spotting is inspired by the work of Philbin et al. [12]. The basic concept is to find pairs of descriptors that should lie closely in descriptor space and pairs that should lie far apart. While it is cumbersome to manually define the pairs, the supervised approach is also infeasable in a query-by-example framework. It is desirable here to infer all information necessary from the raw, unannotated samples only instead of gathering a large annotated set of word images. In [12] the pairs are generated in an unsupervised fashion by randomly selecting image pairs from the supplied collection and finding the descriptor space nearest neighbors.

While this concept would be possible in a word spotting scenario also, we can exploit the structural knowledge about the problem at hand: the keypoints for matching descriptors from two word images should lie approximately at the same relativ position in their respective images. Figure 1 visualizes this concept. In the left hand column, the location of descriptor pairs is shown which are descriptor space nearest neighbors and have approximately the same spatial position. The middle column shows pairs, that are descriptor space nearest neighbors but do not match spatially. Finally, the right hand column shows random pairs of descriptors.

In order to find the three sets of descriptor pairs, two word images have to be selected from which the descriptors are taken. Randomly selecting two images, as is done in [12], leads to a high number of image pairs being discarded as the random selection generates a high amount of mismatching image pairs. While this limitation could be overcome by selecting word pairs in a supervised manner, this requires the annotation of a number of words. This is a highly undesirable characteristic, especially in a query-by-example scenario. In order to overcome this, we propose an unsupervised approach to find corresponding word images: a single run of query-by-example word spotting is performed on the documents at hand. After that, the retrieval list is cut off at a small number of items thus only keeping the very best results. This leaves the system with enough correspondences to infer the three descriptor pair sets while keeping wrong correspondences at a minimum. The selection of corresponding word images is even further refined by only taking descriptors from those image pairs that contain more positive matches than a certain threshold.

After finding the descriptor pairs, a projection function is learned and the projected descriptors are clustered to generate the visual vocabulary. Figure 2 illustrates the descriptor learning process: After generating an initial Bag-of-Feature representation, a single run of word spotting is performed in order to find word images that are similar. The retrieval list is

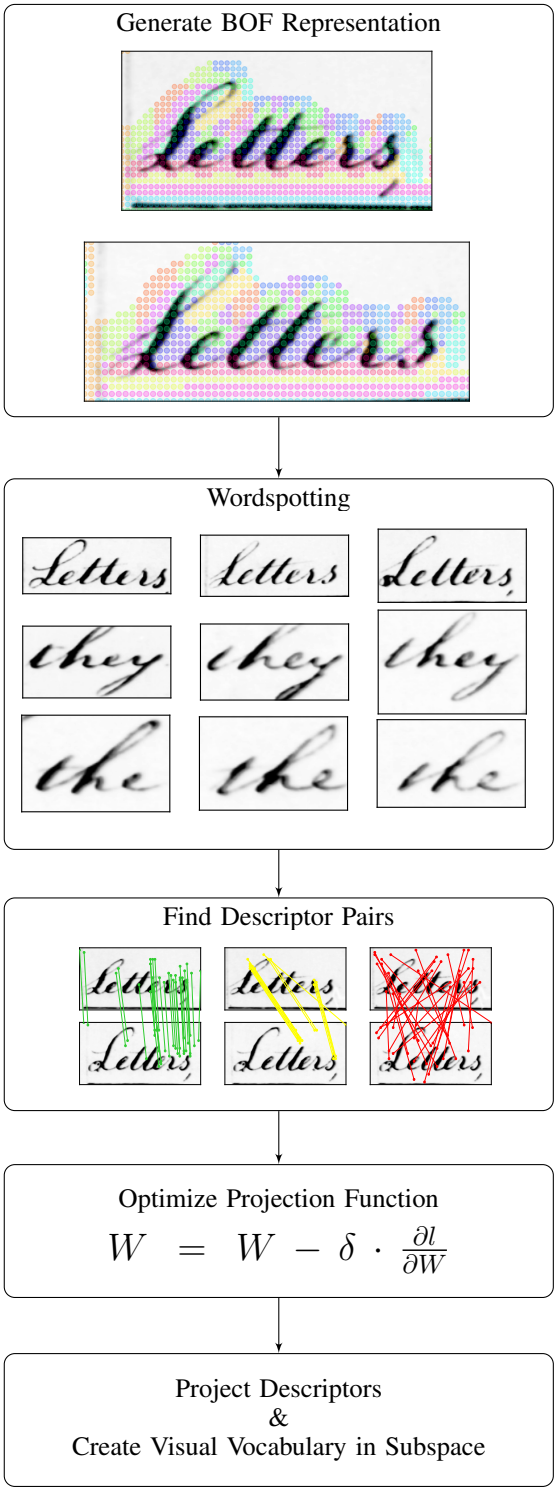

Fig. 2: Schematic overview of the proposed method: After an initial word spotting run the three sets of descriptor pairs are generated. The projection function is learned from the three sets and the projected descriptors are clustered in the subspace to produce the new visual vocabulary.

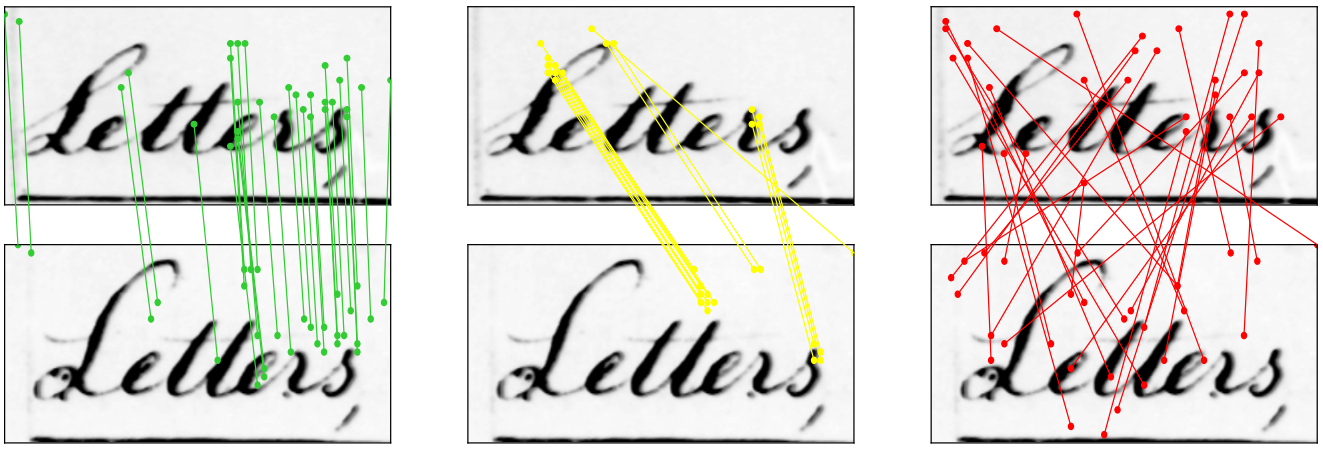

Fig. 1: The figure shows the three different classes of descriptor pairs for two words from the George Washington database. The left column displays matching descriptor pairs (descriptor space nearest neighbors and spatially close). The middle column shows nearest neighbor mismatches (descriptor space nearest neighbors but not spatially close). The right column shows random descriptor pairs.

cut off at a small number of retrieved images in order to infer the descriptor correspondences. Similar to [12], descriptors from two word images are considered putative matches if they are descriptor space nearest neighbors and pass Lowe's second nearest neighbor test [13]. Additionally, the relative position for each descriptor is calculated from the word boundaries. A putative match is considered a match if the relative position of the descriptor varies by only a small amount. In the following, we will refer to the test for relative positions as *spatial test*.

The procedure described above generates three sets of descriptor pairs:

Positive Matches (P) The set of descriptor pairs that are putative matches and pass the spatial test

Nearest Neighbor Mismatches (NM) The set of descriptor pairs that are putative matches but do not pass the spatial test

Random Mismatches (RM) The set of randomly drawn descriptor pairs

The pair-wise intersection of the three sets is empty. Having found pairs of matching and mismatching descriptors, we now try to find a function F that projects the descriptors into a subspace in which positive matches are pushed together while negative matches are pushed apart. In order to obtain this projection function, the loss function

$$\begin{aligned} l(F) = &\sum_{(\mathbf{x},\mathbf{y})\in P} \mathcal{L}(b_1 - d_F(\mathbf{x},\mathbf{y}))+ \\ &\sum_{(\mathbf{x},\mathbf{y})\in NM} \mathcal{L}(d_F(\mathbf{x},\mathbf{y}) - b_1)+ \\ &\sum_{(\mathbf{x},\mathbf{y})\in RM} \mathcal{L}(d_F(\mathbf{x},\mathbf{y}) - b_2) \end{aligned} \tag{1}$$

as proposed by Philbin et al. [12] is optimized with resepct to F. Here,

$$\mathcal{L}(z) = \log(1 + e^{-z}) \tag{2}$$

is the logistic-loss function, which is an approximation of the hinge-loss function, and

$$d_F(\mathbf{v},\mathbf{w}) = ||F(\mathbf{v}) - F(\mathbf{w})||_2 \tag{3}$$

is the Euclidean distance of the two descriptors $\mathbf{v}$ and $\mathbf{w}$ after projection. The margins b_1 and b_2 are used to seperate pairs in P from those in NM while keeping the random pairs from RM as distant as possible. As can be seen in equation 1, when optimizing l w.r.t. F, the distance of pairs in P becomes less than b_1 while the distance from pairs in NM becomes larger than this margin. The margin b_2 accounts for keeping pairs from RM away from each other. When optimizing l, scaling both margins by a constant value only leads to scaling the projected vectors. Thus, b_1 and b_2 can be combined into a single parameter $\frac{b_2}{b_1}$ which is called the margin ratio.

Please note that we do not use a regularization term in l as is done by Philbin et al. [12]. We found that it is not necessary as the presented problem does not seem to be prone to degradation. Thus, we left the costly evaluation of a spectral norm out of the loss function.

For the projection function, we use a linear projection:

$$F(\mathbf{x}) = \mathbf{W}\mathbf{x}. \tag{4}$$

F is optimized by running stochastic gradient descent on the projection matrix $\mathbf{W}$:

$$\mathbf{W}_{t+1} = \mathbf{W}_t - \delta_t * \frac{\partial l}{\partial \mathbf{W}}. \tag{5}$$

In stochastic gradient descent, the training set is divided into a number of batches and $\mathbf{W}$ is optimized for every batch in an iterative fashion. The learning rate δ should be adapted in order for the stochastic gradient descent to reach an optimum. We achieve this by a bold driver technique: after gradient descent has taken place for all batches in a single iteration, we calculate the average loss for each batch. If the loss is less than in the previous iteration step we increase the learning rate by 5 %. Otherwise we reset the model parameters and set the learning rate to half of its previous value. This way the steps of the gradient descent grow bigger as long as the local optimum is still far away and decreases once the algorithm has come close to the local optimum.

After optimization, the matrix $\mathbf{W}$ is used to project the previously generated SIFT descriptors. The projected descriptors

are then clustered to produce the optimized visual vocabulary. A word image representation is then generated as is done in [5] by extracting a two level spatial pyramid from the optimized visual vocabulary.

IV. Experiments

A. Setup

We use the well-known George Washington database in order to show the usefulness of the proposed method. The database consists of a 20 page excerpt from a bigger collection of letters from George Washington and his associates (cf. [1], [4], [5]). Its corresponding ground truth contains a bounding box for each of the 4860 words.

For simplicity, we evaluate our method based on the segmented words from the database. However, our approach could be applied in a segmentation-free scenario as well. We use the experimental setup as proposed in [14]: each word image is used as a query to retrieve a ranked list of the remaining word images. The cosine distance is used as metric to compute the similarity between two word image representations. As a baseline, we use the method proposed in [5]: For each word image a Bag-of-Features representation is generated. This representation is used to generate a two-level spatial pyramid which is then re-weighted by applying the tf-idf transform. Please note that, opposed to the method presented in [5], we do not apply Latent Semantic Analysis as this has been shown to produce inferior results on this specific database. We report the results in terms of *Mean Average Precision (mAP)*. This value can be thought of as the average of the area under the precision-recall curve for each query and is a common performance measure in word spotting.

The visual vocabulary's size is set to 4096 as this value has been shown to be a good trade-off between model complexity and overall accuracy (cf. [5]). Using the two-level spatial pyramid design, each word descriptor vector has a dimensionality of 12288. The retrieval list of the initial word spotting run is cut off after ten words for each query. We found that cutting off at this number yields a high precision per query while generating a large number of descriptor pairs. For each query word and each retrieved word the three sets of descriptors are calculated. The second nearest neighbor ratio is set to 0.8 as is done in [12]. A descriptor pair passes the spatial test if its two descriptor's spatial distance is less than 0.15. Descriptor pairs are only considered in the following optimization if the number of positive matches is greater than 20. This way, descriptor pairs from wrong word pairs are not taking into consideration.

For optimization, we use the well known Theano toolbox [15]. Optimization is done by stochastic gradient descent with a batch size of 1000 and an initial learning rate of 10^{-5}. We use a bold driver technique as described in section III to adapt the learning rate during optimization.

B. Results

For an evaluation of the proposed approach, we examine the performance of the system with respect to changing margin ratios and projection dimensionality. For each data point, we execute five runs of model optimization and quantization. The results can be seen in figure 3. Increasing the margin ratio beyond 1.2 has a negative effect on the performance of the overall system. The reason for this is that the optimization process concentrates on seperating the RM set from the other two while failing to seperate the P and NM set. Furthermore, the second margin has next to no benefit on the optimization. A margin ratio of 1.2 is only marginally better than 1.0 which is equivalent to equal margins b_1 and b_2. Increasing the dimensionality higher than 64 has no or only marginal effect on the performance of the overall system (figure 3b). Though a target dimensionality of 64 leads to a slightly smaller mAP value than using 120 dimensions, the time for calculating the visual vocabulary decreases dramatically from around 82 minutes to less than 50 minutes on a standard Intel Core i7-3770 machine with 16 GB RAM.

TABLE I: Results for segmentation-based query-by-example word spotting on the George Washington database. The results for the proposed approach were obtained at a margin ratio of 1.2 and a subspace dimension of 64.

Method	mAP
non-rigid HOG [14]	44.59
Spatial Pyramid [5]	53.82
Proposed	56.93

Analyzing the retrieval result for each word, we found that especially shorter words had an increase in average precision while some longer words had a decrease in average precision. This might by due to the fact that longer words are more likely to contain identical characters than shorter words. Words that contain identical characters could have descriptor space nearest neighbors that are in the same relative position with respect to a given character but in different relative positions with respect to the word. This kind of descriptor pair would be assigned to the NM set while it actually belongs to the P set.

Finally, table I shows how our results compare to those of a Spatial Pyramid technique [5] and a deformable shape descriptor [14]. As can be seen in the table, the performance is improved by more than $27\,\%$ when compared to the deformable shape descriptors. The relative improvement with respect to the baseline is more than $5.7\,\%$.

V. Conclusion

In this work we presented a descriptor learning pipeline, tailor-made for query-by-example word spotting applications. Starting from a set of corresponding word images, we compute matching and mismatching descriptor pairs. A projection function is optimized to project the descriptor in a subspace in which matching descriptor pairs are pushed together while mismatching descriptor pairs are pushed apart. We evaluated our approach on the well-known George Washington database and demonstrated its usefulness. It proves to outperform the standard SIFT descriptor and the recently introduced non-rigid HOG in a query-by-example word spotting scenario. Not only does our approach produce more discriminative descriptors, but it is also able to decrease the number of dimensions used for each descriptor. This lowers the model's complexity and has a positive effect on the overall runtime of the system. Even though we evaluated our descriptor learning pipeline

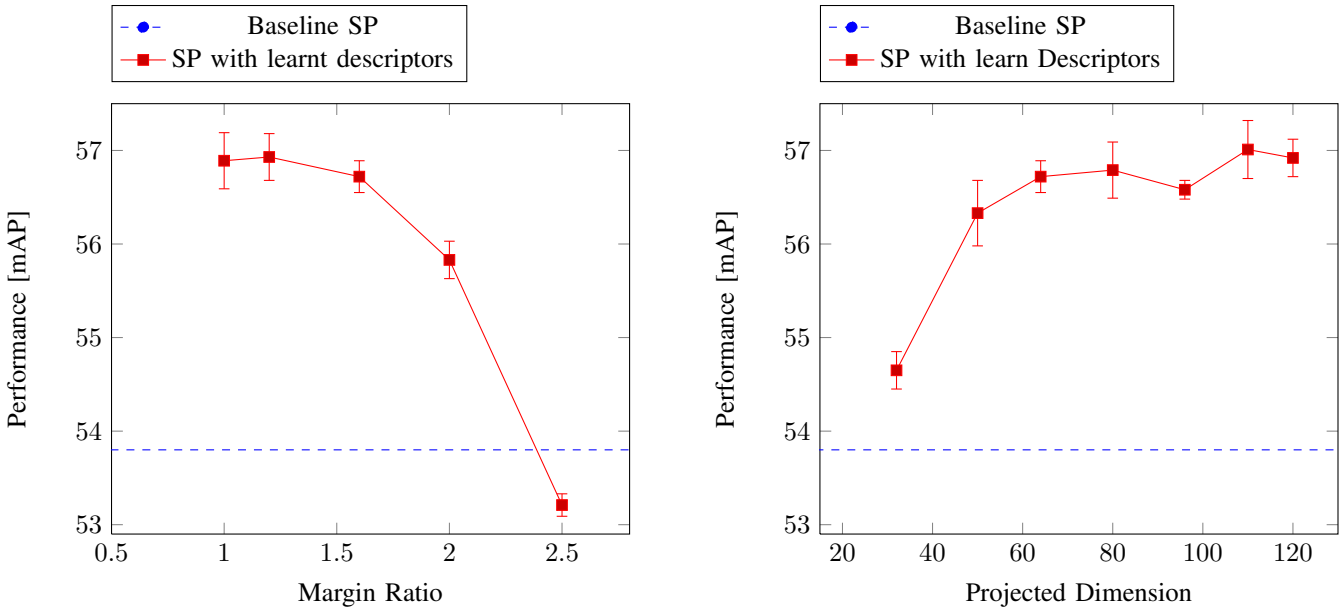

Fig. 3: The figure displays the varying performances for the two parameters evaluated. In a) the Mean Average Precision is plotted over the margin ratio with the dimension of the projected space set to 64. In b) the performance varying subspace dimensionality is shown at a margin ratio of 1.6. The blue plots indicate the results for the baseline spatial pyramid while the red plots show the results for the proposed method. .

in a segmentation based scenario, we are confident that the approach can be applied to segmentation-free word spotting applications with only minor modifications.

REFERENCES

[1] T. M. Rath and R. Manmatha, "Word Spotting for Historical Documents," *International Journal on Document Analysis and Recognition*, vol. 9, pp. 139–152, 2007.

[2] 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*, vol. 34, pp. 211–224, 2012.

[3] M. Rusiñol, D. Aldavert, R. Toledo, and J. Lladós, "Browsing Heterogeneous Document Collections by a Segmentation-Free Word Spotting Method," in *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, 2011, pp. 63–67.

[4] L. Rothacker, M. Rusinol, and G. A. Fink, "Bag-of-Features HMMs for Segmentation-Free Word Spotting in Handwritten Documents," in *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, 2013, pp. 1305–1309.

[5] M. Rusiñol, D. Aldavert, R. Toledo, and J. Lladós, "Efficient segmentation-free keyword spotting in historical document collections," *Pattern Recognition*, vol. 48, no. 2, pp. 545–555, 2015.

[6] D. Aldavert, M. Rusinol, R. Toledo, and J. Llados, "Integrating Visual and Textual Cues for Query-by-String Word Spotting," in *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, 2013, pp. 511–515.

[7] L. Rothacker, S. Vajda, and G. A. Fink, "Bag-of-Features Representations for Offline Handwriting Recognition Applied to {Arabic} Script," in *Proc. Int. Conf. on Frontiers in Handwriting Recognition*, 2012.

[8] R. Salakhutdinov and G. Hinton, "Learning a nonlinear embedding by preserving class neighbourhood structure," in *AI and Statistics*, vol. 3, 2007, pp. 412–419.

[9] A. Frome, F. Sha, Y. Singer, and J. Malik, "Learning globally-consistent local distance functions for shape-based image retrieval and classification," in *Proceedings of the IEEE International Conference on Computer Vision*, 2007.

[10] K. Mikolajczyk and J. Matas, "Improving descriptors for fast tree matching by optimal linear projection," in *Proceedings of the IEEE International Conference on Computer Vision*, 2007.

[11] G. Hua, M. Brown, and S. Winder, "Discriminant embedding for local image descriptors," in *Proceedings of the IEEE International Conference on Computer Vision*, 2007.

[12] J. Philbin, M. Isard, J. Sivic, and A. Zisserman, "Descriptor learning for efficient retrieval," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 6313 LNCS, 2010, pp. 677–691.

[13] D. Lowe, "Object recognition from local scale-invariant features," *Proceedings of the Seventh IEEE International Conference on Computer Vision*, vol. 2, 1999.

[14] J. Almazan, A. Fornes, and E. Valveny, "Deformable HOG-Based Shape Descriptor," in *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, 2013, pp. 1022–1026.

[15] F. Bastien, P. Lamblin, R. Pascanu, J. Bergstra, I. Goodfellow, A. Bergeron, N. Bouchard, D. Warde-Farley, and Y. Bengio, "Theano: new features and speed improvements," *Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop*, pp. 1–10, 2012.