Document Specific Sparse Coding for Word Retrieval

Ravi Shekhar and C.V. Jawahar
Centre for Visual Information Technology,
International Institute of Information Technology Hyderabad, India
Email: ravi.shekhar@research.iiit.ac.in, jawahar@iiit.ac.in

***Abstract*—Bag of words (BoW) based retrieval is an efficient method to compare the visual similarity between two images. Recognition free methods based on BoW have shown to outperform OCR based methods. We further improve the performance by defining a document specific sparse coding scheme for representing visual words (interest points) in document images. Our method is motivated by the successful use of sparsity in signal representation by exploiting the neighbourhood properties. In addition to providing insights into the design of the coding scheme, we also verify the method on two data sets and compare with the recent methods. We have also developed text query based search solution, and we report performance comparable to image based search.**

Keywords*—*Document Image Retrieval, Sparse Coding, Bag of Words.

I. Introduction

Retrieval of relevant document images to a given query word has emerged as an important topic of research in recent years. This has evolved as a successful alternative to the recognition based retrieval for handwritten as well as printed documents in many complex scripts. Functionally, these methods match the query word with a large set of words in the database and retrieve a ranked set of documents based on the similarity (or relevance) to the query. Performance of these recognition free retrieval schemes (often called word spotting) depends on the representation as well as the matching scheme used for measuring the similarity.

In word spotting, word images are represented by appropriate features. Similarity between words are measured with the help of an appropriate distance function. In [1], word images have been represented by profile features and Euclidean distance is naively used to compute the similarity. To take care of the variability in word image lengths, word images are matched using dynamic time warping (DTW) [1]. However, our focus in this work is to derive a retrieval scheme with an index structure in the back end. DTW based technique is not suitable for this purpose.

Scalability in document retrieval has also received significant attention in the recent past. In [2], 10 Million pages were indexed based on locally likely arrangement hashing (LLAH). In their work, pages are indexed and retrieval is done in sub-seconds. This method focuses on pages as query and their variations in imaging, specifically during the camera capture. Our focus is on accurate retrieval at word level. Recent attempts [3], [4] on robust document retrieval use visual Bag of Words (BoW) for representing and matching word images. With BoW representation, and an inverted indexing scheme, one can retrieve relevant documents from a Million documents in sub-seconds time [3]. The BoW based representation is motivated by its successful use in indexing textual documents. For representing images, feature points are quantized and a flexible representation is built by defining a "vocabulary" over a feature space (See Section II-A for more details).

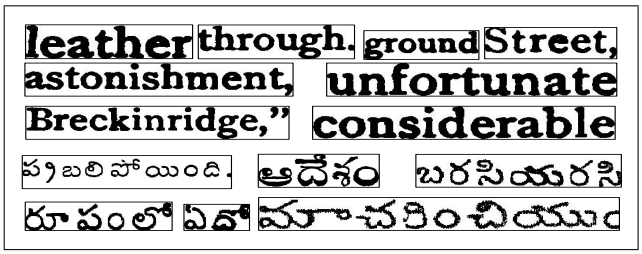

Fig. 1. Sample Images from dataset. Observe different variations in print and noise.

Scale-invariant feature transform (SIFT) [5] features computed at interest points are the most popular features for building BoW representation. In traditional BoW representation, each feature point is represented by exactly one "visual word". The word image is represented as a histogram of the visual words. The quantization of the features to a visual word results in loss of information. Often, this is considered to be introducing some level of robustness (or invariance). Even now, there is no consensus on deciding the size of the vocabulary and methods for learning the vocabulary. There are two basic steps in constructing a BoW representation, given a vocabulary. They are Coding and Pooling. We discuss these two steps in detail in Section II-B. It has been observed that the dictionary/vocabulary is over complete, and the problem of coding can now be formulated as a sparse coding problem with emphasis on learning codes that help in describing the visual content of the image. Many recent methods [6], [7] demonstrated the success of the sparse coding for effectively representing the visual content.

In this work, we have provided document specific sparse coding framework. First level coding is provided by using locality constraint at word level. By using locality constraint at word level, we are able to capture locality between different characters present in the word. As next level of constraint, we have tried to find occurrence of the frequent visual words within a character consisting different variations. Whenever these co-occurring present in the query and retrieved list, more weight is assigned.

As a side product of mining co-occurrence pattern in visual words, we also provide text query support. In text query support, weight of visual words corresponding to every character is calculated off-line. Using calculated values, text query visual word is formulated by summing all visual word weights.

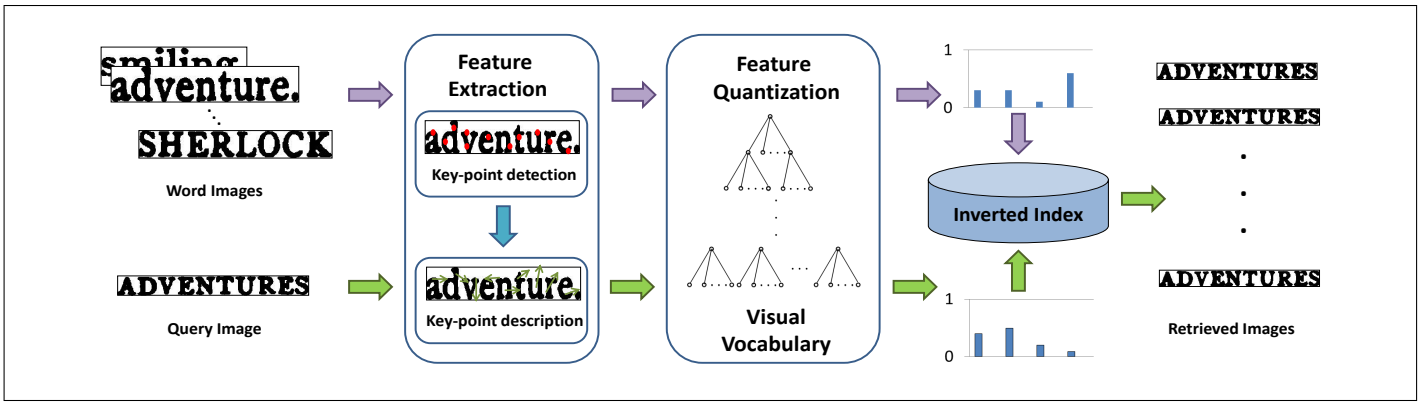

Fig. 2. Word image retrieval using Bag of Visual Words Framework: This process consists of offline and online parts. In offline process, features of word image are calculated and representation is learned using codebook. Learned representation is indexed using inverted file indexing structure. In online process, query images representation is determined and retrieval is performed based on similarity measure.

II. Document Specific Sparse Coding

A. Bag of Visual Words

Bag of Visual Words (BoW) representation [8] is inspired by bag of words representation in text retrieval. In text retrieval, each document is represented by an unordered set of non-distinctive words present in the document, regardless of the grammar and character order. Document is formally represented with the help of frequency of occurrences (histogram) of the words in the vocabulary. These histograms are then used to perform document classification and retrieval. Analogously, an image is represented by an unordered set of non-distinctive discrete visual features. The set of these discrete visual features is called codebook or visual vocabulary. This visual vocabulary is then used to quantize the extracted features by simply assigning the label of the closest cluster centroid. The final representation for an image is the frequency counts or histogram of the quantized features $[f_1, f_2, ...f_i, ..., f_v]$ where f_i is the number of occurrences of i^{th} visual word in the image and v is the vocabulary size. To account for the difference in the number of interest points between images (due to size etc.), the BoW histogram is normalized to have unit $L1$ norm. By representing an image as a histogram of visual words, one can obtain certain level of invariance to the spatial location of objects in the image. However, this creates certain issues in document image representation. All word images containing same characters will have same histogram representations due to the lack of order/structure in the representation. In natural scene images, spatial order is provided by spatial pyramid matching (SPM) [9], which divides images into vertical and horizontal directions subsequently. To provide order in representation, word images are divided vertically in three parts [3] and then indexed. Fig. 2 explains word image retrieval using BoW.

For interest point detection, we have used Harris corner detector which is proven better representation of both natural images [10] and word images [3]. At each of these interest points, we extract a SIFT descriptor to represent the local information as a vector of gradients. SIFT extracts features that are invariant to changes in scale and rotation and are robust to changes in illumination, viewpoint, noise and affine distortion. SIFT points are stable local grey-scale minima and maxima. In SIFT, a neighbourhood is described by a histogram of weighted gradients within a window to yield a 128 dimensional vector using its location, magnitude and orientation. Vocabulary creation is done using computationally efficient Hierarchical K-Means (HKM) [11]. This algorithm clusters the data into C clusters first and then samples in each of these clusters are clustered again recursively. This process is continued until we obtain the required number of clusters. In this work, HKM is preferred over K-means due to its computational efficiency by taking time in $O(logn)$ instead of $O(n)$ for codebook of size n.

B. Coding and Pooling in BoW

The purpose of quantization is to reduce the cardinality of the representation space. Feature descriptor X has an infinite set of possible values and it is restricted to a infinite set of possible vectors when mapped to the codebook. In general, the cost of computing a quantization is $O(m)$, where m represents the cost of computing a single distance. In practice, the most used distance is the squared Euclidean distance, which is expensive since it implies $2q$ operations for feature of dimension q and yields a total cost of $O(mq)$.

Traditionally, vector quantization (VQ) is used to generate code from raw descriptors. It solves the following constrained least square fitting problem:

$$\begin{aligned} &\underset{c}{\operatorname{argmin}} \Sigma_{i=1}^{N} \parallel x_i - Bc_i \parallel^2 \\ &s.t. \parallel c_0 \parallel_{l^0} = 1, \parallel c_i \parallel_{l^1} = 1, c_i \geqslant 0, \forall i \end{aligned} \tag{1}$$

where $X = [x_1, ..., x_N]$ is the descriptor of the image, B is the codebook, $C = [c_1, ..., c_n]$ is the set of code for the image X and N is the number of data points. The optimization goal of VQ is to find a quantized vector code C with a single non-zero element which is approximately equal to X.

One limitation of the codebook approach is the hard assignment of codewords in the vocabulary to image feature vectors. The hard assignment gives rise to two issues: codeword uncertainty and codeword plausibility. Codeword uncertainty is the problem of selecting the correct codeword out of two or more relevant candidates. The VQ approach selects the best representing visual word, ignoring the relevance of other candidates. Codeword plausibility denotes the problem of selecting a codeword without a suitable candidate in the

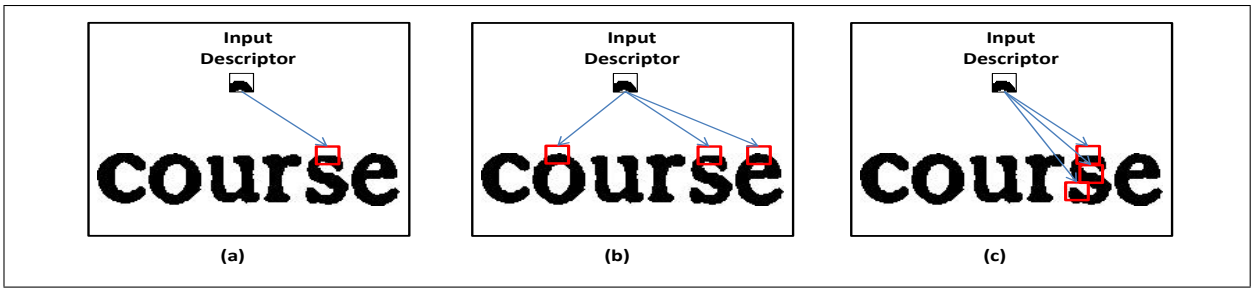

Fig. 3. Locality in Character Space (a) Vector Quantization (VQ) (b) Sparse Coding (SC) (c) Locality Coding. In VQ, each descriptor is assigned to single visual word while in SC, it is assigned to multiple visual words but locality of visual word is lost. In locality coding, each descriptor is assigned to multiple visual words by considering locality of visual words.

vocabulary. The codebook approach assigns the best fitting codeword, regardless of the fact that this codeword is not a proper representative. To overcome this, soft assignment [12], [13] is proposed. A soft assignment coding assigns a local feature to all visual words based on locality. The coding coefficient represents the membership of a local feature to different visual words. The major limitation of this approach is that it can not minimize the reconstruction error.

C. Sparsity Due to Locality in Coding

The basic idea of sparse coding (SC) is to represent a feature vector as linear combination of few bases from a predefined dictionary, hence induce the concept of sparsity. Sparse coding provides low-dimensional approximation of a given signal in a given basis set. Unlike principal component analysis (PCA), sparse coding does not constraint the orthogonality of bases and it turns out that more flexibility is given to adapt the non-linear representation of the data.

However, VQ method generates quantization loss, and is poor in scalability. In order to improve scalability and reduce quantization loss, Sparse coding based Spatial Pyramid Matching (ScSPM) [14] method is proposed to introduce sparse coding to SPM procedure, obtaining non-linear feature representation that works better with linear classifiers. Here, the coding problem becomes a standard sparse coding (SC) problem as follows,

$$\underset{c}{\operatorname{argmin}} \Sigma_{i=1}^{N} \parallel x_i - Bc_i \parallel^2 + \lambda \parallel c_i \parallel_{l^1} \tag{2}$$

where λ is regularization parameter. SC minimizes reconstruction error in equation (2). SC utilizes an over-complete dictionary to linearly reconstruct a data instance. Standard SC with $L1$-norm regularization produces a sparse coefficient vector, it has no control over which attributes to be zeroes (or non-zeroes); in other words, SC might reconstruct a query image by training data from distinct images, and thus is not preferable for the task of classification and retrieval. SC optimization is computationally expensive and its regularization term is not smooth.

A locality constrained linear coding (LLC) [6] is proposed as an alternative to SC, which learns a data representation using nearest codeword. LLC is an adaptation of sparse coding with locality constraints. It has several advantages over sparse coding and vector quantization (VQ). Instead of using sparsity constraint, in LLC, a locality constraint is incorporated into the optimization goal as follows,

$$\underset{c}{\operatorname{argmin}} \Sigma_{i=1}^{N} \parallel x_i - Bc_i \parallel^2 + \lambda \parallel d_i \odot c_i \parallel_{l^1} \quad s.t. 1^T i = 1, \forall i \tag{3}$$

where $\odot$ denotes element-wise multiplication and d_i is the locality adapter which gives freedom for each basis vector proportional to its similarity to the input descriptor x_i. An LLC procedure has three steps: First, for each input descriptor x_i , its K-Nearest Neighbours can be denoted as B_i . Then, x_i can be approximately reconstructed using the set of B_i . At last, the input descriptor x_i is represented using the corresponding parameter c_i for each code in the codebook B. In this way, we use a vector to represent the input image, which is as large as the codebook, no matter what the size of the extracted feature descriptors is.

In VQ coding, each input is coded using only one most similar element from the codebook which leads to large quantization error. In SC and LLC, each input is represented by multiple elements from the codebook, which can better represent the inputs. Furthermore, by applying locality constraint, LLC captures the correlations between similar descriptors. One of major advantages of LLC over SC is that LLC provides approximate solution, which is very computationally efficient compared to SC.

To improve the performance further, we have provided character level coding. It is observed that for a given character, certain visual words are always present. During retrieval process, if we find out that both query and retrieved images are having those visual words present, weight of corresponding visual words is increased by a factor of two. By this, we are able to capture more context at character level.

In BoW, a single descriptor is assigned to a single visual word. Due to quantization error, it is possible that similar descriptors being assigned to different visual words. Also, in case of a descriptor having equal distance with two or more visual words, randomly one of them is assigned. A major limitation of SC-based approaches for classification is that similar data instances are not assured of producing similar coding results.

Fig. 3 explains locality in terms of character space with respect to a given word. It can be observed that in BoW representation input patch value is matched with the best possible match, while there are other candidate possible patches. For sparsity, similar patch value is assigned but it does not ensure the exact reconstruction. In case of locality, it can be observed

that reconstruction of input patch will be better compared to sparsity. Apart from providing better representation of characters, locality constraint captures locality at both word and character levels, which helps to make the retrieval process more accurate.

III. Experimental Results and Discussions

In this section, we will discuss experimental details to show the performance of the proposed method.

A. Experimental Setup

We have used printed books of two languages, namely English and Telugu, for evaluation purpose. English book (English-1601) is "Adventures of Sherlock Holmes" written by Arthur Conan Doyle and Telugu books (Telugu-1716 and Telugu-1718) consist details of Tirupati, a pilgrimage city located in Andhra Pradesh, India and are published in 18^{th} century. Details are provided in the Table I and sample word images are shown in Fig. 1. All the books are annotated at the word level and ground truth was created using technique described in [15].

To evaluate the quantitative performance, multiple query images were generated. The query images are selected such that (i) They have multiple occurrences in the database, (ii) They are mostly functional words and (iii) They have no stop words. The performance is measured by mean Average Precision (mAP). mAP is the mean of the area under the precision-recall curve for all the queries. For English query selection, upper and lower case letters are treated differently, because in image domain, appearance of words will be different i.e, "Holmes" and "HOLMES" are treated as different queries but "Adventure" and "adventure" are treated same because only one character is of different case.

TABLE I. Books Used for the Experiments.

Book	#Pages	#Words
English-1601	363	113008
Telugu-1718	100	21345
Telugu-1716	120	4121

Visual word generation is an important step in quantization based retrieval. In natural scene retrieval, it is observed that large dictionaries perform better [16]. In case of word images, number of possible neighbourhoods is very less compared to natural scene images. Therefore, for a large dictionary, it is very likely that similar words can have different visual words. A dictionary of size $10K$ is generated from selected parts of different books. On this data, keypoint detector and descriptor are applied and then quantization is performed. This dictionary is used for the rest of the word images to find corresponding visual words and their weights.

B. Indexing and Retrieval

For indexing purpose, we have used inverted file indexing. In inverted file structure, for each visual word, a list of corresponding word images is stored in the form of $< visualWord, wordImage_i >$. In retrieval process, visual word corresponding to the query image is used to look up the index file. "tf-idf" weighting scheme is used for similarity matching. Sim_Score (ref. Equation 4) assigns a high weight to a term, if it occurs frequently in the document but rarely in the whole document collection. To compensate the spatial configuration lost during quantization, we re-order the retrieved list using the character order in the query word image. Longest common sub-sequence (LCS) [4] re-ordering on visual word occurrence provides better query matching. The final order of ranked retrieved list is formed using linear combination of Sim_Score and LCS_Scores (ref. Equation 6):

$$Sim\_Score(Q,I) = \frac{\sum_{i \in Q \cap I} w_i}{\sum_{j \in I} w_j} \tag{4}$$

$$w_i = \frac{1}{log(f_i + 1)} \tag{5}$$

$$LCS\_Score(Q,I) = \frac{\sum_{i \in LCS(Q,I)} w_i}{\sum_{j \in Q} w_j} \tag{6}$$

$$\begin{aligned} Score(Q,I) = [\gamma \times Sim\_Score(Q,I)] + \\ [(1-\gamma) \times LCS\_Score(Q,I)] \end{aligned} \tag{7}$$

where, Q and I are visual words corresponding to query and candidate images respectively, w_i is weight of the i^{th} visual word, f_i is the frequency of the i^{th} visual word in the book and γ is a weighting parameter.

C. Performance Evaluation

We have developed text query based search engine for document image search. Here, all queries are given in text as provided in text based search engines like 'Google' and 'Bing' etc. Based on characters provided in the search engine, visual words of query are formulated based on already learned visual words of characters. Using formulated query visual words, text query index is queried to generate initial set of results. Due to segmentation, inter correlation between characters in word is lost, which results in low recall but high initial precision. So, to improve the recall of the system, word images are also indexed and queried based on query expanded result of text query part. By this, we overcome the low recall and performance. For English, we are able to achieve mAP of 0.87 and for Telugu 0.90, which are comparable to query by example.

Now, we explain [3] and [4] with respect to our implementation and query set. Table II shows results on various implementations and we observed the following pattern in it: Both the methods give the same initial result, i.e., using only BoW. The main difference lies in the re-ranking methodology. Though SIFT based re-ranking eliminated quantization, it failed to contain the character order information which the LCS does. As the order of every character is fixed in a given word image, LCS eliminates the need of spatial verification at indexing level and thereby performs better when compared to SIFT based re- ranking. A linear combination of these two scores (ref. Equation 7) preserves the order provided by initial and re-ranked list and boost the mAP by 4-5%. Next, baseline results based on document specific sparse coding are shown. In document specific sparse coding, instead of taking only one nearest neighbour, more than one visual word are considered according to locality. Notion behind this is, it not only gives better representation in terms of character representation by considering its locality but also minimizes quantization error

TABLE II. MAP OF DIFFERENT METHODS ON DIFFERENT BOOKS IN DATASET.

Book	BoW	BoW + SIFT Re-ranking [3]	BoW + LCS Re-ranking [4]	Doc. Coding	Doc. Coding + LCS Re-ranking
English-1601	0.8015	0.8645	0.92	0.8765	**0.9451**
Telugu-1718	0.7834	0.8861	0.918	0.92	**0.96**
Telugu-1716	0.8173	0.8531	0.9036	0.91	**0.95**

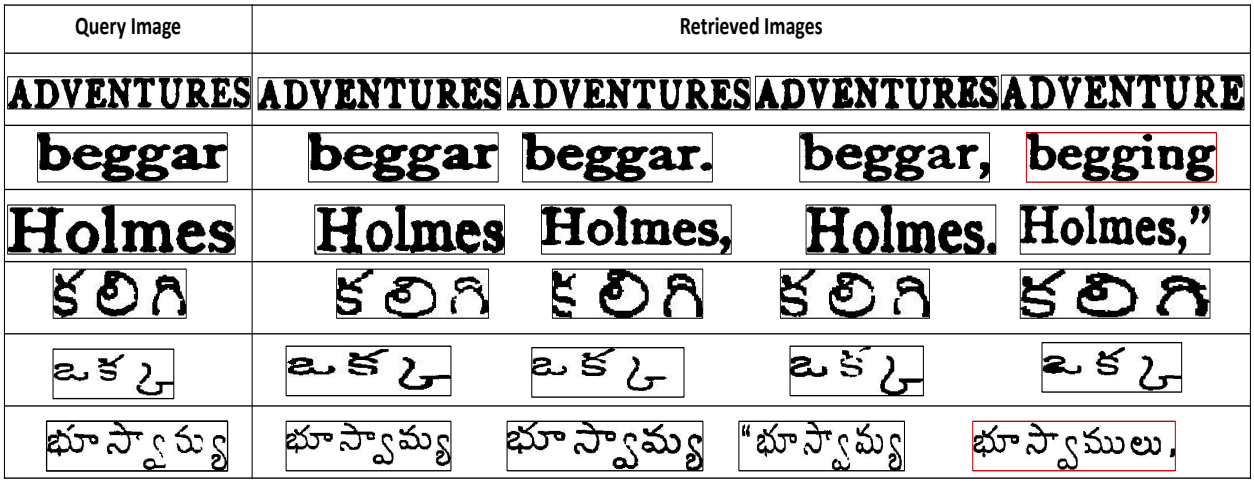

Fig. 4. Sample Retrieved Word Images: Left column shows query image and right column shows its corresponding retrieved word images in the order of retrieved rank. Note that partially correct retrieved words are shown in brown color.

as can be observed in Table II. In our implementation, number of neighbours used is 3. We can observe that document specific sparse coding with inverted file index performs better than BoW. On the retrieved list, we have performed LCS based re-raking and we have reached mAP of 0.93-0.96. It can be observed from Table II, document specific coding consistently outperforms previous methods on different datasets. Fig. 4 shows the qualitative performance of the proposed method on sample images from datasets. In retrieved list, we can observe different variations present. Also, some retrieved words are partially matched with the query words and appear in the top of the ranked retrieved list. Retrieval time for a given query is in sub-seconds of time.

IV. CONCLUSION AND FUTURE WORK

In this work, an efficient document specific coding technique is proposed. Proposed method takes advantage of the property of characters that each character contains specific set of descriptor in its locality. We exploit this fact and assign each feature descriptor to multiple visual words based on its locality in character space. Experimentally, results are shown on English and Telugu scripts and we achieve high performance. Future work includes selecting codebook specific to particular to documents by using semi supervised techniques and designing document specific descriptor.

ACKNOWLEDGEMENT

This work was partly supported by Ministry of Communication and Information Technology, Government of India.

REFERENCES

[1] T. M. Rath and R. Manmatha, "Word spotting for historical documents," *IJDAR*, 2007.

[2] K. Takeda, K. Kise, and M. Iwamura, "Real-time document image retrieval for a 10 Million pages database with a memory efficient and stability improved LLAH," in *ICDAR*, 2011.

[3] R. Shekhar and C. V. Jawahar, "Word Image Retrieval Using Bag of Visual Words," in *DAS*, 2012.

[4] I. Z. Yalniz and R. Manmatha, "An Efficient Framework for Searching Text in Noisy Document Images," in *DAS*, 2012.

[5] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *IJCV*, 2004.

[6] J. Wang, J. Yang, K. Yu, F. Lv, T. S. Huang, and Y. Gong, "Locality-constrained Linear Coding for image classification," in *CVPR*, 2010.

[7] X. Zhou, K. Yu, T. Zhang, and T. S. Huang, "Image Classification Using Super-Vector Coding of Local Image Descriptors," in *ECCV*, 2010.

[8] J. Sivic and A. Zisserman, "Video Google: A Text Retrieval Approach to Object Matching in Videos," in *ICCV*, 2003.

[9] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *CVPR*, 2006.

[10] C. Schmid, R. Mohr, and C. Bauckhage, "Evaluation of Interest Point Detectors," *IJCV*, 2000.

[11] D. Nister and H. Stewenius, "Scalable Recognition with a Vocabulary Tree," in *CVPR*, 2006.

[12] J. van Gemert, J.-M. Geusebroek, C. J. Veenman, and A. W. M. Smeulders, "Kernel Codebooks for Scene Categorization," in *ECCV*, 2008.

[13] J. van Gemert, C. J. Veenman, A. W. M. Smeulders, and J.-M. Geusebroek, "Visual Word Ambiguity," *PAMI*, 2010.

[14] J. Yang, K. Yu, Y. Gong, and T. S. Huang, "Linear spatial pyramid matching using sparse coding for image classification," in *CVPR*, 2009.

[15] C. V. Jawahar and A. Kumar, "Content-level Annotation of Large Collection of Printed Document Images," in *ICDAR*, 2007.

[16] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Object retrieval with large vocabularies and fast spatial matching," in *CVPR*, 2007.