

Document Image Retrieval using Bag of Visual Words Model

Thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science (by Research)
in
Computer Science Engineering

by

Ravi Shekhar
201007008

ravi.shekhar@research.iiit.ac.in



Center For Visual Information Technology
International Institute of Information Technology
Hyderabad - 500 032, INDIA
June 2013

Copyright © Ravi Shekhar, 2013
All Rights Reserved

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled “Document Image Retrieval using Bag of Visual Words Model” by Ravi Shekhar (201007008), has been carried out under my supervision and is not submitted elsewhere for a degree.

Date

Adviser: Prof. C. V. Jawahar

To My Family and Friends

Acknowledgments

I am deeply indebted to my advisor Prof. C. V. Jawahar for his dedication, encouragement, motivation and also for his inspiring guidance and supervision throughout my thesis work. I would also like to thank document understanding research groups at Centre for Visual Information Technology (CVIT), who had made great contribution by sharing ideas, comments and materials. My dearest thanks goes to Anand, Pramod, Praveen and Naveen for their invaluable suggestions and kindness to help me. I am grateful to Phani and annotation team for providing dataset and annotation. I give my warmest gratitude to Satyanarayana (CVIT) for his unreserved support related to administrative matters.

I would like to thank all my classmates not only for their support during research also for making life at IIIT-H memorable one. Special thanks goes to Sushma, not only for correcting my English every time but also helping through out my research. Swagatika, Sumit, Ravi thanks for making life enjoyable during my stay here.

I would also like to thank Shiva, Vivek and Shashi for encouraging to take research as career. I would also like to express my gratitude towards everyone in my family. It would not have been possible to even dream of studying further without their support.

Abstract

With the development of computer technology, storage has become more affordable. So, all traditional libraries are making their collections electronically available on internet or digital media. To search in these collections, better indexing structure is needed. It can be done manually or automatically. But manual indexing is time consuming and expensive. So, automation is the only feasible option. First approach in this direction was to convert document images into text by OCR and then perform traditional search. This method works well for European languages whose optical character recognizers (OCRs) are robust but did not work reliably for Indian and non-European languages like Hindi, Telugu and Malayalam etc. Even for European languages, OCRs are not available in the case of highly degraded document images.

To overcome limitations of OCR, word spotting was emerged as an alternative. In word spotting, given word image is represented as features and matching is done based on feature distance. Main challenges in word spotting are extraction of proper features and feature comparison mechanism. Profile feature is one of the popular ways to represent word image and Euclidean distance is used for comparison. To overcome word length, all images are scaled to same size, due to which lots of information is lost. Later, DTW based approach was proposed for matching which does not require scaling of images. The problem with DTW based approach is that it is very slow and can not be used for practical and large scale application.

In first part of this thesis, we explain a Bag of Visual Words (BoVW) based approach to retrieve similar word images from a large database, efficiently and accurately. We show that a text retrieval system can be adapted to build a word image retrieval solution. This helps in achieving scalability. We demonstrate the method on more than 1 Million word images with a sub-second retrieval time. We validate the method on four Indian languages, and report a mean average precision of more than 0.75. We represent the word image as histogram of visual words present in the image. Visual words are quantized representation of local regions, and for this work, SIFT descriptors at interest points are used as feature vectors. To address the lack of spatial structure in the BoVW representation, we re-rank the retrieved list. This provides significant improvement in the performance.

Later, we have also performed enhancements over BoVW approach. Enhancements are in terms of query expansion, text query support and Locality constrained Linear Coding (LLC) based retrieval system. In query expansion, we have used initial results to modify our initial query and obtained better results. In BoVW model, query is given by example but users are generally interested in query as text

like “Google”, “Bing” etc. Text query is also supported by same model as BoVW. Later, LLC is used to achieve high recall. LLC scheme learns a data representation using nearest codeword and achieves improvement over performance.

In most of the scalable document search, features like SIFT, SURF are used. These are originally designed for natural images. Natural images have lots of variation and extra information in terms of gray images. Document images are binary images, so need features which can be specifically designed for them. We have proposed patch based features using profile features. We have compared our proposed feature with SIFT and obtained similar performance. Our feature has advantage that it is faster to compute compared to SIFT, which makes our pre-processing step very fast.

We have demonstrated that recognition free approach is feasible for large scale word image retrieval. In future work, similar approach can be used for hand written, camera based retrieval and natural scene text retrieval etc. Also, a better and efficient descriptor is needed specifically designed for document words.

Contents

Chapter	Page
1 Introduction	1
1.1 Problem Overview	3
1.2 Contribution	4
1.3 Thesis Organization	4
2 Background and Previous Work	5
2.1 Optical Character Recognition	5
2.2 Word Spotting	8
2.3 Feature Extraction/Representation	11
2.3.1 Profile Feature	11
2.3.2 Shape Context	11
2.3.3 Scale Invariant Feature Transform	11
2.3.4 Speeded Up Robust Features	13
2.4 Summary	13
3 Word Image Retrieval using Bag of Visual Words	14
3.1 Bag of Words	15
3.2 Bag of Visual Words	17
3.2.1 Spatial Verification	18
3.2.2 Re-ranking	19
3.2.2.1 SIFT Based Re-ranking	19
3.2.2.2 LCS Based Re-ranking	20
3.3 Retrieval System Overview	20
3.4 Experiments and Results	21
3.4.1 Dataset Details	21
3.4.2 Results and Discussion	22
3.4.3 Use of Search Engine	25
3.5 Summary and Comments	26
4 Enhancement over Bag of Visual Words based Word Image Retrieval	27
4.1 Query Expansion	28
4.2 Text Query Support	28
4.3 Coding Techniques	30
4.3.1 Vector Quantization	31
4.3.2 Sparse Coding	31

4.3.3	Locality Constrained Linear Coding	31
4.4	Experiments and Results	32
4.4.1	Dataset Details	33
4.4.2	Results and Discussion	33
4.5	Summary and Comments	35
5	Beyond SIFT: Quantized Feature for Robust Document Retrieval	37
5.1	Proposed Feature	39
5.1.1	Projection Profile	39
5.1.2	Bakground/Ink transition	39
5.1.3	Word Profile	40
5.1.4	Proposed Feature Calculation	40
5.2	Fast Pre-processing	41
5.3	Experiments and Results	42
5.3.1	Experimental Setup	42
5.3.2	Results and Discussion	42
5.4	Summary and Comments	44
6	Conclusions and Future Work	45
6.1	Summary and Conclusion	45
6.2	Future Work	46
	Bibliography	48

List of Figures

Figure	Page
1.1 Sample word images from dataset having different types of degradation.	2
2.1 Basic Architecture of OCR.	6
2.2 SIFT descriptor calculation based on gradient of image.	12
3.1 Bag of Words framework in text domain.	15
3.2 Bag of Visual Words Representation. Left: Input Images. Middle: Feature Extraction. Right Upper: Code Book Generation. Right Lower: Histogram Computation.	17
3.3 Codebook Generation Using Hierarchical K Means.	18
3.4 Spatial Verification: Image is divided into 3 parts to provide spatial order for Visual Words.	19
3.5 A word image and its visual words projected on X-axis.	20
3.6 Overview of the indexing and retrieval.	21
3.7 Sample results for All languages. First column shows the query words. Their retrieved images are shown according to the decreasing order of rank.	23
3.8 mAP Vs Query length. Also see the effect of re-ranking.	24
3.9 Sample retrieved list for degraded Images in the decreasing order of rank.	25
4.1 Overview of Query Expansion. Here one query image and its corresponding retrieved list are shown. Green are correct and Red are wrong. Observe that after re-fined query histogram using QE, all the top results are correct.	28
4.2 Overview of Text Query Support.	29
4.3 An example showing the problems of visual word ambiguity in the codebook model. The small dots represent image features, the labeled red circles are visual words determined by unsupervised clustering. The triangle represents a data sample that is well suited to the codebook approach. The difficulty with visual word uncertainty is shown by the square, and the problem of visual word plausibility is illustrated by the diamond [66].	30
4.4 Comparison among VQ, SC and LLC [68]	32
4.5 Sample retrieved word images based on different methods, which have same rank. Note that Text Query give all clean results on the top. It happens because, for Text Query cleaned characters are used for visual word learning.	34
5.1 Block Diagram for Proposed Feature Calculation	38
5.2 Fast Pre-processing Process.	41

5.3 Sample Retrieved Word Images in Order of Retrieved Rank. Note that, partially correct
retrieved words are shown in red color. 43

List of Tables

Table	Page
3.1 Books Details Used for the Experiments.	22
3.2 Precision@10 Statistics on 4 different languages.	22
3.3 mAP Statistics on 4 different languages	24
3.4 Average Retrieval Time and Space taken based on Dataset Size.	24
4.1 Dataset: Used to show Enhancement over BoVW.	33
4.2 Performance Statistics with Query Expansion and Text Query Support.	33
4.3 Baseline Results.	35
4.4 Baseline Results using LLC on Telugu-1716 and Telugu-1718.	35
4.5 Text Query with LLC Based Results on Telugu-1716 and Telugu-1718.	35
5.1 Dataset Used in the Experimentation.	42
5.2 Baseline Results on Proposed Feature	43
5.3 Performance Statistics on Different Books.	43
5.4 Comparison between SIFT and Proposed Feature time taken during feature calculation and quantization.	44

Chapter 1

Introduction

With storage becoming cheaper and imaging devices becoming increasingly popular, efforts are made to digitize and archive large quantity of multimedia data (text, audio, image and video). Lots of research is going in the direction for providing as simple as possible data access to end users. In text domain, significant progress happened in that direction. We have search engines like ‘Google’, ‘Bing’ etc which provide text search over the web. But on the other hand, search multimedia content is in early research stage. Most of multimedia contents are retrieved based on tag stored in it. Not only multimedia contents but old manuscripts, books are being digitized. However access to these contents is still a big challenge.

In 1971, Project Gutenberg [4] is launched. The main aim of this project was to provide everyone access to famous and important books, manuscripts. They have converted content of books into text and made them freely available to everyone. Here, searching is done on text level. Later, many digital library projects came up like Universal Library (UL) [3] and Digital Library of India (DLI) [1] etc. The aim of these digital libraries is to digitize all literary, artistic, and scientific works for better access to traditional materials, easier preservation, and make documents freely accessible.

Effective access to these libraries is limited by the lack of efficient retrieval schemes. The use of text search methods requires efficient and robust OCRs. For European languages, many efficient and robust OCR are available [16], for example, OmniPage pro from Caere, FineReader from ABBYY, Text Bridge from Xerox and Capture from Adobe etc. But non-European languages like Indian languages do not have robust and efficient OCRs [58]. Due to unavailability of efficient and robust OCRs, searching in image domain is explored. In this research work, we present efficient and scalable approaches that enable us in accessing the content of document image collections in the image domain itself.

Retrieval by Recognition

This method is based on using a recognizer to convert an image to text and then searching the results using a text search engine. Text similarity search is well developed. It involves looking for some simple measures, like edit distance, and deciding whether to produce that word in the result. In the text domain, grouping or clustering of text words to build index structures is simple. An example of recognition



Figure 1.1 Sample word images from dataset having different types of degradation.

based techniques is gHMM approach by Chan et al. [18], suggested for printed and handwritten Arabic documents. The performance of recognition based search techniques depends on the accuracy of the recognizer. Words that do not exist in the language are generated by the results of erroneous recognizers. This makes the search engines consider these words as unique. As a result, the index size is increased by words that do not exist in the language and which are never searched. Search in document images converted into text is challenging, as there are no robust OCRs available [46] for most Indian languages.

Retrieval by Image Matching

The search in a collection of words involves looking for similar words [63, 38]. The simple solution is to match the query word image with all the words in the collection and report the similar ones. Direct matching of image pixels may fail even in the presence of small changes in sizes of images [48]. Moreover, there will be a number of variations occurring in words due to artifacts, digitization process and size changes etc [49, 12]. Searching similar words in a collection is challenging. However, many methods have overcome this problem with the invention of new techniques. The word images are represented by a set of features [47], usually vectors or sequences of numeric values. These representations are used for word image matching. Thus, similar words can be searched using the matching the technique. A similarity measure that looks for approximate matches, to some extent, may be one of the methods for image matching. This method of searching similar words by image matching technique is expensive. As a result of matching query with every word of the collection, the search may take more time to generate the results. Hence, exhaustive search techniques are not suitable for search in word image collections.

Word Image Representation

We observe that the words in document images are represented by a set of features. Usually, features are vectors of numeric values. The features used for representing the objects in different applications may differ for similar objects. The change in the feature values would be due to a change in the ap-

pearance of the objects or the viewing angle. Even in document images, the appearance of words may change due to a number of factors that arise at the time of digitization. The print style of the documents and the type of font also affect the representation. The representation of similar words printed in different fonts vary considerably. The other factors that affect the appearance and representation of words are (see Figure 1.1):

- Excessive dusty noise on the paper.
- Large ink-blobs joining disjoint characters or components of characters.
- Vertical cuts due to folding of the paper.
- Cuts in arbitrary direction due to paper quality or foreign material.
- Degradation of printed text due to the poor quality of paper and ink.
- Floating ink from facing pages.

1.1 Problem Overview

Available document image retrieval techniques are not feasible for collections of document images. Most of the existing techniques based on complex image matching methods are not good enough for the huge collection of document images. The major hurdles in achieving efficient search are the processing time and the word image matching techniques [48].

Many possible solution are proposed in literature to provide content level access. Most of these solutions are query by example. A sample image is provided as a query and similar word level images are retrieved. But the end user demands textual query support like in popular search engines ‘Google’ and ‘Bing’ etc.

To index a large collection of books, a better representation is needed. The word matching methods are expensive and become more expensive when words are represented by high dimensional feature vectors [48]. High dimensional feature vectors not only pose problem for matching it also increase the pre-processing time for indexing. For huge datasets it, but become even critical. Representations for similar words, which may vary due to variations in image sizes, degradations like noise etc, should be similar such that it produces better results which matching.

In our work, we assume that pre-processing of document images and segmentation of text images into words, that are required for the recognition as well as for the retrieval tasks are available. The study of pre-processing and text-block segmentation are beyond the scope of the present work.

1.2 Contribution

In this research work, we focus on the retrieval from document image collections. Some of our contributions are the following:

- Proposed an efficient word image matching scheme based on Bag of Visual Word representation to effectively compare printed document images. Our method is highly language independent and scalable. The efficiency of proposed method is shown experimentally on four Indian languages. We have demonstrated the scalability of the method using 1 Million word images. We also benchmarked our system on noisy text.
- Provided enhancement over BoVW representation, in term of Query Expansion (QE), Text Query support and improvement over coding technique using Locality Constrained Linear Coding (LLC). QE is provided over retrieved list. Till now, query by example is being used for retrieval. We tried to provide textual support to the system. LLC is being used for improving coding techniques.
- Proposed patch based word image representation, which takes advantage that document images are binary. Proposed representation is robust to popular documents degradation, cuts and merges etc. Performance shown is comparable to popular feature representation SIFT.

1.3 Thesis Organization

This thesis is organized in five chapters. The first chapter provides an overview of the general background and the problem setting. Motivation behind the present work and the major contributions are also briefly described. In chapter two, we present the state of the art in the area of Optical Character Recognition and Document Image Retrieval System.

An efficient technique for word image retrieval is presented in Chapter 3. We have presented a document retrieval system based on BoVW. To show language independence, we have experimented on 4 different Indian languages. We have demonstrated the scalability of the method using 1 Million word images. Retrieval is done in sub-seconds.

Enhancement over BoVW technique is presented in Chapter 4. Query Expansion and Text query support is proposed. Also, further improvement over coding technique is proposed using Locality Constrained Linear Coding (LLC).

In Chapter 5, patch based feature is proposed for document retrieval system. We have used image patch for representing word image and performed retrieval based on feature calculated on those patches.

Finally, Chapter 6 contains the summary of contributions of this work, possible scope for the work in future and final conclusions.

Chapter 2

Background and Previous Work

Nowadays a huge collection of handwritten and printed documents are electronically available on the Internet and many of them are stored in digital libraries. For example, Digital Library of India (DLI) [1] and Universal Library (UL)[2] [3] are the two digital libraries which store scanned documents of different languages. Searching these images is practically not possible, if there is no textual representation or the ground truth available. All these factors made fast access to the contents of the document images important and challenging research problem. Initially, search was done by manual intervention, i.e., documents are indexed by some person. This process has disadvantages like it is expensive and there is high probability of errors which make the entire procedure untrustworthy. Later, Optical Character Recognition (OCR) based systems have been used. In OCR, document images are converted into text and then these texts are used for searching purpose. This method works well for European languages whose OCR are robust but does not work reliably for non-European languages like Hindi, Malayalam, Bangala and Telugu etc. Even for European languages, OCR are not available for highly degraded document images. Later, direct image domain matching is proposed. In this method, word images are represented using features and retrieval is done by comparing features. In this direction, significant work is done in [49], where handwritten word images are represented using profile feature and retrieval is done based on Dynamic Time Warping (DTW). DTW based method performs well for handwritten documents but takes 1 sec to compare two word images [49], which makes it in-feasible for large collection of documents.

2.1 Optical Character Recognition

A typical OCR process consists of binarization of scanned image followed by applying some pre-processing steps such as skew correction, noise removal and text-graphics separation etc. The pre-processed image is segmented into lines, words and characters consecutively. The segmented character undergoes a feature extraction step and is given to a suitable classifier for recognition. Basic architecture of OCR is shown in Figure 2.1. Due to the complexity of Indian scripts, there are many challenges in each of these steps and none of them are trivial tasks. Some of the challenges with Indian scripts are

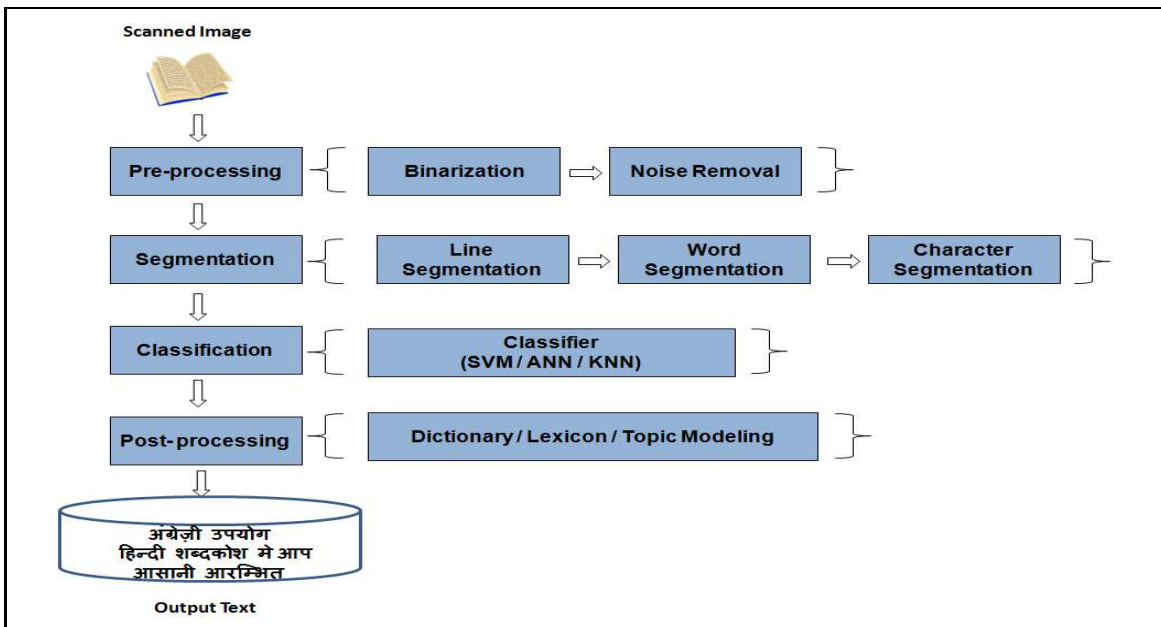


Figure 2.1 Basic Architecture of OCR.

(1) the presence of head line or “Shirorekha” in Hindi which results into difficulty in segmenting the characters properly (2) the number of unique classes (characters) are high which makes the classifier design challenging (3) vowel modifiers in conjunction with consonants make the character segmentation difficult. A detailed study of challenges in developing OCR for Indian languages is given in [10].

In OCR based retrieval, word level indexing is done based on text obtained by OCR. For Indian and non-European languages, OCR’s error rate is very high due to complexity of script. This implies that incorrect word will only get indexed in database. So, even if user provides correct query corresponding word will not be retrieved due to OCR error.

OCR system has properties of a good document search engine, i.e., (i) Fast access, (ii) Efficiency, (iii) Low computational cost, (iv) Acceptable accuracy. It is more automated and less expensive, which makes it to have fast access and thereby improves accuracy. So, OCR has been used for recognition and retrieval of document images, OCR takes document images as input and automatically outputs their digital format or text documents.

In summary, OCR procedure can be summarized as below:

- **Input:** Document images
- **Preprocessing:** It involves binarization, skew detection and normalization. Image enhancement is done to remove noise and increase contrast.
- **Segmentation:** It involves line segmentation, word segmentation and then character segmentation consecutively.

- **Feature extraction:** Features are extracted to make the document image invariant to rotation, translation and scaling etc. Many global and local features can be used. These can be based on significant properties of document images such as profile and structure etc.
- **Classification:** Extracted features are used to train a classifier. When a new document image is given as input, classifier recognizes it by the trained information. Classifiers such as Neural Networks, Hidden Markov Models (HMM), Support Vector Machines (SVM), Bayesian classifier and Template matching can be used.
- **Post-processing:** Post-processing is used to improve the recognized document image results and thereby it increases the accuracy. Character n-gram, domain knowledge, dictionary and semantic knowledge can be used for this purpose.
- **Output:** Text documents

There are many commercial OCRs available, for example, OmniPage pro from Caere, FineReader from ABBYY, Text Bridge from Xerox and Capture from Adobe, Tesseract from HP labs etc. For some scripts, OCR is efficient system for recognition [28]. OCRs promise high accuracy in case of some Latin languages [16] and some non-Latin languages [17]. But all OCRs perform well for non-degraded high quality document images.

A comprehensive survey on indexing and retrieval methods for document images is given in [26]. In this Doermann et al. presented retrieval of documents using OCRs. It mainly concerns about English language. There are no efficient OCRs available for Indian and non-Latin languages. In such case, OCR indexes incorrect words also as correct words due to which there will be high error rate, which in turn leads to poor accuracy. [64, 43] discuss some approaches to overcome errors of OCR. We discuss OCRs developed for some Indian languages (Hindi, Telugu, Bangla and Malayalam) and then for some other languages in next paragraphs.

A nice survey on recognition of Indian scripts is presented in [46]. In this, Pal et al. reported that most of the development of OCRs is concentrated on mainly two popular languages, i.e., Hindi and Bangla. For Hindi Language, K- Nearest Neighbour (KNN) based OCR and Neural Network based OCR are presented in [14] and [19] respectively. In [19], this work is extended for Bangla language. Chaudhuinary et al. [20] proposed a complete OCR system for printed Bangla language. Their method involves compound character recognition using a tree classifier and template matching. In [33], Jawahar et al. presented OCR for Hindi and Telugu documents. Their method is based on Principal Component Analysis (PCA) followed by Support Vector Machine (SVM). Jithesh et al. [35] developed an OCR for Malayalam language, in which they implemented a two level segmentation, feature extraction and classification using decision tree. A spell checker is used as post-processor, which improved the accuracy. In [5], Negi et al. presented an OCR for Telugu language, for which they used an approach based on connected components and fringe distance template matching.

Cowell et al. [23] presented Amharic character recognition using an algorithm based on fast signature. In this, they concerned more about developing better feature extraction for script identification.

In [71], Yaregal et al. proposed direction field tensor to recognize Ethiopic characters. For recognition, primitive structural features along with their spatial relationships are used. Spatial relationships are obtained using a tree structure. Worku et al. [8] used Hidden Markov Random Field (HRMF) for recognizing Amharic bank checks. Contextual information based on syntactical structure is used to improve accuracy. Edward et al. [27] used generalized Hidden Markov Models (gHMM) for Medieval Latin scripts. In this, they convert script into text using gHMM and retrieval is done based on that. They reported 75% of transcription accuracy.

In [64], Taghva et al. built a search engine based on the similarity between query word and the words which are consisted in database. Ishitani [29] came up with a method which can be tolerant of OCR errors for document images. In this, keyword matching is used for searching a string pattern from OCR results. In [18], Chan et al. proposed a method for searching Arabic documents. Segmentation and recognition are done together. They use gHMMs with a bi-gram transition and Kernel Principal Component Analysis (KPCA) for discriminating characters. Zhidong et al. [72] presented Bbn Byblos OCR system, in which they use HMM with bi-gram transition model, is used for recognition.

Recognition methods for Indian languages [17, 46] and for many non-Latin languages are not available. There are no robust and reliable recognizers are available for those languages. Hence, a present challenge in OCR systems is effectiveness [41]. There is a necessity of alternate approaches for effective access to document images. Effectiveness can be achieved by using only image properties for document image indexing and retrieval.

Lee et al. [40] provides adaptive language and image model for improving book OCR. They have used document specific model for both model. Each model adapts to shapes and vocabularies within given book for correction hypothesis. Each model exploits the redundancy in font and vocabularies to detect inconsistencies. They have shown 25% improvement over Tesseract OCR.

Kluzner et al. [37] uses a modified hierarchical optical flow with a second-order regularization term to compare each new character with the set of super-symbols (character templates) by using its distance maps. The classification process is based on a hybrid approach combining measures of geometrical differences (spatial domain) and distortion gradients (feature domain). Using this method, they have got significant improvement on Historical books.

Namboodiri et al. [9] proposed a semi-supervised SVM based framework that can incorporate the unlabeled data for improvement of recognition performance. They have extended two class SVM to large-class problems by incorporating a participation term into the optimization process.

2.2 Word Spotting

In OCR, one has to recognise every character and later word is formulated based on these characters. So, if one character is wrongly recognised in a word whole word is wrong even if rest other characters are recognised correctly. To overcome, this limitation recognition free approaches are proposed. In this, the most popular one is word spotting. In word spotting, every given word image is represented using

features. Similarity between words is determined by similarity measure. In this method, document is treated as collection of word images. So, first of all document is segmented into its corresponding lines and then into words. Each document is indexed by the visual image features of the words present in it. Word level matching has been attempted for printed [21] as well as online [30] and offline [47] handwriting documents. They are useful for locating similar occurrences of the query word.

Rath et al. [47] were able to locate a specific word in a handwritten text by matching image feature for historical documents. The word spotting approach [48, 47] has been extended to searching words from printed document images of newspapers and books. Dynamic time warping (DTW) based word-spotting algorithm for indexing and retrieval of online documents is also reported in [30]. Features for matching words are computed from the constituent strokes. The DTW, a dynamic programming technique used to align sequences, is applied for matching the features of the word images. In the word spotting approach proposed by Rath et al. [47, 49], word images are matched with each other and then clustered. Each cluster is then annotated by a person. Alternatively, Jawahar et al. [34, 12] has shown that in the case of printed books one can synthesize the query image from a textual query for making the system more usable. To simplify the process of querying, a word image is generated for each query and the cluster corresponding to this word is identified. In such methods, efficiency is achieved by significant online computation.

Ataer and Duygulu [11] tried word spotting for handwritten Ottoman documents where they use successive pruning stages to eliminate irrelevant words. Gatos et al. [38] used word spotting for old Greek typewritten manuscripts for which OCRs did not work. One advantage of word spotting over traditional OCR methods is that it takes advantage of the fact that within corpora such as books the word images are likely to be much more similar. The traditional OCRs do not take advantage of such facts. In addition, techniques that work at the symbol level of word images, like [18], are very sensitive to segmentation errors. Segmentation of Indian language document images at symbol level is challenging due to the complexity of the scripts.

Word spotting in different scripts like Devanagari, Arabic and Latin using the shape features has also been demonstrated. Global word shape features are used for finding word similarities during search. The word image features, for example gradient, structural and concavity (GSC) features which measure the image characteristics at local, intermediate and large scales and hence approximate a heterogeneous multi resolution paradigm to feature extraction. The features are extracted by dividing each image into 4×8 rectangles and contain 384 bits of gradient feature, 384 bits of structural features and 256 bits of concavity features, giving a binary feature vector of length 1024. The distance between a word to be spotted and all the other words in the documents being matched against is computed using a normalized correlation similarity measure.

Rusiñol et al. [51] have proposed segmentation free word spotting for heterogeneous document image collections. They have used patch based frame work for retrieval. Documents are first divided into patches of size 300×75 pixels with densely sampled at each 25 pixels. These patches are then represented using SIFT feature. On these SIFT features Bag of Visual Word framework is applied to define

similarity between images. Indexing is done using Latent Semantic Indexing, which uses $tf - idf$ for ranking retrieved list.

Roy et al. [50] applied word spotting using string matching of character primitives. Text string is described as a sequence of primitives which consists of a single character or a part of a character. Primitive segmentation is performed analysing text background information that is obtained by water reservoir technique. Primitives are then clustered using template matching. In retrieval, similar words are found using approximate string matching.

Abidi et al. [7, 6] have used word spotting techniques towards building searchable digital Urdu libraries and handwritten text. Their method is based on segmentation of Urdu text in to partial words and representing each partial word by a set of features. To search a specific word, query is provided in the form of an image. Comparing the features of the partial words in the query image with the ones already indexed, and ranked list of words is returned. Three types of matching technique are used Smart Sorting, DTW and Relative Distance Matching and Combining (RDMC). Smart sorting is done based on aspect ratio and convex area. DTW is performed in partial words profile features. RDMC is used to merge the spotted partial words into complete words and eliminate the irrelevant ones.

Saabni et al. [53] present a fast search of handwritten Arabic word parts within large lexicons. First it warps multiple appearances of each word-part in the lexicon for embedding into the same Euclidean space. The embedding is done based on the warping path produced by the DTW process while calculating the similarity distance. In the next step, all samples of different word-parts are resampled uniformly to the same size. The kd -tree structure is used to store all shapes representing word parts in the lexicon. Fast approximation of k -nearest neighbours generates a short list of candidates to be presented to the next step. In the last step, the Active-DTW [62] algorithm is used to examine each sample in the short list and give final accurate results.

Kesidis et al. [36] presented a method to find the cut-off for ranked list in order to provide the best trade off between recall and precision rates. Cut-off threshold is determined by approximately maximizing the expected F -score, which combines the distance of each ranked word with its cumulative moving average.

As discussed, the performance of some image matching techniques is acceptable in finding similar words. These methods use paradigms like dynamic programming and correlation etc. to achieve efficient word matching. Computational complexity is the major problem when the data set to be processed increases. The performance degrades drastically with the data set size. In [49], word images have been represented by profile features and similarity between images is determined using Euclidean distance. To take care of word image length, word images are matched using dynamic time warping [49]. DTW based technique is computationally expensive to match similarity between given two image. It takes 1 second to get similarity between two images [49], which makes practically impossible for search engines where dataset will be in Millions. One way to overcome this is to index the features computed. Indexing will reduce the similarity calculation time compared to direct matching but indexing raw features will

take large amount of disk space which can not be put into the main memory. To overcome this, better mechanism is required for representing word images.

2.3 Feature Extraction/Representation

Feature extraction is the problem of gathering information from raw data, which is most relevant for a given application. Many different features have been employed in image processing and pattern recognition for representation of document images [25]. Features are broadly classified into two categories: global features and local features. In global features, image is represented by single vector while in local features, first keypoint is selected and based on keypoints neighborhood feature is designed. In document image, both types of features are used in the literature. Some of the popular features are: profiles feature, moments, Gradient based binary feature (known as GSC features), SIFT, SURF and Shape Context etc. Here we will describe popular features used in word image retrieval.

2.3.1 Profile Feature

Profile feature provide a coarse way of representing word images for matching. Projection, transition, upper and lower profiles are features considered for the representation of word images. Projection and transition profiles capture the distribution of ink along one of the two dimensions in a word image, upper and lower word profiles capture part of the outlining shape of a word. More details can be found in Section 5.1. These features were employed for matching handwritten words [48, 49].

2.3.2 Shape Context

The shape context descriptor was proposed by Belongie et al. in [52]. This descriptor allows to measure shape similarity by recovering point correspondences between the two shapes under analysis. In a first step, a set of interest points has to be selected. Given these points, the shape context captures the distribution of points within the plane relative to each point of the shape. A histogram using log-polar coordinates counts the number of points inside each bin.

2.3.3 Scale Invariant Feature Transform

Scale Invariant Feature Transform (SIFT) [42] is one of the most popular descriptor proposed in literature. SIFT is widely used in different computer vision applications like image retrieval [60] and image classification [68] etc. Recently, in document community also many application of SIFT can be found as word image retrieval [51, 69], logo retrieval [31] and page retrieval [61] etc. SIFT, as described in [42], consists of four major stages: (1) scale-space peak selection; (2) keypoint localization; (3) orientation assignment; (4) keypoint descriptor. In the first stage, potential interest points are identified by scanning the image over location and scale. This is implemented efficiently by constructing a Gaussian pyramid

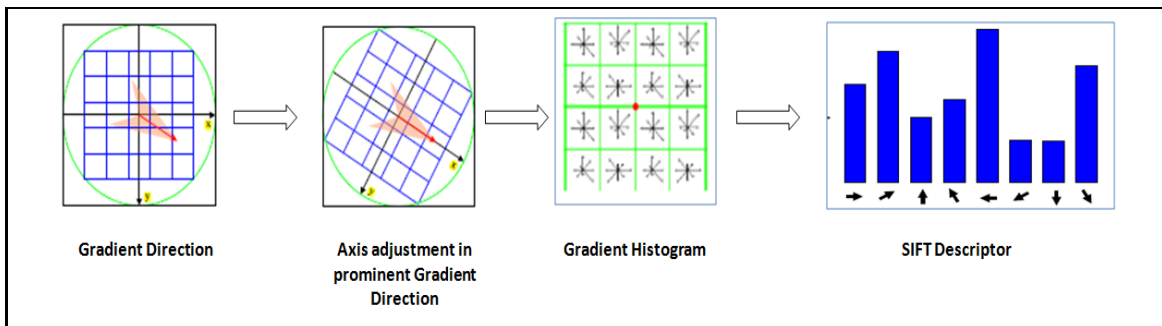


Figure 2.2 SIFT descriptor calculation based on gradient of image.

and searching for local peaks (termed keypoints) in a series of difference-of-Gaussian (DoG) images. In the second stage, candidate keypoints are localized to sub-pixel accuracy and eliminated if found to be unstable. The third identifies the dominant orientations for each keypoint based on its local image patch. The assigned orientation(s), scale and location for each keypoint enables SIFT to construct a canonical view for the keypoint that is invariant to similarity transforms. The final stage builds a local image descriptor for each keypoint, based upon the image gradients in its local neighborhood (discussed below in greater detail). The final (keypoint descriptor) stage of the SIFT algorithm builds a representation for each keypoint based on a patch of pixels in its local neighborhood (see Figure 2.2).

SIFT involves detecting salient locations in an image and extracting descriptors that are distinctive yet invariant to changes in viewpoint and illumination etc.

The SIFT feature computation can be summarized by the following steps:

1. Gradually Gaussian-blur the input-image to construct a Gaussian-pyramid.
2. Construct the Difference of Gaussian (DOG) pyramid by computing the difference of any two consecutive Gaussian-blurred images in the Gaussian pyramid.
3. Find local maximums and local minimums in the DOG space and use the locations and scales of these maximums and minimums as key-point locations in the DOG space.
4. Compute gradients around each key-point (at least a 16×16 region) at the key-point scale and assign an orientation to each key-point based on nearby gradients.
5. Compute histogram of 8-direction Gaussian weighted gradients in 16 sub-blocks (minimum size is 4×4).
6. Concatenate the 16 histograms from 16 sub-blocks to form a 128 dimensional vector as a feature descriptor.

2.3.4 Speeded Up Robust Features

Speeded up robust features (SURF) [15] is proposed as an alternative to SIFT which is of less dimension and can be computed faster than SIFT. SURF descriptor for a given patch is calculated by first equally subdividing a given patch into a 4 grid. For each subsection, the Haar wavelet response D_x and D_y are computed in the x and y directions respectively. SURF descriptor calculates the 4 attributes ($\Sigma D_x, \Sigma D_y, \Sigma |D_x|, \Sigma |D_y|$) per interest point. In document retrieval, SURF is used in logo retrieval [31] and sign board detection [56] etc.

2.4 Summary

Our objective is to retrieve similar word images from a collection of document images. Retrieval is done with the help of the following approaches: Recognition-based approaches, Recognition-free approaches or a combination of them. Recognition-based approaches requires an OCR system to convert document images into text and then search in the recognized text. Effectiveness of recognition-based approach greatly depends on the availability of robust character recognition schemes. Recognition-free approaches, on the other hand, search for retrieval of relevant documents without explicit recognition (in the image domain). For Indian languages, robust and efficient OCRs are not available. So, recognition free approach is preferred for Indian languages and in this thesis we explore the same. The document images are represented and accessed at word level. The words of the documents are represented using some features. The features are, generally, vectors containing numeric values of high-dimensions. In this thesis, we also tried to explore word image representation.

Chapter 3

Word Image Retrieval using Bag of Visual Words

With the development of computer technology, storage has become more affordable. So, all traditional libraries are making their collection electronically available on Internet or digital media. To search in these collection, better indexing structure is needed. It can be done manually or automatically. But manual indexing is time consuming and expensive. So, automation is the only feasible option. Retrieval of relevant word images from a database of word images is a challenging problem. There are three primary dimensions to this problem: (i) How to represent the word images? (ii) How to match/compare two word image representations? and (iii) How to retrieve efficiently and accurately when the size of the database grows?. All these problems are relatively easy when the representation is text, which can be obtained using an Optical Character Recognition (OCR) system. However for many languages (especially for Indian Languages) reliable and robust OCR systems are still not available [58]. Many of these languages have rich heritage, and large quantity of printed material exist in them. They are now getting digitized and archived, but handicapped with the content level access to the collection [54].

Word spotting [49] has emerged as a promising method for recognition free retrieval. Here, word images are represented using some features, and comparison is done with the help of an appropriate distance metric. Due to appearance based nature of the matching, word spotting has the advantage that it does not require prior learning. Such word matching schemes have been popularly used in document image retrieval. For example, accessing historic handwritten manuscripts [49], searching documents in a collection of printed documents [13] etc. In traditional word spotting, word images are often represented using a sequence of feature vectors and compared using Dynamic Time Warping (DTW). Word spotting with DTW works well. However it takes approximately one second to compare two word images [49]. This makes it practically infeasible in case of large database, where millions of word images are present.

With the success of document image retrieval, scalability issues have surfaced. In [65], 10M pages are indexed, and the retrieval process takes only 38ms. This is achieved with the help of a memory intensive hashing scheme. The focus of their work is in retrieving *similar pages* with the help of an invariant descriptor. Such representations are too coarse for describing word images for the content level access. Methods like approximate nearest neighbor search [55] are also used to compare word images using a vector space representation. However such methods are also memory intensive. At the

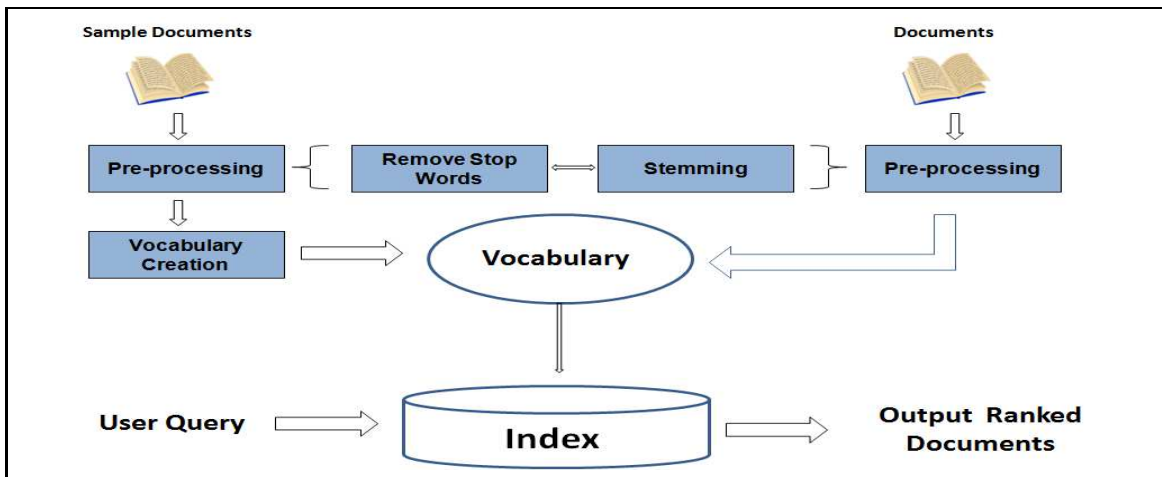


Figure 3.1 Bag of Words framework in text domain.

same time, we notice that text search engines are scalable to billions of documents comfortably. This motivates us to explore an alternative approach to word image retrieval.

We use BoVW representation for retrieval of word images. This is motivated by multiple factors (i) Bag of Words (BoW) representation has been the most popular representation for document (text) retrieval. There are scalable (and even distributed software) solutions available. (ii) BoVW method has shown to perform excellently for recognition and retrieval tasks in images and videos [60, 39]. (iii) Being a loose representation, BoVW representation can retrieve sub-words, which is difficult with the popular vector space models. However, this paper does not exploit the full power of this flexibility in retrieving partial matches. In BoVW, an image is represented by an unordered set of non-distinctive discrete visual words. In retrieval phase, an image is retrieved by computing the histogram of visual word frequencies, and returning the word image, with the closest (measured by the cosine of the angles) histogram. This can also be used to rank the returned word images. A benefit of this approach is that, matches can be effectively computed. Therefore, images can be retrieved with no delay.

3.1 Bag of Words

Document retrieval is extensively studied in text domain. One of early and popular method to search in text domain is based on Bag of Word (BoW) approach, see Figure 3.1. BoW approach provides mechanism to represent documents. BoW is generally divided into followings: Text pre-processing, Stemming and Vocabulary Selection. Using selected vocabulary document is indexed using inverted file index and retrieval is performed using term frequency inverse document frequency ($tf - idf$).

In text pre-processing step, a text document is split into a stream of words by removing all punctuation marks and by replacing tabs and other non-text characters by single white spaces. This tokenized representation is then used for further processing. The set of different words obtained by merging all text documents of a collection is called the dictionary or vocabulary of a document collection. After

getting vocabulary, vocabulary size is reduced by filtering and stemming. Initial filtering is done in term of removing stop words like articles, conjunctions, prepositions, etc. Removing stop words bear little or no content information on document representation. Furthermore, words that occur extremely often can be said to be of little information content to distinguish between documents, and also words that occur very seldom are likely to be of no particular statistical relevance and can be removed from the vocabulary. Next is to try to map verb forms to the infinite tense and nouns to the singular form. In Stemming, try to build the basic forms of words, i.e. strip the plural s from nouns, the ing from verbs, or other affixes, for example 'read', 'reading' etc is assigned same stem 'read'. A stem is a natural group of words with equal (or very similar) meaning. After the stemming process, every word is represented by its stem.

To further decrease the number of words that should be used also indexing or keyword selection algorithms can be used. In this case, only the selected keywords are used to describe the documents. A simple method for keyword selection is to extract keywords based on their entropy. Entropy gives a measure how well a word is suited to separate documents by keyword search. As vocabulary words a number of words that have a high entropy relative to their overall frequency can be chosen, i.e. of words occurring equally often those with the higher entropy can be preferred. In order to obtain a fixed number of vocabulary terms that appropriately cover the documents, a simple greedy strategy is applied: From the first document in the collection select the term with the highest relative entropy as an vocabulary. Then mark this document and all other documents containing this term. From the first of the remaining unmarked documents select again the term with the highest relative entropy as an index term. Then mark again this document and all other documents containing this term. This process repeated until all documents are marked, then unmark them all and start again. The process is terminated when the desired number of vocabulary terms have been selected. Although the vocabulary is a set, an arbitrary ordering fix for it so we can refer to word 1 through word V where V is the size of the vocabulary.

Once the vocabulary has been fixed, each document is represented as a vector with integer entries of length V . If this vector is x then its j^{th} component x_j is the number of appearances of word j in the document. The length of the document is $n = \sum_{j=1}^V x_j$. For documents, n is much smaller than V and $x_j = 0$ for most words j . This gives sparse representation of a given document.

Next step is to index these documents using created vocabulary. Indexing is mostly done using inverted file index, which is most popular and efficient way to index these vectors. In inverted file index consists entry for all words and each word is linked with all the documents where the word is present and also its frequency is stored. While is retrieval phase, for a given query corresponding word in probed in index and all documents which are having given query words is retrieved. These retrieved documents are ranked using $tf - idf$, defined as following:

$$W_{ij} = f_{ij} \times (\log N - \log d_j) \quad (3.1)$$

where W_{ij} is weight of term j in document i , f_{ij} is frequency of term j in document, N is the number of documents in the collection, and d_j is the number of documents containing the term j . Thus

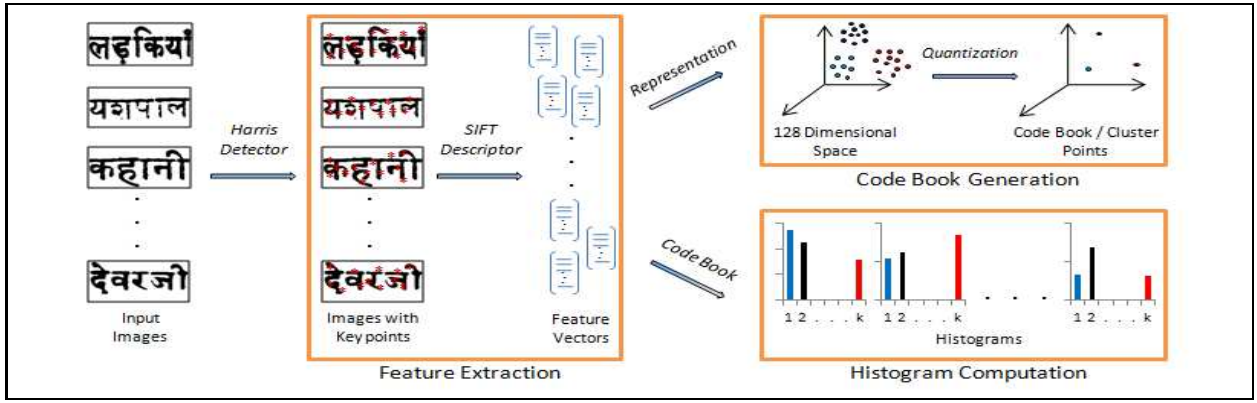


Figure 3.2 Bag of Visual Words Representation. Left: Input Images. Middle: Feature Extraction. Right Upper: Code Book Generation. Right Lower: Histogram Computation.

the weight W_{ij} quantifies the extent of relevance of the document for a given word. Relevant documents usually have high $tf - idf$ weights relative to other documents.

3.2 Bag of Visual Words

The BoVW model is inspired by the success of using BoW in text classification and retrieval. In BoW model, each document is represented by an unordered set of non-distinctive words present in the document, regardless of the grammar and word 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 vocabulary. In the case of a document image, one can think of the glyphs as the vocabulary and a word can be defined as a bag of these glyphs. 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. For example, the word ‘DAS’ and ‘SAD’ are same for this representation due to the lack of order/structure in the representation. This reduces the precision in a retrieval task. We address this issue, while exploiting the computational advantages of the BoVW representation as explained in the next section.

Robustly segmenting a word image into the corresponding glyphs is practically impossible, specially for Indian languages where a single character (or connected component) can be composed of multiple glyphs. Moreover, in case of degraded documents, even extraction of characters become very difficult. Therefore, we represent the characters with the help of “interest points” like corners and blobs. At each of these interest points, we extract a Scale Invariant Feature Transform (SIFT) [42] descriptor to describe the local information as a vector of gradients. Space of SIFT descriptors is continuous, and we discretize the space by clustering SIFT vectors (often with K means) obtained from a small collection

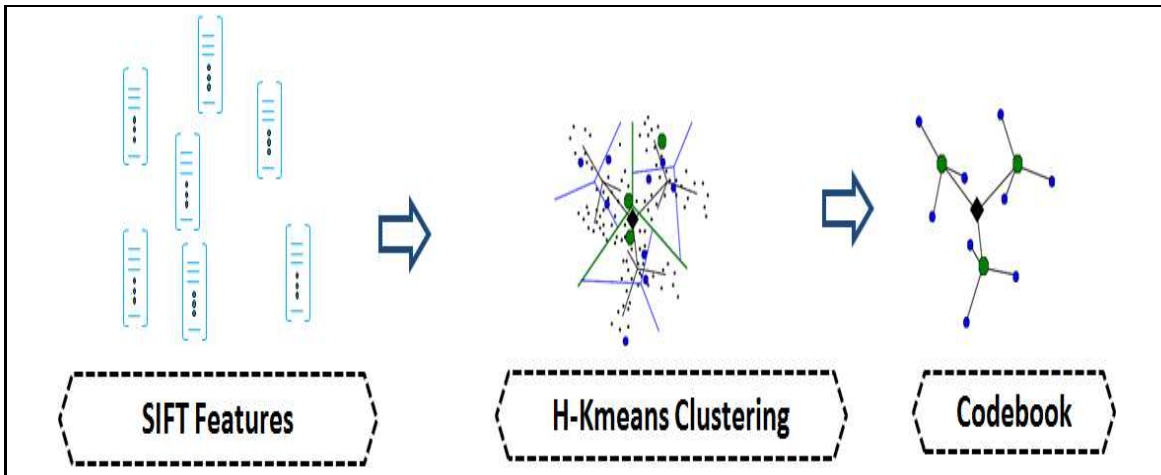


Figure 3.3 Codebook Generation Using Hierarchical K Means.

of documents. We use a vocabulary of 10000, and a clustering solution based on $K(=1000)$ means is not scalable. We use a computationally efficient Hierarchical K Means [45] for this purpose. This algorithm clusters the data into C clusters first (where C is typically ≤ 10) and then samples in each of these clusters are clustered again recursively. This process is continued until we obtain the required number (in our case 10000) clusters. This process is explained in Figure 3.3. This can give more than 1000 times speed up in practice.

This visual vocabulary is then used to quantize the extracted features by simply assigning the label of the closest cluster centroid. This is carried out by rolling down the sample from the root to the leaf of the vocabulary tree [45]. The final representation for an image is the frequency counts or histogram of the quantized SIFT features $[f_1, f_2, \dots, f_i, \dots, f_k]$ where f_i is the number of occurrences of i^{th} visual word in the image and k is the vocabulary size. To account for the difference in the number of interest points between images (due to size etc.), the BoVW histogram is normalized to have unit L1 norm.

Interest points are computed on word images using Harris corners. Harris corner detector is a popular interest point detector due to its strong invariance to rotation, scale and image noise. We also tried extracting the Maximally Stable Extremal Regions (MSER) [44] from the word images. However, that did not help much in our case. At each of the interest points, SIFT [42] descriptors were extracted. In SIFT, a neighbourhood is described by a histogram of weighted gradients within a window to yield a 128 dimensional vector.

3.2.1 Spatial Verification

One of main limitations of BoVW is that it ignores the spatial relationships between visual words, i.e. it does not consider the order of the visual words. Therefore, the retrieved word images from the Lucene have poor precision. To overcome these limitations, we need to consider the order of the visual words. An elaborate storage of spatial information using a graph-like structure is computationally pro-

Image	Its Parts
स्थिति	स्थिति
परिस्थितियों	परिस्थितियों
परिस्थिति	परिस्थिति

Figure 3.4 Spatial Verification: Image is divided into 3 parts to provide spatial order for Visual Words.

hibitive. We use the the Spatial Pyramid Matching (SPM) [39] for this purpose. In this method, image is repeatedly subdivided and histograms of local features are computed at increasingly fine resolutions. In our case, we divide the image into three parts along columns only as shown in Figure 3.4. It was observed that, if image is divided into more than three parts, there is no significant improvement in the performance. The spatial order of the characters is thus enforced by considering the sub regions. Thus the first part of a query image can match only with the first part of a database image. Similarly for other parts

3.2.2 Re-ranking

After retrieving results, we analysed that some of correct word is retrieved but ranked lower in the list. To bring those retrieved words into top of list by re-ranking it. Re-ranking is done based on SIFT matching and Longest Common Subsequence(LCS) of visual words.

3.2.2.1 SIFT Based Re-ranking

The initial retrieved results are then re-ranked explicitly to improve the overall performance of the system. Retrieved word images using BoVW and query word image are divided into three parts as explained earlier. SIFT matching is done for the corresponding parts i.e., original and three parts of both the query image and the top-k (in our implementation k = 250) retrieved images. We match the SIFT vectors by computing the distance between the SIFT vectors as well as the ratio of the best match to the second best match as in [42]. For two images (I_i & I_j), score is given by the normalization of the number of unique match points with respect to the sum of number of features in both the images.

$$Score = \frac{\#Match\ Points}{\#SIFT\ in\ I_i + \#SIFT\ in\ I_j} \quad (3.2)$$

Total score for a retrieved image is determined by weighted sum of scores of all parts.

$$Total\ Score = Score_{original} + \frac{1}{3} \sum_{i=1}^3 Score_i \quad (3.3)$$

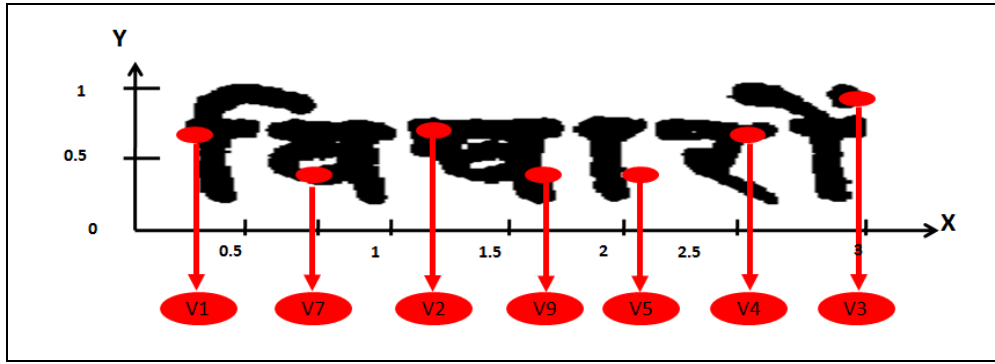


Figure 3.5 A word image and its visual words projected on X-axis.

where $Score_{original}$ is score for entire image and $Score_i$ is score for i^{th} part of the image. Retrieved images are re-ranked according to Total Score. Images with high score are kept at the top of the list.

3.2.2.2 LCS Based Re-ranking

Another method for re-ranking the retrieve list is based on order of the visual words in the word images. It is observed that the respective order of letters is not supposed to change along the X-axis, see Figure 3.5. This is true even for text written in different fonts, faces and sizes. Therefore it is sufficient to project the visual words on the X-axis and compare the resulting sequence of visual words. There have to be a large number of visual words having the same order in both sequences. This problem turns out to be a search for the longest common subsequence (LCS) which can be solved quite efficiently for short sequences using dynamic programming [24]. Length of LCS to calculate the configuration similarity between the two images.

$$LCS_Score = \#LCS(I_i, I_j) \quad (3.4)$$

where $\#LCS(I_i, I_j)$ is number of LCS between I_i and I_j

3.3 Retrieval System Overview

In this section we describe our retrieval system. Figure 3.6 shows the overview of the system. The system is divided into two parts i.e., indexing and retrieval. This is in addition to the one time computation of the vocabulary. Indexing comprises of three steps as follows: (i) Features are extracted from word images, (ii) Histograms are created by vector quantization, and (iii) Database is created by indexing word images using an inverted file index. In retrieval process, first two steps are similar to the indexing. Then histogram is finally given to the index structure and images are retrieved in a ranked manner. We have used Lucene [2], a popular, reliable and open source search engine, for indexing.

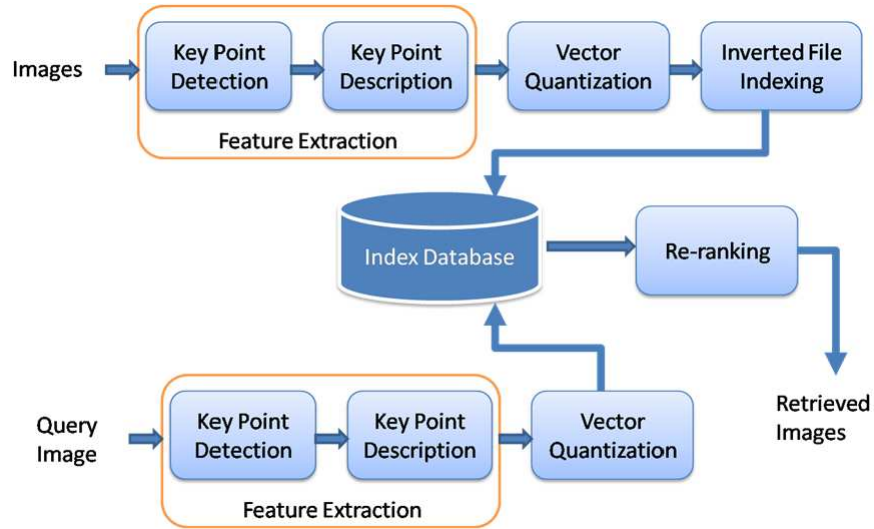


Figure 3.6 Overview of the indexing and retrieval.

Histogram computation (Figure 3.2) is carried out with the help of a precomputed vocabulary. For constructing the vocabulary, features are extracted from a subset of database images. Images are selected such that it covers all possible alphabets of the languages of interest. Clustering is done on the feature vectors extracted out of these images. Collection of centers obtained from this clustering is called the vocabulary as shown in Figure 3.2.

3.4 Experiments and Results

In this section, we present results to demonstrate the utility and scalability of the proposed system. The proposed method of indexing and retrieval is tested on a collection of word images on 4 different Indian languages to show language independence. We verify our method on more than 100K annotated word images in four different Indian languages. In order to demonstrate the scalability of the method further, we conduct experiment on a database of more than 1 Million words in Hindi. We measure the quantitative performance using mean Average Precision (mAP), and obtain an mAP of more than 0.75 across the entire collection.

3.4.1 Dataset Details

To demonstrate the utility across languages, we use a large data set of 100K words. Datasets contain four different Indian languages (Hindi, Malayalam, Telugu and Bangla) with significant change in structure. Two of them have a headline and the other two do not have. Two of them Aryan languages and the other two are Dravidian languages. Details of the data set are given in Table 3.1. All the books are annotated at the word level and ground truth was created using [32].

Languages	Dataset Type	#Books	#Pages	#Words
Hindi	Large	4	427	112677
Malayalam	Large	6	610	108767
Telugu	Large	5	742	131156
Bangla	Large	3	363	124584
Hindi	Huge	32	3992	1008138

Table 3.1 Books Details Used for the Experiments.

Language	#Images	#Query	Prec@10	Prec@10 after Re-ranking	Prec@10 after Sp. Ver.
Hindi	112677	138	0.6808	0.7820	0.7865
Hindi	1008138	138	0.5894	0.7022	0.7062
Malayalam	108767	101	0.6962	0.7991	0.8188
Telugu	131156	131	0.6483	0.7328	0.7495
Bangla	124584	125	0.7806	0.8766	0.8947

Table 3.2 Precision@10 Statistics on 4 different languages.

3.4.2 Results and Discussion

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 precision at 10 (Prec@10) and mean Average Precision (mAP). The Prec@10 shows how accurate top 10 retrieved results are. Our method is giving 0.8543 Prec@10, even in the case of huge dataset (see Table 3.2). The mAP is the mean of the area under the precision-recall curve for all the queries. A direct BoVW solution gave only a mAP of around 0.65. With our enhancements based on re-ranking and spatial verification, mAP increases to more than 0.75 as shown in Table 3.3. Some of the example queries and retrieved words are shown in Figure 3.7, where one can observe the print variations and degradations (like cuts and merges). It is also observed that, if the length of the query increases, the performance (mAP) also improves. This is shown in Figure 3.8. This is natural because, longer words have richer histogram and more discriminative power. In the case of shorter query words we were obtaining results where query is a substring of the retrieved word. We also analyzed the maximum possible mAP for the same retrieved list, that can be achieved with the help of an ideal re-ranking (i.e., all the correct images according to ground truth will be on the top of retrieved list). As it can be seen in Figure 3.8, our re-ranking method is quite comparable to the ideal re-ranking, especially for longer words.

To show the scalability, we use a huge dataset of 1M words in Hindi (see Table 3.1). The retrieval time from Lucene required for this dataset is summarized in the Table 3.4, on a system with 2 GB RAM and Intel® Core TM 2 Duo CPU with 2.93 GHz processor. Further the mAP for this dataset is

Query	Retrieved Results				
कर्मचारियों	कर्मचारियों	कर्मचारियों	कर्मचारियों	कर्मचारियों	कर्मचारियों
आख्यायिका	आख्यायिका	आख्यायिका	आख्यायिका	आख्यायिका	आख्यायिका
कथा-साहित्य	कथा-साहित्य	कथा-साहित्य	कथा-साहित्य	कथा-साहित्य	कथा-साहित्य

(a) Hindi Results

Query	Retrieved Results				
गিয়েছিল	গিয়েছিল	গিয়েছিল	গিয়েছিল	গিয়েছিল	গিয়েছিল
পশুপতিবাবুর	পশুপতিবাবুর	পশুপতিবাবুর	পশুপতিবাবুর	পশুপতিবাবুর	পশুপতিবাবুর
চারপাশে	চারপাশে	চারপাশে	চারপাশে	চারপাশে	চারপাশে

(b) Bangla Results

Query	Retrieved Results				
ఇన్నాళ్ళూ	ఇన్నాళ్ళూ.	ఇన్నాళ్ళూ	ఇన్నాళ్ళూ	ఇన్నాళ్ళూ	ఇన్నాళ్ళూ'
పెట్టుకుని	పెట్టుకుని	పెట్టుకుని,	పెట్టుకుని	పెట్టుకుని	పెట్టుకుని
జర్నలిస్టు	జర్నలిస్టు	జర్నలిస్టు	జర్నలిస్టు	జర్నలిస్టు	జర్నలిస్టు

(c) Telugu Results

Query	Retrieved Results				
ഹിറ്റ്‌ലറുടെ	ഹിറ്റ്‌ലറുടെ	ഹിറ്റ്‌ലറുടെ	ഹിറ്റ്‌ലറുടെ	ഹിറ്റ്‌ലറുടെ	ഹിറ്റ്‌ലറുടെ
കണ്ടപ്പോൾ	കണ്ടപ്പോൾ	കണ്ടപ്പോൾ	കണ്ടപ്പോൾ	കണ്ടപ്പോൾ	കണ്ടപ്പോൾ
മിണ്ടിയില്ല.	മിണ്ടിയില്ല.	മിണ്ടിയില്ല.	മിണ്ടിയില്ല.	മിണ്ടിയില്ല.	മിണ്ടിയില്ല.

(d) Malyalam Results

Figure 3.7 Sample results for All languages. First column shows the query words. Their retrieved images are shown according to the decreasing order of rank.

Language	#Images	#Query	mAP	mAP after Re-ranking	mAP after Spatial Verification
Hindi	112677	138	0.8437	0.8719	0.8770
Hindi	1008138	138	0.8059	0.8509	0.8543
Malayalam	108767	101	0.7668	0.8328	0.8581
Telugu	131156	131	0.8507	0.8668	0.8830
Bangla	124584	125	0.8498	0.9022	0.9182

Table 3.3 mAP Statistics on 4 different languages .

#Images	Retrieval Time	Index Size
25K	50ms	28 MB
100K	209ms	130 MB
0.5M	411ms	550 MB
1M	700ms	1.2 GB

Table 3.4 Average Retrieval Time and Space taken based on Dataset Size.

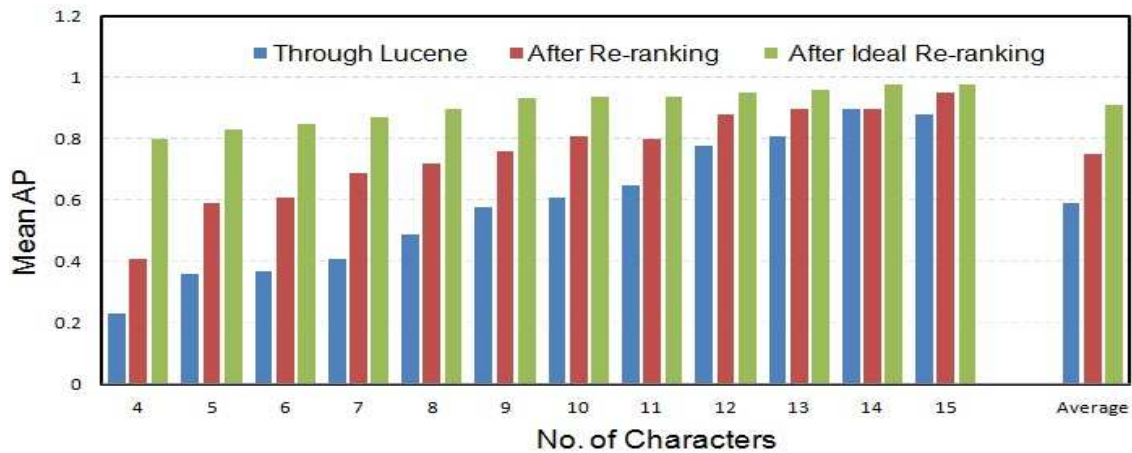


Figure 3.8 mAP Vs Query length. Also see the effect of re-ranking.

Query	Retrieved Results				
Napoleon's	Napoleon's	<i>Napoleon's</i>	Napoleon's	Napoleon's	Napoleon's
French	French	French	French	French	French
Russians	Russians	Russians	Russians	Russians	Russians

Figure 3.9 Sample retrieved list for degraded Images in the decreasing order of rank.

comparable to that of huge dataset (see second and third rows of Table 3.3). The drop in performance is of the order of 0.08, which can be attributed to the fact that the list retrieved by the Lucene is of the same size (in our size 250) in both the cases. It is natural that with huge dataset, we will have more occurrences (more than 250) in the database and a complete recall can not be obtained in the present setting. However, this can be easily improved by increasing the list from 250. Applicability of the method on a huge dataset verifies our claim that the proposed system is scalable to a huge dataset.

To benchmark our results, we also considered a dataset of English words which are “visually” similar in quality. This is done by annotating English books from a public digital library. Using our method, we obtained an mAP of 0.77 for English. This validates that the results on Indian languages are quite comparable to those of English. To know the limits of degradation which our representation can handle, we created a set of degraded English words. Commercial OCRs failed to recognize these words. We indexed these words along with the dataset. We see that our retrieval system retrieves these words from the 100K database of English words. Some of the retrieved degraded words are shown in Figure 3.9. Note that there are more than 300 occurrences for these word in the database and the AP for these word are more than 0.75. In general, we observe that our retrieval system is reasonably robust to cuts and merges in word images. SIFT descriptors are argued to be not robust to all possible document degradations. An improved descriptor can make our system compatible with a wide variety of document degradations.

3.4.3 Use of Search Engine

An inverted index is one of the popular and efficient indexing structures for BoW histograms. These index structures are implemented in many search engines. We use Lucene [2], a popular open source search engine for the present work. Each visual word (term) points to a list of word images (document) that contain it.

Internally, Lucene creates frequency file that contains the list of documents along with the term frequencies. If Lucene finds a term that matches with the search word in the term information file, it will visit the list in the frequency file to find which documents contain the term.

In retrieval phase, Lucene does a phrase (collection of terms) scoring. For a given phrase, approximate phrase *idf* is calculated with sum of terms. Then it calculates actual *tf* of phrase. Similarity between query and document is calculated by dot product of two histograms.

3.5 Summary and Comments

We have presented a document retrieval system based on BoVW. Our method is highly language independent and scalable. The efficiency of proposed method is shown experimentally on four Indian languages. We have demonstrated the scalability of the method using 1 Million word images. Our future work includes (i) Learning document-specific local descriptors (ii) Use of better solutions than Lucene (iii) Use of noisy OCR outputs along with the BoVW representation (iv) Removing the re-ranking step, which is relatively time consuming.

Chapter 4

Enhancement over Bag of Visual Words based Word Image Retrieval

Lots of old books, manuscripts are being digitized in the recent past. A large collection of Indian script books are digitized in Digital Library of India (DLI) project. Searching in these books is real challenge. Traditionally, these books are converted into text using Optical Character Recognition (OCR) and search is being done on text domain using traditional text search approach. This mechanism works well only when reliable OCR is available, which is not in the case of Indian scripts [58]. Due to lack of availability of robust OCRs, image-based indexing and retrieval method are explored. In image based retrieval system, searching is done at word level. In this direction, [49] proposed word spotting method. In traditional word spotting, word images are often represented using a sequence of feature vectors and compared using Dynamic Time Warping (DTW). Word spotting with DTW works well. However, it takes approximately one second to compare two word images [49]. This makes it practically infeasible in case of large database. Recently [59] and [69] used Bag of Visual Words (BoVW) based retrieval system. In BoVW, descriptor is extracted using key points of word images and this descriptor is quantized using pre-computed visual words. BoVW provides effective method for matching similarity between given two word images. But due to lack of order in BoVW, spatial consistency is lost. To overcome this, the Spatial Pyramid Matching (SPM) [39] is used. The SPM divides a given image into several segments in different scales, then computes the BoVW histogram within each segment and concatenates all the histograms to form a high dimension vector representation of the image. Similar to SPM in natural scene, in [59], word image is divided into three parts to enforce order. In this chapter, we have gone beyond BoVW based retrieval and improved upon it. First, we have observed that initial retrieved results are mostly correct. So, based on these top- k results, we formulated new query and retrieved the better results, see Section 4.1. In next enhancement, we provided text query support to our retrieval system. Text query support is provided in the same framework, see Section 4.2. Later it is found out that hard assignment of visual word can be improved using soft assignment. Soft assignment is provided using sparse coding [66]. Improvement on sparse coding is provided in [68] as the form of approximate Locality constrained Linear Coding (LLC) algorithm which allows a fast implementation of Local Coordinate Coding (LCC).

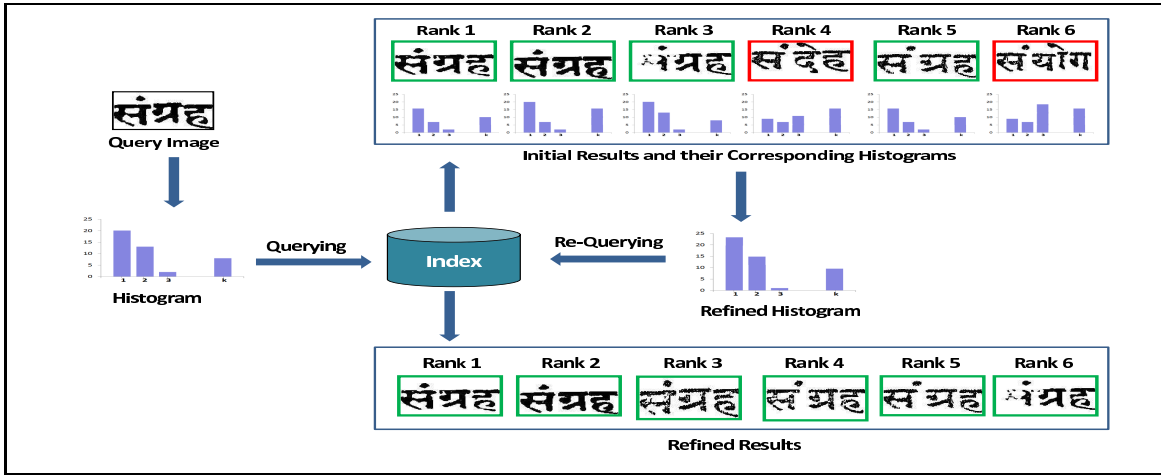


Figure 4.1 Overview of Query Expansion. Here one query image and its corresponding retrieved list are shown. Green are correct and Red are wrong. Observe that after re-fined query histogram using QE, all the top results are correct.

4.1 Query Expansion

Query Expansion (QE) is simple and efficient method for improving search performance in text search engines. In QE, highly ranked retrieved results are used to formulate the query again and better results are retrieved based on new improved query. If selected top ranked results are mostly correct, then the new formulated query will have more relevant information. Chum *et al.* [22] first introduced automatic query expansion in visual domain. The popular and efficient one is average query expansion. In average query expansion method, for a given query image, BoVW vectors of top results are averaged together with the BoVW vector of query, and this resulting new query expanded BoVW vector is used to re-query the index, see Figure 4.1. We have also used similar approach for query expansion. We take top- p results from the retrieved list and formulate the new query based on their visual words. While assigning weight to each of these visual words, we have taken rank of the word also into the consideration. Weight factor (determined empirically) is defined as $1/2^{rank}$. We also observed that few of the visual words are present in all of the considered top results. This implies that these visual words are specific to given query word. Therefore, we have increased weight of these visual words by two fold.

4.2 Text Query Support

One of the eventual goals of document image retrieval community researcher is to support text query in document image retrieval system. Here, we propose a novel and simple framework for text query support. In BoVW approach, all images are represented as histograms. Indexing and retrieval are done based on these histograms. We observed that the histograms of all variants of a given character are more

Algorithm 1 Algorithm for Query Expansion.

Input : Query image Q and p number of top images to be considered.

1. Find the histogram h_Q corresponding to query image Q
2. Query the index using h_Q and retrieved list o
3. Select top- p images from o and formulate new query histogram by
$$H_Q = \sum_{i=1}^p 1/2^i \times h_{o_i}$$
where h_{o_i} is histogram corresponding to i^{th} ranked image in o
4. Re-query the index using H_Q and output the refined list O

Output : Better retrieved list O .



Figure 4.2 Overview of Text Query Support.

or less similar. Only difference arises for some of cluster centres where their frequencies may vary, but cluster centres will be the same.

Based on the above observations, we have selected a small subset data with ground truth such that all possible characters are present at least twice in it and learnt the character specific visual words, see Figure 4.2. First of all, these selected words are segmented at character level using connected component approach and visual words of each character are determined using corresponding ground truth. For a given character with different variations, we have two or more sets of visual words. To take care of variations, we consider average of visual words of all variants of that character. Thus, given a textual query, we can synthesise the BoVW histogram of that word. Since our histogram is obtained by composing character histograms, “context” is missing in many SIFT vectors. Due to this, recall for this retrieved list is not high but initial precision is very high. To improve recall, we have also indexed word images and used query expansion as explained in Section 4.1 to formulate query histogram based on initial results of text query index.

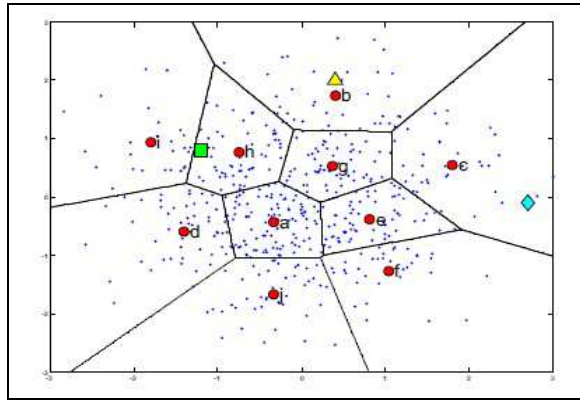


Figure 4.3 An example showing the problems of visual word ambiguity in the codebook model. The small dots represent image features, the labeled red circles are visual words determined by unsupervised clustering. The triangle represents a data sample that is well suited to the codebook approach. The difficulty with visual word uncertainty is shown by the square, and the problem of visual word plausibility is illustrated by the diamond [66].

4.3 Coding Techniques

In BoVW approach, image is treated as a collection of local features where each feature is represented by a visual word from the codebook. One limitation of the BoVW approach is the hard assignment of visual words in the vocabulary to image feature vectors [66]. The hard assignment gives rise to two issues: visual word uncertainty and visual word plausibility. Visual word uncertainty refers to the problem of selecting the correct visual word out of two or more relevant candidates. The codebook approach merely selects the best representing visual word, ignoring the relevance of other candidates. Visual word plausibility denotes the problem of selecting a visual word without a suitable candidate in the vocabulary. The codebook approach assigns the best fitting visual word, regardless the fact that this visual word is not a proper representative. Figure 4.3 illustrates both these problems. Accordingly, the hard assignment of visual words to image features overlooks visual word uncertainty, and may label image features by non-representative visual words. Hard assignment problem is solved using soft assignment. In soft assignment, instead of assigning one visual word to each feature, weighted visual word is assigned. By soft assignment, uncertain features will have equal weight to every candidate visual word.

4.3.1 Vector Quantization

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

$$\operatorname{argmin}_c \sum_{i=1}^N \|x_i - Bc_i\|^2 \quad (4.1)$$

$$s.t. \|c_0\|_{l^0} = 1, \|c_i\|_{l^1} = 1, c_i \geq 0, \forall i$$

where $X = [x_1, \dots, x_n]$ is the descriptor of the image. B is the codebook. $C = [c_1, \dots, c_n]$ is the set of codes for the image X . 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 .

4.3.2 Sparse 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. Apart from principal component analysis, 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 data.

$$\operatorname{argmin}_c \sum_{i=1}^N \|x_i - Bc_i\|^2 + \lambda \|c_i\|_{l^1} \quad (4.2)$$

However, VQ method generates quantization loss, and is poor in scalability. In order to improve scalability and reduce quantization loss, ScSPM [70] method is proposed to introduce sparse coding to SPM procedure, obtaining nonlinear feature representation which works better with linear classifiers. Here the coding problem becomes a standard sparse coding problem. A major limitation of SC-based approaches for classification is that similar data instances do not guarantee to produce similar coding results.

4.3.3 Locality Constrained Linear Coding

A Locality constrained Linear Coding (LLC) scheme learns a data representation using nearest code-word and achieves improved classification performances over the standard SC did.

Locality constrained Linear Coding (LLC) [68] 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:

$$\operatorname{argmin}_c \sum_{i=1}^N \|x_i - Bc_i\|^2 + \lambda \|d_i \odot c_i\|_{l^1} \quad (4.3)$$

$$s.t. \mathbf{1}^T c_i = 1, \forall i$$

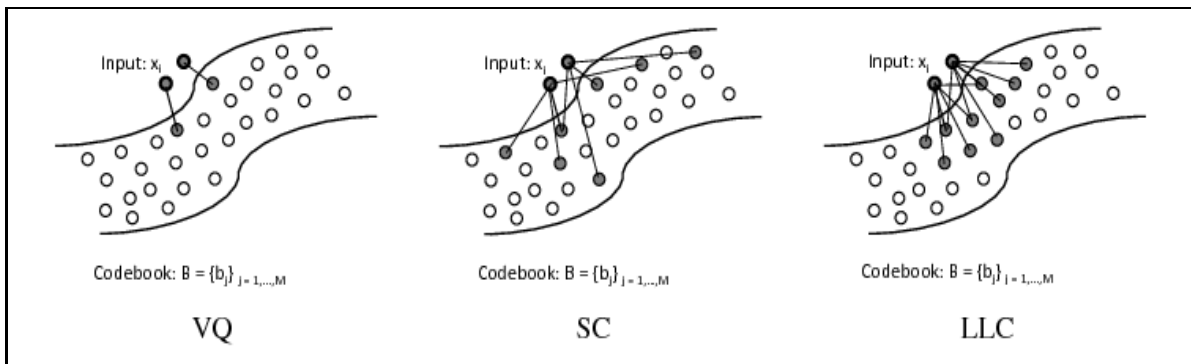


Figure 4.4 Comparison among VQ, SC and LLC [68]

where \odot denotes element-wise multiplication and d_i is the locality adaptor 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 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 whatever the size of the extracted feature descriptors is. The difference among VQ, SC and LLC coding is represented in Figure 4.4. In VQ coding, each input is coded using only one most similar element from the codebook. This leads to large quantization error. While 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.

In VQ, each descriptor is represented by a single basis in the codebook. Due to quantization errors, similar descriptors may have different representation and it does not contain any information about spatial ordering in codebook. While in LLC, each descriptor is represented by multiple bases, and LLC code captures the correlations between similar descriptors by sharing bases. Similar to LLC, SC also achieves less reconstruction error by using multiple bases but in SC regularization is not smooth. In SC similar patches may have different bases due to sparsity constrained, while in LLC due to locality constrained similar patches will have similar codes.

LLC code captures the correlations between similar descriptors by sharing bases, this provides better reconstruction. Due to the explicit locality adaptor in LLC, similar patches will have similar codes.

4.4 Experiments and Results

In this section, we provide experimental details to prove the proposed method. Different dataset is used and performance is measured in term of mAP and Prec@10.

Algorithm 2 Algorithm for Locality constrained Linear Coding.

Input: Feature x_i and Codebook $B = b_j$ for $j = (1...V)$, where V is vocabulary size and K number of neighbours to be considered.

1. Find K-Nearest Neighbors of x_i , denoted as B_{x_i}
2. Reconstruct x_i using B_{x_i} $c^* = \underset{c}{\operatorname{argmin}} \| x_i - c_i^T B_{x_i} \|^2$
 $s.t. \sum_j^K c_j = 1$
3. c_i is an $V \times 1$ vector with K non-zero elements whose values are the corresponding c^* of Step 2.

Output: c_i , LLC encoding for x_i .

Language	#Pages	#Words
Hindi(HP1)	420	112411
Telugu (TP1)	632	161274
Telugu(Telugu-1716)	120	4121
Telugu(Telugu-1718)	100	21345

Table 4.1 Dataset: Used to show Enhancement over BoVW.

4.4.1 Dataset Details

We have used books of two languages for experimentation purpose. Two languages are selected such that one is having ‘Sirorekha’ (Hindi) and another not (Telugu). Dataset for these are same as used in Section 3.4.1 and details are again provided in Table 4.1. We have also used two other Telugu books, namely Telugu-1716 and Telugu-1718 for experimentation. These books are used for establish direct comparison of our proposed method with [69]. Both books contain more than 100 pages. Details are provided in Table 4.1.

4.4.2 Results and Discussion

In this section, we will first present results based on utilities of query expansion and text query support on HP1 and TP1. Table 4.2 presents results based on these two mechanisms. We can observe that query expansion improves BoVW’s performance by 4% for Hindi and 2% for Telugu in mAP. These improvement occurs due to the fact that initial results are mostly correct in BoVW model. Text query

Dataset	#Query	BoVW		Query Expansion		Text Query	
		mAP	mPrec@10	mAP	mPrec@10	mAP	mPrec@10
HP1	100	62.54	81.3	66.09	83.86	56.32	73.89
TP1	100	71.13	78	73.08	79.89	69.06	78.83

Table 4.2 Performance Statistics with Query Expansion and Text Query Support.








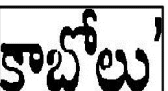
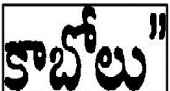








Query Image	Retrieved Images				
 BoVW					
 Query Expansion					
क़ाबूँलु Text Query					

Figure 4.5 Sample retrieved word images based on different methods, which have same rank. Note that Text Query give all clean results on the top. It happens because, for Text Query cleaned characters are used for visual word learning.

based results for both the languages are also comparable to original BoVW based search. Here it is can be noted that text query based results for Hindi and slightly in lower side. This is due to the fact that Hindi has ‘Sirorekha’ which makes image level character segmentation difficult and every character will have line over it so similar visual word corresponds to it. Sample result based on query expansion and text query support are presented on Figure 4.5.

Next, we will present [59] and [69] based on our implementation. Instead of presenting final results as a whole, we have sub-divided results into parts and try to analyse the pattern in them, as shown in Table 4.3. We analysed that both methods result into same initial results, i.e., only based on BoVW. Main difference lies in re-raking methodology. LCS performs better than SIFT based re-ranking because SIFT based re-raking helped only in eliminating quantization where as by considering longest common sequence of visual words, LCS has taken care of order of the visual words which is also important in the case of word images because order of characters is fixed in a given word image. This also eliminates the need of spatial verification at indexing level. Taking linear combination of these two scores also does not eliminate order provided by any of the methods. Here, we can observe that both methods produce similar results with the difference of only 3-4%.

In the next set of experimentation, we have shown baseline results based on LLC. In LLC, instead of taking only one nearest neighbour, more than one cluster centres are considered. Notion behind this is, it not only gives better representation but also minimizes quantization error as can be observed in Table 4.4. We can observe that LLC with inverted file index performs better than BoVW. On the obtained results, we have performed LCS based re-raking and we have reached mAP of 0.95.

In the next phase of experimentation, 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 engine like ‘Google’ and ‘Bing’ etc. Based on characters provided in the search engine, query’s visual words are formulated

Book	Method	mAP
Telugu-1716	BoVW	0.8173
Telugu-1716	BoVW + SIFT Re-ranking	0.8531
Telugu-1716	BoVW LCS Re-ranking	0.8867
Telugu-1716	Linear Combination	0.9063
Telugu-1718	BoVW	0.7834
Telugu-1718	BoVW + SIFT Re-ranking	0.8861
Telugu-1718	BoVW + LCS Re-ranking	0.8798
Telugu-1718	Linear Combination	0.918

Table 4.3 Baseline Results.

Book	Method	mAP
Telugu-1716	LLC	0.91
Telugu-1718	LLC	0.92
Telugu-1716	LLC +LCS	0.95
Telugu-1718	LLC +LCS	0.96

Table 4.4 Baseline Results using LLC on Telugu-1716 and Telugu-1718.

based on already learned visual words of characters. Using formulated query visual words, text query index (first indexed) 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 recall of the system, second index is queried based on query expanded result of text query part. Second index is created using LLC methodology and results overcome the low recall and are comparable to those of query by image, see Table 4.5.

4.5 Summary and Comments

We have presented enhancement over BoVW based system in term of query expansion (QE), text query support and Locality constrained Linear Coding (LLC). QE improves performance and we are able to text query on printed documents having performance comparable to BoVW. By using LLC, we are able to show superior performance over existing state-of-the-art systems which has been shown on Telugu data. As future work, we need to provide better mechanism to learn character level visual

Book	Method	mAP
Telugu-1716	Text Query	0.8413
Telugu-1718	Text Query	0.8209
Telugu-1716	Text Query with QE	0.9217
Telugu-1718	Text Query with QE	0.8978

Table 4.5 Text Query with LLC Based Results on Telugu-1716 and Telugu-1718.

words for text query support. Computation of SIFT descriptor and histograms of word images is time consuming process, which needs better.

Chapter 5

Beyond SIFT: Quantized Feature for Robust Document Retrieval

Traditional document image indexing applications have used primarily two approaches, one based on text search which requires efficient and robust optical character recognizers (OCR) and another based on word spotting. OCR output may have high rates of errors due to many factors such as high document degradation, different font style etc. As a result, the search over the OCR output is less accurate. Apart from these disadvantages, reliable and robust OCR systems are still not available for many languages (especially for Indian Languages) [57].

The word spotting [49] approach for building of document images ensures that identical words have similar visual appearances. Similarity between word images is ensured by matching the word images. In recent works, primarily two approaches have been used for word image matching: pixel level matching and feature based matching. In pixel level matching, raw pixel values of character/word are used to compute similarity between images. In feature based matching, global or local feature for word images is calculated and similarity is measured based on feature distance.

In global feature based matching, profile feature is extensively used. Profile feature matching is mainly done by dynamic time warping (DTW) [49] mechanism. DTW based method works for small dataset but the time taken during matching is more due to which it is not scalable to large dataset. Apart from DTW, shape context [52] based matching is also used. In the Shape Context framework, a set of points is sampled on the boundary of the shape. A Shape Context is associated with each point and represents the arrangement of the each point with respect to other points. The set of the histograms for the point set is further used for establishing correspondence between points on two shapes. The similarity between two objects is measured by a matching cost. Here, matching is done by graph matching, which is computationally expensive.

Alternatively, local feature based retrieval is scalable and efficient. Scale-invariant feature transform (SIFT) [42] matching is currently most preferred algorithm for local feature based matching. In the SIFT framework, a set of points is selected based on key-point detector. SIFT descriptor associated with each key-point is calculated. Based on pre-computed visual words, word image is represented as histogram. The similarity between two words is measured by dot product between word image histograms [60], [69], [59].

SIFT, as described in [42], consists of four steps: (i) Scale-space peak selection, (ii) Key-point localization, (iii) Orientation assignment and (iv) Key-point descriptor. In the first step, potential interest points are identified by scanning the image over location and scale. This is implemented efficiently by constructing a Gaussian pyramid and searching for local peaks (key-points) in a series of difference-of-Gaussian (DoG) images. In the second step, candidate key-points are localized to sub-pixel accuracy and eliminated if found to be unstable. The third identifies the dominant orientations for each key-point based on its local image patch. The assigned orientation(s), scale and location for each key-point enables SIFT to construct a canonical view for the key-point that is invariant to similarity transforms. In the final (key-point descriptor) step of the SIFT algorithm, a representation for each key-point is built based on a patch of pixels in its local neighborhood.

One of the main disadvantages of SIFT based retrieval is that, its time consumption for key-point detection and key-point descriptor. Therefore, time taken for pre-computation step is very high for huge datasets. Originally, SIFT is designed for natural scene images, so it does not take advantage of the fact that document images are binary. SIFT produces good results when document image is clean, but performance deteriorates when severe degradation is present [55].

We designed a document specific descriptor for word image retrieval. In this approach, we divide word images into patches and calculate different profile features. For better representation of word image, Gaussian blur is applied. By applying blur, we are able to handle cuts present in images. This also reduces the effect of noise present in the document. Profile calculation on a given patch is very fast process, i.e, it reduces the time taken for feature calculation.

Our second contribution to this work is in terms of fast pre-processing. We observed that for a given patch, its feature vector will be same. So, if we pre-compute feature corresponding to a given patch and store it as lookup table, we can directly get feature whenever given patch is spotted. It can also be observed that, for given feature vector and codebook, the visual word in which feature will fall, are also fixed. This implies that, for a given patch, its corresponding visual word is always same. We exploit this fact to calculate corresponding visual word for every patch and store in the lookup table. Using this lookup table we are able to achieve 3 times speedup in visual word assignment when compared to SIFT.

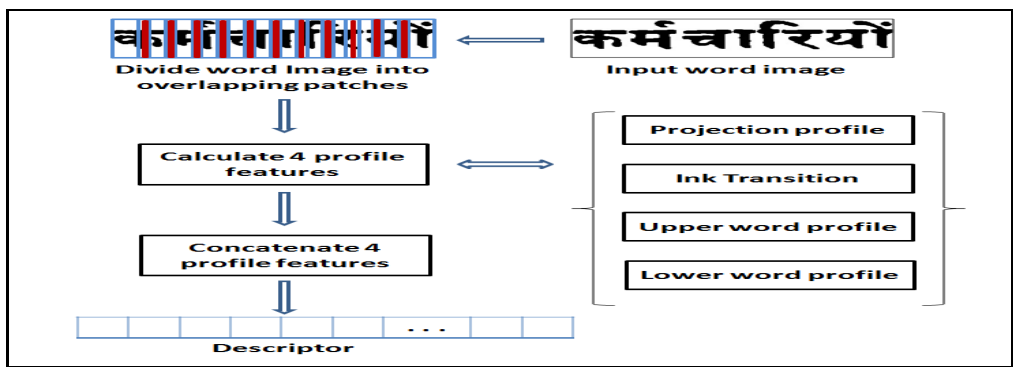


Figure 5.1 Block Diagram for Proposed Feature Calculation

The chapter is organized as follows. Section 5.1 provides details of proposed feature extraction mechanism and Section 5.2 describes how visual word details can be found out using neighborhood. Section 5.3 provides details of experimental setup and results. Section 5.4 discusses the future direction of research.

5.1 Proposed Feature

In this section, we will describe how image patches can be used to describe the given binary word image. We have used global features to describe the patches and formulated our new features. We have taken advantage of the fact that document images are binary images. Before calculating the features, we have removed inter-word variations like skew, slant angle etc. All the calculated features are normalized to range [0...255]. First of all, we describe profile features in the following section and then describe our proposed feature. Profile features are divided into three main categories, namely, Projection Profile, Word Profile and Background/ink Transitions.

5.1.1 Projection Profile

Projection profile (F_1) measures the distribution of black pixels in the word image. It is calculated by summing the black pixel values in the image in any of the three directions: horizontal, vertical and/or diagonal. For a word image, $w(i, j)$, vertical projection profile is defined as:

$$F_1 = \frac{1}{N} \sum_{i=1}^{i=N} w(i, j) \quad (5.1)$$

where $w(i, j) = 1$ and N is height of word image.

5.1.2 Background/Ink transition

This feature (F_2) records the number of transitions from an ink pixel to the background one. Such feature is important for representing internal structural shape of word images. It is defined as:

$$F_2 = \frac{1}{t} \sum_{i=1}^{i=N} tran(i, j) \quad (5.2)$$

where t is the maximum possible transitions in writing a character in the script and $tran(i, j)$ is defined as:

$$tran(i, j) = \begin{cases} 1, & \text{if } w(i-1, j) = 0 \ \& \ w(i, j) = 1. \\ 0, & \text{otherwise.} \end{cases} \quad (5.3)$$

5.1.3 Word Profile

Word profiles capture a part of the outlining shape of a word. Two types of word profiles, upper (F_3) and lower (F_4) word profiles are used. Upper word profile is defined as the distance of first black pixel from the top of the image and lower word profile is defined as the distance of first black pixel from the bottom of the image. The upper and lower word profiles can be calculated as follows:

$$\begin{aligned} F_3 &= \frac{(i - upbound)}{N} \\ F_4 &= \frac{(i - lowbound)}{N} \end{aligned} \tag{5.4}$$

where, i is the first ink pixel in a row when seen from top and bottom for F_3 and F_4 respectively, $upbound$ and $lowbound$ represent the upper boundary and the lower boundary of a word image, respectively.

5.1.4 Proposed Feature Calculation

To calculate patch based feature, first of all, we divided the word image vertically into patches having fixed size of p pixels, called the patch size. For a given patch, we have calculated profile features and concatenated them horizontally to form a patch feature. This patch feature is used for representation of patch. For every column of a patch, we will get 4 features (projection, upper word profile, lower word profile and ink transition). So, for a patch of size p , patch feature will be represented as $1 \times 4p$ vector. Figure 5.1 describes method to calculate Proposed Feature.

We observed that different word images are of different size. This leads to situation where even some patch is present in both images, but they do not match with each other. To overcome this, we have used overlapping patches in two different mechanisms. First, we took o pixels, call overlap size, such that it is common in consecutive patches. Second, we calculated profile features on different patch sizes. These two mechanisms cover variations due to different sizes of word images. In practice, we have used patch sizes $p = 3, 5, 7$ with overlap size $o = 1, 2, 3$ respectively.

The advantage of calculating feature in this way is that, it is free from keypoint detector's performance. Different keypoint detectors looking for points through image can be represented as edge detectors look for edges, SIFT detector looking for stable points etc. By selecting consecutive points, we are removing dependency on detector performance.

A word is made up by putting different characters in a specific order. By taking varying patch from one end to another, we are actually capturing features of characters present in the word. In feature matching level, we are finding which characters are present in query image.

Proposed feature is robust to different degradations present in the document images. In general, we find salt & pepper noise, gaussian noise, cuts, merges, overwritten and underline etc. in documents. By using standard noise removal techniques, we clear our documents first. The feature is designed such that only one or two of profile features will be affected in case of cuts and merges. In this way, while

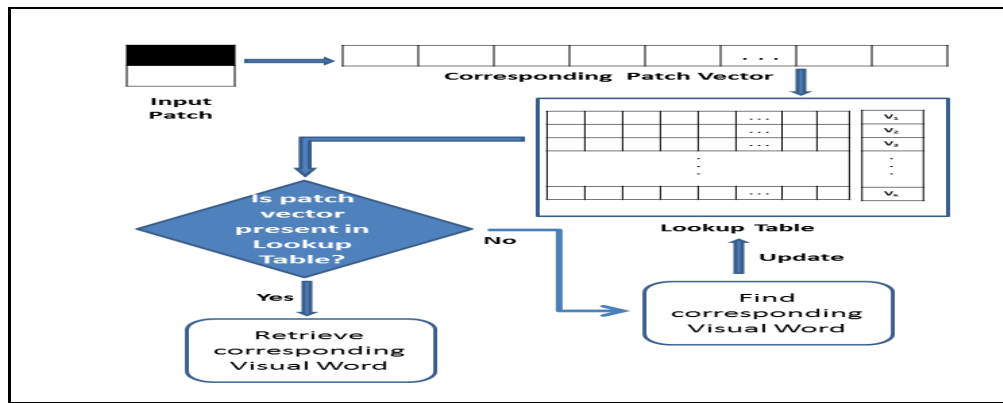


Figure 5.2 Fast Pre-processing Process.

matching there is high probability that it will match with correct corresponding feature because rest three or two profile feature values are similar.

5.2 Fast Pre-processing

An image retrieval system can be divided into two parts, i.e. indexing and retrieval. This is in addition to the one time computation of the vocabulary. Indexing comprises of three steps as follows: (i) Features are extracted from word images (ii) Histograms are created by vector quantization and (iii) Database is created by indexing word images using an inverted file index. Feature extraction can also be divided into two parts (a) Keypoint detection and (b) Keypoint description.

In our proposed approach, we reduce time consumption during feature extraction and histogram calculation. We are taking advantage of two things: document image is binary and the most of the local descriptor calculation is based on keypoint's neighbourhood. Suppose that a given descriptor considers $M \times N$ neighbourhood for descriptor calculations. For a binary image, this neighbourhood will have $2^{M \times N}$ possibilities only. So, if we pre-compute these possibilities and store descriptors in a lookup table, our descriptor calculation takes only $O(1)$ time. Apart from this, for a given descriptor, cluster center is also fixed in which it will fall. Due to these properties, if we create a lookup table consisting of $2^{M \times N}$ entries which store the cluster number corresponding to the decimal value of the $M \times N$ neighbourhood pixel values. In this way, for a given binary document image we can calculate histogram of image with no delay.

In general, total number of possible patch values for a given patch size will be very high. So pre-computing visual words for all of them will be time consuming and it will take more space for storage. In practice, we have observed that total number of possible patches are not much. So, we have used dynamic lookup table. Dynamic lookup table is first initialized at clustering step, then whenever a new patch occurs, corresponding visual word is calculated and added in the lookup table.

Language	Quality	#Pages	#Words
English (English - 1601)	Good	363	113008
Telugu (Telugu - 0001)	Good	142	32126
Telugu (Telugu-1718)	Noisy	100	21345

Table 5.1 Dataset Used in the Experimentation.

5.3 Experiments and Results

5.3.1 Experimental Setup

In this section, we will show experimentally the usability of our proposed feature. We have used printed books of two languages, namely English and Telugu, evaluation purpose. English book is “Adventures of Sherlock Holmes” written by Arthur Conan Doyle. Document images are downloaded from the Internet Archives website¹. Other two books are printed in Telugu script. These two books are selected based on the quality of the books. We have used two types of books, such that its image quality is of good, and noisy. Table 5.1 provides details of dataset. All the books are annotated at the word level and ground truth was created using [32].

To evaluate the quantitative performance, multiple query images were generated. These 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.

5.3.2 Results and Discussion

To evaluate the performance of the proposed feature, we have used BoVW framework. In BoVW framework, each word image is represented by an unordered set of non-distinctive visual words present in the image. Image is formally represented with the help of frequency of occurrences (histogram) of the visual words in the vocabulary. Vocabulary is pre-computed and its size selection is one of important factors in deciding performance. Experimentally, we have found out that smaller vocabulary size(1K) performs better. This can also be attributed to the fact that number of characters are low and which essentially limits the number of possible features.

In our experimentation, we have taken patch size p equals to 5 pixels. To capture better context of word images, we have also used overlapping patches for representing word image. Size of overlapping patch was taken as 2 pixels.

In Table 5.2, we have shown performance of proposed feature based on different methods. We have used 3 types of enhancements, namely spatial verification [59], query expansion [22] and re-ranking [69] [59]. For re-ranking, we have used Longest Common Sub-sequence(LCS) [69] which is faster to compute. We can see from Table 5.2 that spatial verification provides consistent improvement.

¹The Internet Archives: Digital Library, www.archive.org, 2012

Method	mAP
Proposed Feature	0.6183
Proposed Feature with Overlapping	0.7214
Query Expansion	0.78
Spatial Verification	0.83
LCS Re-ranking	0.8481

Table 5.2 Baseline Results on Proposed Feature

Book	SIFT	Proposed
English - 1601	0.93 [69]	0.90
Telugu - 0001	0.92	0.9417
Telugu-1718	(0.918)0.94 [69]	0.954

Table 5.3 Performance Statistics on Different Books.

LCS based re-ranking improves performance which due to fact that the way feature is designed. By design itself, proposed feature is calculated by taking horizontal slice/patch of word image which in turn provides better order of characters in a given word image. Table 5.3 compares performance over all printed books used in the experimentation. Note that, final ranking of query is done by using linear combination of original retrieved score and re-ranked score.

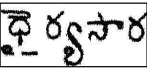
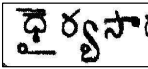
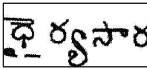
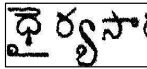

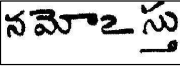
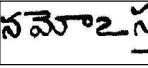
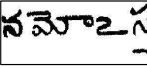

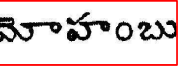
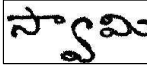

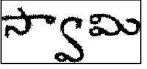
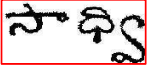

Query	Retrieved Results				
					
					
					

Figure 5.3 Sample Retrieved Word Images in Order of Retrieved Rank. Note that, partially correct retrieved words are shown in red color.

Next, we have compared time taken by SIFT and Proposed Feature. Feature calculation consists of two steps: keypoint detection and keypoint description. In SIFT calculation, we have used Harris detector and descriptor from [67]. In Proposed Feature, there is no detector process and only time consuming process is profile feature calculation. As can be seen from Table 5.4, Proposed Feature calculation is 2 times faster than SIFT². With proposed fast pre-processing mechanism, we are able to achieve 3 times speed up in quantization step. Note that, when fast pre-processing mechanism is used, we do not need to calculate feature, we will directly get visual word assigned corresponding to given

²It can be noted that, SIFT and our feature are implemented in C language and MATLAB respectively. C based implementation is faster than that of MATLAB.

Feature	Feature Computation Time	Quantization Time
SIFT	0.0374	1.1189
Proposed Feature	0.0191	0.3974

Table 5.4 Comparison between SIFT and Proposed Feature time taken during feature calculation and quantization.

patch. It mean that system will not consume time in feature calculation. Some of sample retrieved list with corresponding query is shown in Figure 5.3.

5.4 Summary and Comments

In this work, we have presented patch based feature for word image retrieval. Profile feature is calculated on overlapping patches and it is treated as local descriptor. Initial experimentation shows that performance of proposed feature is comparable to SIFT based retrieval and faster to compute. More context is needed in this feature calculation mechanism.

Chapter 6

Conclusions and Future Work

6.1 Summary and Conclusion

As diverse and large quantity of documents are getting archived by many applications including the digital libraries, access to the content of such collections becomes a challenging task. We attempt to design schemes for document image retrieval so as to facilitate access to digitized document collections. In India, there are number of languages with their own indigenous scripts. Accordingly, there is a bulk of printed materials available in libraries, information centers, museums and other institutions. But, the document analysis and understanding area has been given less emphasis for these languages in the recent past.

In this thesis, we have presented a mechanism for retrieval in language independent large document image collections. Bag of Visual Words (BoVW) model is used. Use of text search methods requires efficient and robust optical character recognizers (OCRs), which are presently unavailable for Indian languages. The approaches used for word spotting with dynamic time warping tend to be slow for large collection of books. Direct matching of images is inefficient due to the complexity of matching and thus impractical for large databases.

The performance of the method is presented with the help of experiments on real data sets from 4 different Indian language documents namely Hindi, Telugu, Bangla and Malayalam. The efficiency is demonstrated by conducting experiments on a collection of 1 Lakh of word images from all languages. Scalability of the method is tested on 1 Million word images of Hindi. The limit of degradation of the proposed method is also tested on words from English language. It can be concluded from the experiments that, the method is efficient for retrieval from large document image collections because it takes time in only sub-seconds.

Further, we have proposed enhancement over BoVW in terms of query expansion(QE), text query support and Locality constrained Linear Coding (LLC). QE improves performance and we are able to text query on printed documents having performance comparable to BoVW. By using LLC, we are able to show superior performance over state-of-the-art systems for English and Telugu datasets.

Later, we tried to improve upon descriptor for document images. Scale-invariant feature transform (SIFT) is widely used descriptor for natural scenes as well as document images. We tried to use patch along with profile features as descriptor and able to achieve comparable performance as SIFT with fast descriptor calculation.

6.2 Future Work

From here onward, a promising research area can be evolved in the direction of building recognition-free search systems for document images. Apart from this, following also can be explored further.

- The effect of combination of different fonts in a single collection can be one possible direction for exploring the feasibility of the proposed technique and improving it.
- Similar method can be explored in hand written, camera based documents etc.
- Text query support method need to be generalized. Lots of annotated data is available, so characters' vocabulary need to be learned automatically based on this data.
- Lots of research is done in terms of building OCRs for Indian languages. There is necessity of combining recognition free and recognition based methods.
- We have shown initial results of patch based methods. There is more scope to improve it.

Related Publications

- **Ravi Shekhar** and C. V. Jawahar, “*Document Specific Sparse Coding for Word Retrieval*”, In 12th IAPR International Conference on Document Analysis and Recognition (ICDAR), August 25-28, 2013, Washington, DC, USA.
- Praveen Krishnan*, **Ravi Shekhar*** and C. V. Jawahar, “*Content Level Access to Digital Library of India Pages*”, In Proceedings of 8th Indian Conference on Vision, Graphics and Image Processing (ICVGIP), December 16-19, 2012, Mumbai, India (Oral) (*: Equal Contribution).
- **Ravi Shekhar** and C. V. Jawahar, “*Word Image Retrieval Using Bag of Visual Words*”, In Proceedings of 10th IAPR International Workshop on Document Analysis Systems (DAS), pages 297-301, March 27-29, 2012, Queensland, Australia.

Bibliography

- [1] Digital library of india. <http://dli.iit.ac.in/>.
- [2] Lucene. <http://lucene.apache.org/>.
- [3] The universal library. <http://www.ulib.org>.
- [4] Project guten berg. at <http://www.guten berg.org>, 2003.
- [5] C. B. A. Negi and B. Krishna. An ocr system for telugu. In *International Conference on Document Analysis and Recognition*, 2001.
- [6] A. Abidi, A. Jamil, I. Siddiqi, and K. Khurshid. Word spotting based retrieval of urdu handwritten documents. In *International Conference on Frontiers in Handwriting Recognition*, 2012.
- [7] A. Abidi, I. Siddiqi, and K. Khurshid. Towards searchable digital urdu libraries - a word spotting based retrieval approach. In *International Conference on Document Analysis and Recognition*, pages 1344–1348, 2011.
- [8] W. Alemu and S. Fuchs. Handwritten amharic bank check recognition using hidden markov random field. In *Document Image Analysis and Retrieval Workshop*, 2003.
- [9] A. Arora and A. M. Namboodiri. A semi-supervised svm framework for character recognition. In *International Conference on Document Analysis and Recognition*, pages 1105–1109, 2011.
- [10] D. Arya, C. Jawahar, C. Bhagvati, T. Patnaik, B. Chaudhuri, G. Lehal, S. Chaudhury, and A. Ramakrishna. Experiences of integration and performance testing of multilingual ocr for printed indian scripts. In *JMOCR and AND*, page 9, 2011.
- [11] E. Ataer and P. Duygulu. Retrieval of ottoman documents. In *Multimedia Information Retrieval*, pages 155–162, 2006.
- [12] A. Balasubramanian, M. Meshesha, and C. V. Jawahar. Retrieval from document image collections. In *IAPR International Workshop on Document Analysis Systems*, pages 1–12, 2006.
- [13] A. Balasubramanian, M. Meshesha, and C. V. Jawahar. Retrieval from document image collections. In *IAPR International Workshop on Document Analysis Systems*, pages 1–12, 2006.
- [14] V. Bansal and R. M. K. Sinha. A complete OCR for printed hindi text in devanagari script. In *International Conference on Document Analysis and Recognition*, 2001.
- [15] H. Bay, A. Ess, T. Tuytelaars, and L. J. V. Gool. Surf: Speeded up robust features. *Computer Vision and Image Understanding*, 110(3):346–359, 2008.

- [16] A. Bhaskarbhata, S. Madhavanath, M. P. Kumar, A. Balasubramanian, and C. V. Jawahar. Representation and annotation of online handwritten data. In *International Workshop on Frontiers in Handwriting Recognition*, 2004.
- [17] S. H. K. C. Y. Suen, S. Mori and C. H. Leung. Analysis and recognition of asian scripts - the state of the art. In *International Conference on Document Analysis and Recognition*, 2003.
- [18] J. Chan, C. Ziftci, and D. A. Forsyth. Searching off-line arabic documents. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1455–1462, 2006.
- [19] B. B. Chaudhuri and U. Pal. An OCR system to read two indian language scripts: Bangla and devanagari (hindi). In *International Conference on Document Analysis and Recognition*, 1997.
- [20] B. B. Chaudhuri and U. Pal. A complete printed bangla ocr system. *Pattern Recognition*, 31(5):1997, 531-549.
- [21] S. Chaudhury, G. Sethi, A. Vyas, and G. Harit. Devising interactive access techniques for indian language document images. In *International Conference on Document Analysis and Recognition*, pages 885–889, 2003.
- [22] O. Chum, J. Philbin, J. Sivic, M. Isard, and A. Zisserman. Total recall: Automatic query expansion with a generative feature model for object retrieval. In *IEEE Conference on Computer Vision*, pages 1–8, 2007.
- [23] J. Cowell and F. Hussain. Amharic character recognition using a fast signature based algorithm. In *International Conference on Information Visualization*, 2003.
- [24] S. Deorowicz. Solving longest common subsequence and related problems on graphical processing units. *Software - Practice and Experience*, 40(8):673–700, 2010.
- [25] R. O. Duda, P. Hart, and D. Stork. *Pattern Classification*. John Wiley & Sons, 2001.
- [26] A. Dutta and S. Chaudhury. Bengali alpha-numeric character-recognition using curvature features. *Pattern Recognition*, 26(12):1757–1770, 1993.
- [27] J. Edwards, Y. W. Teh, D. A. Forsyth, R. Bock, M. Maire, and G. Vesom. Making latin manuscripts searchable using ghms. In *Neural Information Processing Systems*, 2004.
- [28] R. T. Hartley and K. Crumpton. Quality of ocr for degraded text images. In *ACM Conference on Digital libraries*, 1999.
- [29] Y. Ishitani. Model-based information extraction method tolerant of OCR errors for document images. In *International Conference on Document Analysis and Recognition*, 2001.
- [30] A. K. Jain and A. M. Namboodiri. Indexing and retrieval of on-line handwritten documents. In *International Conference on Document Analysis and Recognition*, pages 655–659, 2003.
- [31] R. Jain and D. S. Doermann. Logo retrieval in document images. In *IAPR International Workshop on Document Analysis Systems*, pages 135–139, 2012.
- [32] C. V. Jawahar and A. Kumar. Content-level annotation of large collection of printed document images. In *International Conference on Document Analysis and Recognition*, pages 799–803, 2007.

- [33] C. V. Jawahar, M. N. S. S. K. P. Kumar, and S. S. R. Kiran. A bilingual ocr for hindi-telugu documents and its applications. In *International Conference on Document Analysis and Recognition*, 2003.
- [34] C. V. Jawahar, M. Meshesha, and A. Balasubramanian. Searching in document images. In *Indian Conference on Vision, Graphics and Image Processing*, pages 622–627, 2004.
- [35] K. Jithesh, K. G. Sulochana, and R. R. Kumar. Optical character recognition (OCR) system for malayalam language. In *National Workshop on Application of Language Technology in Indian Languages*, 2003.
- [36] A. L. Kesidis and B. Gatos. Efficient cut-off threshold estimation for word spotting applications. In *International Conference on Document Analysis and Recognition*, pages 279–283, 2011.
- [37] V. Kluzner, A. Tzadok, D. Chevion, and E. Walach. Hybrid approach to adaptive ocr for historical books. In *International Conference on Document Analysis and Recognition*, pages 900–904, 2011.
- [38] T. Konidaris, B. Gatos, K. Ntzios, I. Pratikakis, S. Theodoridis, and S. J. Perantonis. Keyword-guided word spotting in historical printed documents using synthetic data and user feedback. *International Journal on Document Analysis and Recognition*, 9(2-4):167–177, 2007.
- [39] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2169–2178, 2006.
- [40] D. S. Lee and R. Smith. Improving book OCR by adaptive language and image models. In *IAPR International Workshop on Document Analysis Systems*, pages 115–119, 2012.
- [41] X. Lin. DRR research beyond COTS OCR software: A survey. In *In SPIE Conference on Document Recognition and Retrieval XII*, 2005.
- [42] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.
- [43] K. Marukawa, T. Hu, H. Fujisawa, , and Y. Shima. Document retrieval tolerating character recognition errorsevaluation and application. *Pattern Recognition*, 30(7):1997, 1361-1371.
- [44] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *British Machine Vision Conference*, 2002.
- [45] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2161–2168, 2006.
- [46] U. Pal and B. B. Chaudhuri. Indian script character recognition: A survey. *Pattern Recognition*, 37(9):1887–1899, 2004.
- [47] T. M. Rath and R. Manmatha. Features for word spotting in historical manuscripts. In *International Conference on Document Analysis and Recognition*, pages 218–222, 2003.
- [48] T. M. Rath and R. Manmatha. Word image matching using dynamic time warping. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 521–527, 2003.
- [49] T. M. Rath and R. Manmatha. Word spotting for historical documents. *International Journal on Document Analysis and Recognition*, pages 139–152, 2007.

- [50] P. P. Roy, J.-Y. Ramel, and N. Ragot. Word retrieval in historical document using character-primitives. In *International Conference on Document Analysis and Recognition*, pages 678–682, 2011.
- [51] M. Rusiñol, D. Aldavert, R. Toledo, and J. Lladós. Browsing heterogeneous document collections by a segmentation-free word spotting method. In *International Conference on Document Analysis and Recognition*, pages 63–67, 2011.
- [52] J. M. S. Belongie and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 509–522, 2002.
- [53] R. Saabni and A. Bronstein. Fast key-word searching via embedding and active-dtw. In *International Conference on Document Analysis and Recognition*, pages 68–72, 2011.
- [54] K. P. Sankar, V. Ambati, L. Pratha, and C. V. Jawahar. Digitizing a million books: Challenges for document analysis. In *IAPR International Workshop on Document Analysis Systems*, pages 425–436, 2006.
- [55] K. P. Sankar, C. V. Jawahar, and R. Manmatha. Nearest neighbor based collection ocr. In *IAPR International Workshop on Document Analysis Systems*, pages 207–214, 2010.
- [56] G. Schroth, S. Hilsenbeck, R. Huitl, F. Schweiger, and E. G. Steinbach. Exploiting text-related features for content-based image retrieval. In *ISM*, pages 77–84, 2011.
- [57] S. Setlur and V. Govindaraju, editors. *Guide to OCR for Indic Scripts*. 2009.
- [58] S. Setlur and V. Govindaraju(editors). Guide to OCR for indic scripts. In *Springer*, 2009.
- [59] R. Shekhar and C. V. Jawahar. Word image retrieval using bag of visual words. In *IAPR International Workshop on Document Analysis Systems*, pages 297–301, 2012.
- [60] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *IEEE Conference on Computer Vision*, volume 2, pages 1470–1477, 2003.
- [61] D. Smith and R. Harvey. Document retrieval using SIFT image features. *Journal of Universal Computer Science*, 17(1):3–15, 2011.
- [62] M. Sridha, D. Mandalapu, and M. Patel. Active-DTW : A generative classifier that combines elastic matching with active shape modeling for online handwritten character recognition. In *International Conference on Frontiers in Handwriting Recognition*, 1999.
- [63] S. N. Srihari, H. Srinivasan, C. Huang, and S. Shetty. Spotting words in latin, devanagari and arabic scripts. *Vivek: Indian Journal of Artificial Intelligence*, 16(3):2006, 2-9.
- [64] K. Taghva, J. Borsack, and A. Condit. Evaluation of model-based retrieval effectiveness with OCR text. *ACM Trans. Inf. Syst.*, 14(1):1996, 64-93.
- [65] 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 *International Conference on Document Analysis and Recognition*, 2011.
- [66] J. van Gemert, J.-M. Geusebroek, C. J. Veenman, and A. W. M. Smeulders. Kernel codebooks for scene categorization. In *European Conference on Computer Vision*, pages 696–709, 2008.

- [67] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms, 2008. <http://www.vlfeat.org/>.
- [68] J. Wang, J. Yang, K. Yu, F. Lv, T. S. Huang, and Y. Gong. Locality-constrained linear coding for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3360–3367, 2010.
- [69] I. Z. Yalniz and R. Manmatha. An efficient framework for searching text in noisy document images. In *IAPR International Workshop on Document Analysis Systems*, pages 48–52, 2012.
- [70] J. Yang, K. Yu, Y. Gong, and T. S. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1794–1801, 2009.
- [71] A. Yaregal and J. Bigun. Ethiopic character recognition using direction field tensor. In *International Conference on Pattern Recognition*, 2006.
- [72] L. Zhidong, S. Richard, N. Premkumar, B. Issam, and M. John. Advances in the bbn byblos OCR system. In *International Conference on Document Analysis and Recognition*, 1999.