Annotation of Images and Videos based on Textual Content without OCR

Pramod Sankar K., Million Meshesha, C. V. Jawahar

Centre for Visual Information Technology,
International Institute of Information Technology, Hyderabad, India
jawahar@iiit.ac.in

Abstract. Most of the existing CBIR and CBVR techniques do not scale well to large collections of images and videos, requiring large retrieval times. The success of text retrieval has prompted many to reconsider text based search, built upon image and video annotations. Embedded text within an image could be used for annotation, but the lack of reliable OCRs for many languages is a handicap. We propose a framework where word-images extracted from visual documents are matched with keyword images. The text equivalent of the keywords is used to annotate the image and an index is built upon these annotations. To overcome the lack of language processing modules, we use text clustering to identify words with the same stem, and cluster the corresponding word images. Our techniques are applicable to various types of visual documents such as images from WWW, broadcast video recordings and scanned document images.

1 Introduction

Content based image retrieval (CBIR) [1], and Content based video retrieval (CBVR) algorithms have been focusing on content-level access to visual data. The CBIR and CBVR techniques were used to build many successful prototype systems for indexing and retrieval of multimedia data. However, Content based retrieval (CBR), which is based on feature matching between a query image and those in the database, is highly computationally intensive. If an image has l interest points represented by a d-vector, with N images to search, each query would require $O(d^2.l^2.N)$ computations. Even for moderately large N, the search time would be many seconds, if not minutes or hours. However, since users prefer quick search engines, the popular systems are still text-based. In these systems, images and videos are annotated with the surrounding text, assuming that the image will be described in the text. A complementary approach would be to use the textual content *within* an image or video for annotation. We present a framework that enables efficient annotation of multimedia data, over which a text based retrieval system could be built. The techniques we discuss encompass a wide range of visual documents.

About 42% of WWW images contain text, of which 59% contain at least one word that does not appear in the corresponding text [2]. Videos, especially broadcast news have overlaid text that specificies the name of the subject, story title etc. For experimentation purposes, we have developed a system which directly processes news videos from various channels, in different languages [3]. Another category of visual data that pre-dominantly consists of text is that of document images. Many projects are underway across the globe to scan books and store them in large digital libraries. Digital Library of India (DLI) [4], has so far digitized 115,496 books and made them available online. In context of DLI, we could estimate the retrieval time. In 34M page images there would be about *10 billion* words. Let us optimistically assume that it takes about 0.1 second to compare a set of ideal features of one image to another. Comparing a query with every other word would need 10^9 seconds or 31.7 years. The complete library will be ten times larger with 1M books, and therefore, an order of magnitude improvement in feature matching will still not help.

Though an OCR [5, 6] could be used to recognize the text, limited OCR accuracy, especially in degraded images [7], is a serious handicap. Moreover, many Indian, African and Arabic languages do not have reliable OCRs. The problem is accentuated by the lack of language processing modules as well, in these languages. The retrieval from hand written documents has very similar constraints. A successful approach [8] was to avoid explicit recognition, by performing matching in the image space. We had adapted this approach in [9] to build a system that accepts a textual query, renders it to a word image and performs matching with other word images in the collection. The close matches are retrieved for the user. However, image matching is computationally intensive and thus time consuming when performed over large collections.

Instead of using image matching for retrieval, we use it for automatic annotation. We obtain a set of textual keywords and render them to obtain word images which are matched with the extracted text. The images are annotated by the keyword they match with. Unlike the image being processed to obtain the annotation, we find appropriate images and videos that a keyword can annotate. We call this scheme *Reverse Annotation.* Our framework is depicted in Figure 1. We build a repository of visual documents from different types of documents and index these documents using the approach described in this paper. The focus here is on *word image clustering* and *text clustering* for building a text based search index. For sake of clarity, our previous work is depicted by dotted lines.

The contributions in this work are: (i) We propose an annotation scheme called Reverse Annotation, where keywords are assigned to images and videos based on their textual content (ii) We design an indexing scheme based on word image matching techniques and (iii) We formulate a text clustering approach to overcome the lack of language processing modules such as stemmers. For the rest of this discussion we shall use "image" to mean both image and video, unless specified explicitly, since the concepts we discuss apply uniformly to both media.

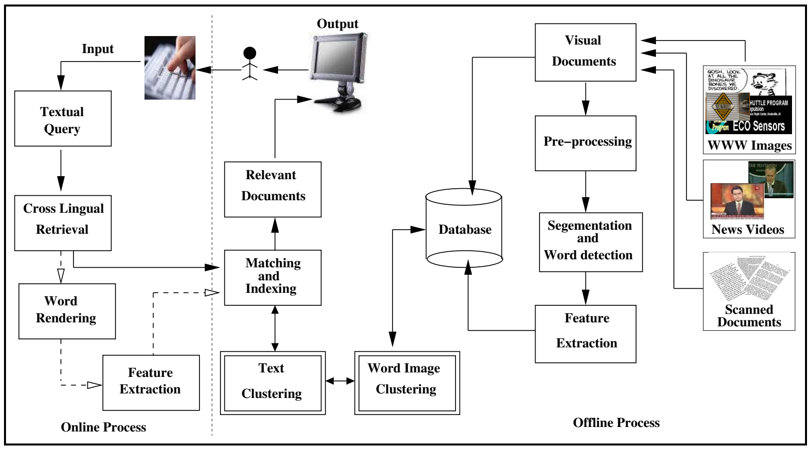


Fig. 1. Framework for Searching Visual Documents

2 Annotation and Indexing of Visual Documents

Any representation of an image, or visual document, in a form that makes it search-able in a database, could be called the annotation of the image. However, in the literature, annotation has come to mean assigning textual keywords to documents. Indexing is the mechanism by which the representation is stored to efficiently search for a query, and retrieve the appropriate documents. We shall briefly discuss below the various annotation and indexing schemes for images.

Legacy Systems: The early image retrieval systems were basically databases extended to handle images. The images were files in a directory, accessed by linear search or entries in a DBMS. Retrieval systems were designed for large number of records, but for limited number of attributes. Hash tables enabled search in $O(logN)$, but multidimensional indexing is computational and memory intensive. Since the data had to be exact strings, images were manually text-annotated. Systems were built only for specific domains such as geographic images, maps and medical images [10], where human effort was affordable. However, the cost and subjectivity of manual annotation and the constraint of exact match were shortcomings of this approach. Text-based retrieval systems, such as Google, use surrounding text of an image for annotation. In this approach, each image is annotated by a large number of words that are not related to it. The results are often not satisfactory with many false positives even for such regular queries as *aircraft*, *car* etc [11].

Feature Based Representation and Retrieval: Content based retrieval enabled querying by an example image to retrieve *similar* images. Global features (histograms, correlograms, geometric histograms) and/or local features (interest points, textures, silhouette, spatial layout) are extracted from the images and a similarity measure is computed. Close matches with the query are retrieved for the user. Images are often segmented into objects to compute features, but segmentation in generic images is not yet reliable. In case of videos, representative frames of video shots are identified [12] over which a CBIR system is applied. However, shot and key frame detection techniques are not very robust. The time complexity of feature extraction is generally $O(w^2)$ for images (w is width of image) and $O(w^2.n)$ for videos (n is number of frames). $O(d^2)$ computations are required for comparing two d-vectors, which is repeated on k images in the collection. For videos, if there are m videos with k shots each, the complexity is $O(m.k.d^2)$. Since, this order of computation is performed after the user submits a query, interactive retrieval is out of reach.

To retrieve multi dimensional data, data structures were designed for search and retrieval efficiency. For similarity-based databases, tree-based index ($O(logN)$) is used. k-d trees, Quad-trees, R-trees or their variants (R*-trees, hB-trees, X-trees etc) and entropy based balancing of the k-d tree show incremental improvements in performance. However, the performance of these data structures degrades rapidly with increase in dimensionality. For example boosting techniques compute as many as 10,000 features, while in a wavelet histogram, each pixel has a 9-d vector compressed to 512^2 histograms of 512 bins per image [13]. Considering this against the fact that the performance of R-trees degrades by a factor of 12 as the number of dimensions doubles, the indexing schemes do not scale as required. Geometric hashing was used [14] to efficiently retrieve large dimensional data, but the retrieval time is several seconds.

Back to Text-Based Retrieval: Though earlier objective in image and video retrieval was to enable retrieval, recently, scalability, performance and computational complexity issues are the major concerns (due to the massive increase in image collections). Instead of using features, if images are annotated with text, they can be efficiently indexed and reliably retrieved using existing text-search schemes. There is thus a renewed interest in text based search with focus on automatic annotation of images. Soft annotation using Bayes Point machines[15], 2-d HMMs [16], generative language models and word-to-word correlations were explored. The LSA model was used to identify semantically meaningful subspaces in the visual-textual feature space[17].

The approach used in this paper is an extension of Google's strategy of textual annotation of multimedia. Instead of annotating with text around an image, we annotate based on textual content *within* images. However, we don't need an OCR to recognize the text. We identify a set of keywords useful for annotation and render them to obtain the pre-defined objects that are detected in the image. We call this approach of finding appropriate images to be labeled by a particular keyword as *Reverse Annotation.* Our work is similar to that of [18], except that instead of searching for learned visual words, we are looking for words that come from a well defined alphabet.

3 Annotation of Word Images

Text Extraction Text extraction from images and videos is well known in the literature [19]. We pre-process the image to remove noise, skew etc and use a segmentation algorithm to extract text at the word level. The extracted text is tagged with the source image and its location in it.

Feature Extraction Since the images come from a wide variety of sources, the text is varied in font type, style, size, image quality etc. The invariance toward these is achieved by appropriate choices of features and similarity metric. We extract word profiles (upper word profile, lower word profile, projection profile, transition profile) and structural features (mean, standard deviation, region-based moments, zeroth- and first-order moments) which result in a sequence of feature vectors for each word image. Features are normalized by the maximum value for each vector to make them font size independent.

3.1 Similarity Measure and Clustering

The similarity measure between the extracted features is computed by Dynamic Programming (DP) between the corresponding feature vectors. The advantages of using a DP approach are: (i) It preserves the ordering information by enforcing a local continuity constraint, (ii) It minimizes a cumulative distance consisting of available local distances, (iii) It can easily handle features of different lengths, (iv) Covariance and other statistical measures are not required over the entire collection, (v) It is quickly computable - $O(m.n)$ where m, n are feature lengths, and (vi) Considerable speed up can be achieved by constraining the search space in the DP array. An example of matching the upper word profiles of two images "direct" and "redirected" is shown in Figure 2 (a). The points where the profiles match contribute to the matching score. This is normalized with the length of the vector, and the process is repeated over all the feature vectors. The similarity measure thus computed is used for clustering.

Since the word images do not belong to a vector space only pairwise distances can be obtained. We therefore use hierarchical agglomerative clustering (HAC) since (i) It works based on pair-wise distances (ii) The thresholds can be easily set and quickly evaluated (iii) The procedure is inherently incremental, thus allowing for new images to be added to existing clusters and (iv) Is very efficient in computation - $O(N^2), N$ being the number of word images. We begin by assigning a singleton cluster to each image. At each iteration we find close clusters and merge them. The iterations terminate when clusters cannot be merged any further. During clustering we rank the documents based on relevance computed by the term-frequency (TF) / inverse document frequency measure (IDF). TF counts the occurrence of a word in a document, and IDF measures the inverse of number of documents containing the word. IDF is given by $IDF = log_2 N - log_2 d_j$, where N is the number of documents, and d_j is the number of documents containing the word j. We remove the stop-words, which usually have high TF and less IDF. The closest word to the other words in the cluster is the representative word.

$$c_i = \min_j \forall(j, k \in C_i) D[j, k]$$

The clustering algorithm is described as follows

Algorithm: Clustering Word Images

$D[i, j]$ stores the similarity measure between words w_i and w_j, τ is a similarity threshold and α is a language specific measure for stop words.

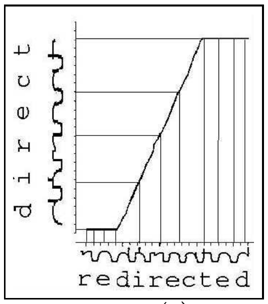
```
1. Initialize n clusters, $C_1, C_2, ..., C_n$, each cluster containing one word
2. For each pair of words $w_i$ and $w_j$ with $D[i, j] < \tau$
     – add $w_j$ to the cluster of $w_i$ namely, $C_i$
3. Identify cluster centroid $c_i$
4. Compute $IDF = log_2 N - log_2 d_k$
5. If $IDF < \alpha$,
   then reject the cluster as stop-word cluster; update the stop-word list
6. Measure the relevance of each cluster using the cluster centroid $c_i$
7. Sort words in descending order of their relevance and rank them

End
```

Incremental Clustering After obtaining $C_1, C_2, ..., C_m$ from above, newer words are incrementally clustered with the existing ones. We maintain that the radius of a cluster C_i, $max_{r \in C_i} d(c_i, r)$ is less than a threshold λ, and the inter-cluster distance $d(c_i, c_j)$ is at least γ. For each new word w_x to be clustered we do the following: (i) Check if w_x is a stop word, (ii) For each c_i compute the DP distance between $d(c_i, w_x)$, (iii) If $min(d) < \lambda$ then add w_x to cluster with min d and recompute relevance of documents else create new cluster for w_x

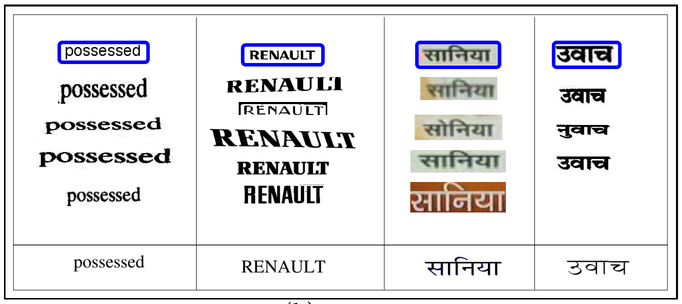
For incremental clustering the computational complexity is $O(n.C)$, where n is number of words to be clustered and C is the number of clusters. Since the number of clusters is much smaller than the number of words, this procedure is much efficient compared to the $O(n^2)$ re-clustering algorithm. Also, it allows for reusing the existing clusters, thereby not wasting the computing effort that was required to build them.

3.2 Reverse Annotation of Clusters

The word image clusters could be used to build a search index, where documents are annotated by the cluster representatives. When a word is searched for, it is rendered and the image is matched with the cluster centroids. Documents from the matched cluster are retrieved [9]. Since this online process is computationally expensive, we would like to build a text index for each document using the word image clusters. Manual annotation of clusters is prohibitive and OCRing the representative words is not accurate. The solution we propose is what we call Reverse Annotation.



(a)



(b)

Fig. 2. (a)DTW based Matching of feature vectors (b) Example word image clusters - the representative image is highlighted, and the textual keyword is given underneath each cluster

We obtain a collection of unique words from a text corpus of the language, and render them to obtain word images. Features are extracted and DP matching is used to compare the keyword image against the representative words of each cluster. The cluster that is closest to a keyword image is annotated by the keyword. We thus obtain a textual equivalent for each word cluster and thus a text representation for each image. We have essentially used feature based techniques to annotate images, so that a text-based search could be enabled. A few example clusters are shown in Figure 2 (b). The clusters consist of examples from WWW images, videos and scanned documents. Example clusters from English and Hindi are shown with the textual word they are annotated with. The cluster centroid is marked in blue.

4 Overcoming the Lack of Language Processing Modules

We had so far discussed the framework that enables annotation in the absence of an OCR. In this section we address the problem of the lack of language processing modules (LPM). A LPM such as a morphological analyzer segments a given word into prefix, stem-word, suffix and gives information regarding gender, number and tense of the subject. During index building, prefixed and suffixed words are indexed by the stem word only. This ensures that documents containing "talk", "talking", "talked" etc. are retrievable for the query "talk". This improves the recall for the retrieval system, by indexing documents which would otherwise be indexed by a different word.

As shown in Figure 3 (a), the words *possess*, *possessed*, *possessing* etc are clustered separately. This is because our features and similarity measure handle font size and style variations, but not word-form variations. We could extend the DP algorithm to perform partial matching between features [20]. Instead of using the score at $D[M, N]$, we could backtrack to find the most optimal path from $[M, N]$ to $[0, 0]$ in the DP array, the incline shown in Figure 2. By eliminating the horizontal and vertical portions we compute the cost of the optimal path which is the partial matching score between the two feature vectors. However, we would like to avoid computation in image domain and use the quicker text domain techniques instead.

4.1 Combining Image and Text Clusters

The common sub-sequence length is used to cluster words with the same stem. The edit distance accounts for only insertions and deletions at the beginning and end. Substitution is considered to change the stem word. We do not make any assumptions regarding the prefixes and suffixes of the language and their lengths. However, the measure does not distinguish between different stem words that overlap. For example *ease, cease, eased, lease, tease, ceased* are all treated as compound words of "ease", while *cease, lease, tease* are valid stem words themselves. To handle such cases we chose to allow soft-clustering, where one word could belong to more than one cluster. The clustering proceeds as:

```
Given: a set of words $W_1, W_2, ..., W_n$ and a threshold $\tau$

1. For each word $W_i$
   (a) Find all $W_j$ where $d(W_i, W_j) < \tau$
   (b) Add all $W_j$ to the cluster $C_i$
2. Compute cluster representatives $c_i$
   End
```

We set the similarity threshold τ based on examining a few examples. The obtained clusters are used to estimate the number of stem words in the language. Since search index consists of stem words and many nouns, this gives us an idea of the size of the index. We used a dictionary of 105902 English words of which 89208 words have no prefixes or suffixes. The remaining words were soft-clustered to obtain 16694 clusters. For Telugu, we collected 63287 words from a limited text corpus, encoded in *itrans*. The clustering resulted in 51610 single words and 11677 clusters. The most frequently occurring prefixes and suffixes in English are *-ed, -ing, -d, -er, -ly*; *in-, re-, un-, dis-, de-*, and in Telugu are *-nu, -na, -mu, -a, -ni*; *n-, y-, ya-, pra-, m-*. The knowledge of the possible prefixes and suffixes and their distribution for each word is an important input for building automatic morphological analyzers. However, we have not yet dealt with changes in tense, i.e., words like *make* and *made* are still considered as belonging to different clusters.

The text clusters were evaluated against ground truth obtained from a morphological analyzer of English words. The morph contains 317273 words along with their stems. 84% of the words in the morph and 93% of the words in the corpus were found to belong to the cluster of their stem word. The learnt prefixes and suffixes were used to identify the stem word, which coincided with the actual stem word 78% of times. Thus, the text clusters provide a decent approximation of the morphological analyser.

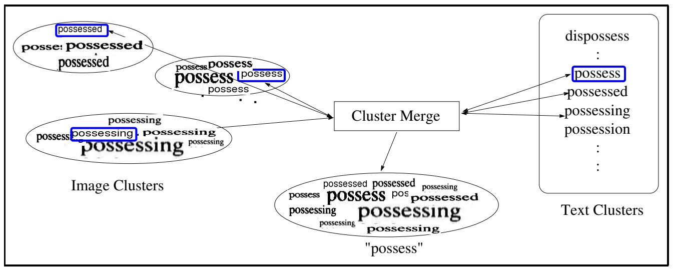


Fig. 3. Merging word image clusters based on text clustering

The text clusters are used to refine the annotations of word image clusters. All word image clusters that belong to the same text cluster are merged and annotated with the representative word of the text cluster. This has a drastic reduction in the number of index terms and thus better performance of the retrieval system. An example is shown in Figure 3 (a), where the the different image clusters corresponding to *possess, possesses and possessing* are merged into one cluster named “possess”. The relevance of documents based on TF-IDF is recomputed and the documents are sorted accordingly. The annotations of the word image are propagated back to the source image using the tag information.

4.2 Search and Retrieval from Annotations

From the labels of the clusters we have a textual annotation for each image. A search engine can be built by considering each image as a document of these annotations. The index file is the list of annotations, where each index term points to all source images for the words in the cluster. When a query is given, the index is looked up and the corresponding images are retrieved. To enable multi-lingual retrieval, we use transliteration-based and dictionary-based translation schemes to obtain multi-lingual equivalents to the search word. Each of these multiple search queries is separately searched for in the index, and the corresponding images are retrieved [9]. Using a collection of 3000 word images, the retrieval speed of the system was 0.16 seconds to search and retrieve relevant documents and 0.34 seconds to transfer the image to the user. The search speed is thus at par with general text retrieval systems.

5 Annotating non-text images using Semi-Supervised Learning

The framework presented above works well in the case of images containing text. However, it is limited to precisely those images which have extractable text. To annotate *non-text* images we could use the annotations of similar images with textual content. Semi supervised learning which uses both labeled and unlabeled data, is ideally suited for our purposes. We pose the annotation of

No. of constraints	Precision	Recall	Mutual information
0	0.3391	0.35963	0.01845
28	0.373	0.567	0.074
105	0.6079	0.6146	0.3826
153	0.5949	0.6091	0.3569
528	0.5902	0.5789	0.3892
703	0.6682	0.7835	0.5742
820	0.8718	0.8734	0.7682
1431	0.70408	0.6963	0.5743

Table 1. Improvement of cluster measures with annotation constraints

images as a labeling problem, with the images that were already annotated as labeled examples, and non textual images as unlabeled. A weighted graph is constructed with vertices as labeled and unlabeled images and the edge weights being a feature-based distance metric. The learning problem is formulated as a Hidden Markov Random Field (HMRF), and the objective function for semi supervised clustering is derived from the posterior energy of the HMRF.

We applied this framework to learn annotations of faces from videos based on text annotated frames. We used a popular face detector over 100 images of 3 famous personalities, and applied Euclidean distance over the eigen faces. The metric pairwise constrained K-means algorithm was used with the constraints obtained from image annotations. When two images have the same annotation, we constrain them as *must-link* in the HMRF and images with different annotations as *cannot-link*. Thus, with n annotated images we can extract $n.(n-1)/2$ constraints. The clusters are evaluated based on ground truth, and precision, recall and mutual information measures are calculated. These measures improve considerably with increase in number of constraints as shown in Table 1.

The best results were with 820 constraints, or 41 annotated images. We thus get satisfactory results with about 41% annotated images. With a further increase in number of constraints, the performance drops, which is due to the over constraining of points resulting in a local minima. It should be noted that the face images were not corrected for pose and illumination variations, which results in poor accuracy of the classifier, as reflected in the precision-recall with zero constraints. Better results are expected when these variations are handled. Though the distance metric is very primitive, the annotations help in properly clustering the images. A few samples from one of the clusters are shown in Figure 4. Images in the cluster are annotated by the label of the cluster. Further formalization and analysis of this stage are part of a subsequent work.

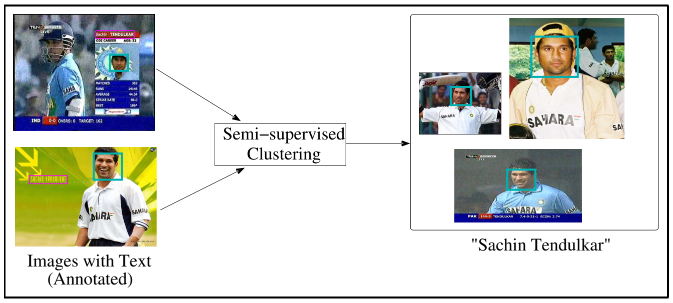


Fig. 4. Example of semi-supervised clustering to annotate non-text images

6 Conclusion

We have successfully applied the reverse annotation framework to annotate and index a large collection of document images from the digital library. We have enabled multi-lingual search and the engine is available online [9]. Though what we have proposed is in itself a computational intensive method, it should be noted that all the computation needs to be done offline, just once. Using the annotations at the end of this procedure, the online retrieval will be an efficient text-based search. Relying on the scalability of text search, our approach will enable quick and efficient retrieval from large collections of images and videos.

References

1. Datta, R., Li, J., Wang, J.Z.: Content-based image retrieval: approaches and trends of the new age. In: Proc. ACM SIGMM International Workshop on Multimedia Information Retrieval. (2005) 253–262
2. Kanungo, T., Lee, C.H., Bradford, R.: What fraction of images on the web contain text? In: First International Workshop on Web Document Analysis (WDA2001), Seattle, Washington, USA (Sept. 2001)
3. Jawahar, C., Balakrishna, C., Balamanohar, P., Nataraj, J.: Video retrieval based on textual queries. In: 13th International conference on Advanced Computing and Communication (ADCOM). (2005)
4. Pramod Sankar, K., Vamshi Ambati, Lakshmi Pratha, Jawahar, C. V.: Digitizing a million books: Challenges for document analysis. In: 7th International Workshop on Document Analysis Systems, DAS. (2006) 425–436
5. Lopresti, D., Zhou, J.: Locating and recognizing text in www images. Information Retrieval **2** (2000) 177–206
6. Sato, T., Kanade, T., Hughes, E., Smith, M., Shin-ichi Satoh: Video OCR: Indexing digital news libraries by recognition of superimposed caption. In: ACM Multimedia Systems Special Issue on Video Libraries. (1998)
7. Doermann, D.: The indexing and retrieval of document images: A survey. In: Computer Vision and Image Understanding (CVIU) 70. (1998) 287–298
8. Rath, T., Manmatha, R.: Word image matching using dynamic time warping. Proc. Computer Vision and Pattern Recognition (CVPR) **2** (2003) 521–527

9. Balasubramanian, A., Million Meshesha, Jawahar, C.V.: Retrieval from document image collections. In: 7th International Workshop on Document Analysis Systems, DAS. (2006) 1–12
10. Chang, S.K., Hsu, A.: Image information systems: Where do we go from here? IEEE Trans. on Knowledge and Data Engineering (1992)
11. Fergus, R., Fei-Fei, L., Perona, P., Zisserman, A.: Learning object categories from google's image search. In: Proc. International Conference on Computer Vision. (2005) 1816–1823
12. Zhang, H., Kankanhalli, A., Smoliar, S.: Automatic partitioning of full-motion video. Multimedia Systems **1** (June 1993) 10–28
13. Smeulders, A., Worring, M., Santini, S., Gupta, A., Jain, R.: Content-based image retrieval at the end of the early years. IEEE Trans. Pattern Analysis and Machine Intelligence **22** (2000) 1348
14. Schmid, C., Mohr, R.: Local grayvalue invariants for image retrieval. IEEE Transactions on Pattern Analysis & Machine Intelligence **19** (1997) 530–534
15. Chang, E., Kingshy, G., Sychay, G., Wu, G.: CBSA: content-based soft annotation for multimodal image retrieval using bayes point machines. IEEE Trans. on Circuits and Systems for Video Technology (2003) 26–38
16. Li, J., Wang, J.: Automatic linguistic indexing of pictures by a statistical modeling approach. IEEE Trans. on Pattern Anal. and Mach. Intell. **25** (2003) 1075–1088
17. Monay, F., Gatica-Perez, D.: On image auto-annotation with latent space models. In: Proc. ACM Int. Conf. on Multimedia (ACM MM). (2003)
18. Sivic, J., Zisserman, A.: Video Google: A text retrieval approach to object matching in videos. In: Proceedings of the International Conference on Computer Vision. Volume 2. (2003) 1470–1477
19. Jung, K., Kim, K.I., Jain, A.K.: Text information extraction in images and video: A survey. Pattern Recognition **37** (May 2004) 977–997
20. Jawahar, C.V., Million Meshesha, Balasubramanian, A.: Searching in document images. In: 4th Indian Conference on Computer Vision, Graphics and Image Processing (ICVGIP). (2004) 622–627