Image Retrieval using Textual Cues

Anand Mishra[1] Karteek Alahari[2,*] C. V. Jawahar[1]

[1]CVIT, IIIT Hyderabad, India [2]Inria, France

Abstract

We present an approach for the text-to-image retrieval problem based on textual content present in images. Given the recent developments in understanding text in images, an appealing approach to address this problem is to localize and recognize the text, and then query the database, as in a text retrieval problem. We show that such an approach, despite being based on state-of-the-art methods, is insufficient, and propose a method, where we do not rely on an exact localization and recognition pipeline. We take a query-driven search approach, where we find approximate locations of characters in the text query, and then impose spatial constraints to generate a ranked list of images in the database. The retrieval performance is evaluated on public scene text datasets as well as three large datasets, namely IIIT scene text retrieval, Sports-10K and TV series-1M, we introduce.

1. Introduction

It is estimated that over 380 billion photos were captured in the past 12 months, which is 10% of all the photos ever taken by humanity! In the context of such ever-growing large data collections, there are many challenging problems like searching for, and retrieving relevant content. One approach to retrieval uses *text as a query*, with applications such as Google image search, which relies on cues from meta tags or text available in the context of the image. The success of this approach is rather limited by the quality of the meta tags and the contextual text. An alternate approach like Video Google [21] enables image search using *image as a query*, by finding visually similar regions in the database. Although this method exploits the visual content, it may not necessarily be sufficient. For instance, consider two photos of restaurants shown in Figure 1. There is very little visual information to suggest that these two are images of restaurants, and thus are unlikely to be retrieved together by such methods. However, the fact that both these images contain the word *restaurant* is a very useful cue in grouping them. In this work, we aim to fill this gap in image retrieval with *text as a query*, and develop an image-search based on the textual content present in it.

Figure 1. *Consider an example query for restaurants. Here we show two images of restaurants, which have insufficient visual cues (such as building style) to group them into a single restaurant category. On the other hand, the text "restaurant" appearing on the banner/awning is an indispensable cue for retrieval. We present a text-to-image retrieval method based on the textual content present in images.*

The problem of recognizing text in images or videos has gained a huge attention in the computer vision community in recent years [5,9,13,18,19,24,25]. Although exact localization and recognition of text in the wild is far from being a solved problem, there have been notable successes. We take this problem one step further and ask the question: *Can we search for query text in a large collection of images and videos, and retrieve all occurrences of the query text?* Note that, unlike approaches such as Video Google [21], which retrieve only similar instances of the queried content, our goal is to retrieve instances (text appearing in different places or view points), as well as categories (text in different font styles).

Plausible Approaches. One approach for addressing the text-to-image retrieval problem is based on text localization, followed by text recognition. Once the text is recognized, the retrieval task becomes equivalent to that of text retrieval. Many methods have been proposed to solve the text localization and recognition problems [6, 9, 12, 13, 15]. We adapted two of these methods for our analysis with the implementation from [1, 2]. We transformed the visual text

*WILLOW project-team, Département d'Informatique de l'École Normale Supérieure, ENS/Inria/CNRS UMR 8548, Paris, France.

content in the image into text, either with [15] directly, or by localizing with [9], and then recognizing with [13]. In summary, we recognize the text contained in all images in the database, search for the query text, and then rank the images based on minimum edit distance between the query and the recognized text.

Table 1 shows the results of these two approaches on the street view text (SVT) dataset. Note that both of them fail to achieve a good performance. This poor show is likely due to the following: (i) The loss of information during localization/recognition is almost irreversible. (ii) The recognition methods are not query-driven, and do not take advantage of the second or the third best predictions of the classifier. (iii) The variation in view point, illumination, font style, and size lead to incorrect word localization and recognition. In other words, these approaches heavily rely on the localization and the recognition performance, making them susceptible to failures in both these phases.

In terms of not relying on an explicit localization, the closest to our work is [24]. Although it is a method for spotting (detecting and recognizing) one of the few ($\sim$50) lexicon words in one image. In contrast, we aim to spot query words in millions of images, and efficiently retrieve all occurrences of query. Thus our goals are different. Furthermore, the success of [24] is largely restricted by the size of the lexicon. We have performed two tests to show that adapting it to our problem is inferior to our proposed approach. (i) Using all the query words as lexicon, it gives a mean AP of 21.25% on the SVT dataset (see Table 1). (ii) Using their character detection, and then applying our indexing and re-ranking schemes, we obtain an mAP of 52.12%, about 4% lower than our approach.

Another plausible approach is based on advancements in retrieving similar visual content, e.g. bag of words based image retrieval [21]. Such methods are intended for instance retrieval with *image as a query*. It is not clear how well text queries can be used in combination with such methods to retrieve scene text appearing in a variety of styles.

Proposed Method. We take an alternate approach, and do not rely on an accurate text localization and recognition pipeline. Rather, we do a query-driven search on images and spot the characters of the words of a vocabulary[1] in the image database (Section 2.1). We then compute a score characterizing the presence of characters of a vocabulary word in every image. The images are then ranked based on these scores (Section 2.2). The retrieval performance is further improved by imposing spatial positioning and ordering constraints (Section 2.3). We demonstrate the performance of our approach on publicly available scene text datasets. For a more comprehensive study, we not only need a large dataset with diversity, but also a dataset containing multiple occurrences of text in different fonts, view points and illumination conditions. To this end, we introduce two video datasets, namely *Sports-10K* and *TV series-1M*, with more than 1 million frames, and an image dataset, IIIT scene text retrieval (STR). To our knowledge, the problem of text-to-image retrieval has not been looked at in such a challenging setting yet.

[1]We define vocabulary as a set of possible query words.

Method	**mAP**
Neumann and Matas [15]	23.32
SWT [9]+ Mishra *et al*. [13]	19.25
Wang et al. [24]	21.25

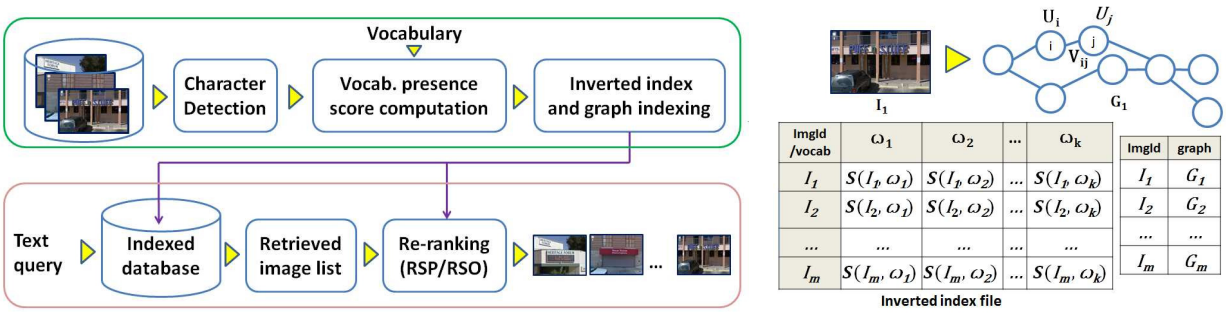
Table 1. *Baseline results for text-to-image retrieval on the street view text dataset [24] are shown as mean average precision (mAP) scores. All the unique ground truth words in the dataset are used as queries. The first two methods, based on the state-of-the-art text localization and recognition schemes, perform poorly. Wang et al. [24] is a word spotting method, which detects and recognizes the lexicon words in an image. In comparison, our approach, which does not rely on an exact localization and recognition pipeline, achieves an mAP of 56.24 (see Table 4).*

2. Scene Text Indexing and Retrieval

Our retrieval scheme works as follows: we begin by detecting characters in all the images in the database. After detecting characters, we have their potential locations. We assume that a set of vocabulary words, is given to us *a priori*. We then spot characters of the vocabulary words in the images and compute a score based on the presence of these characters. Given our goal of retrieving images from a large dataset, we need an efficient method for retrieval. To achieve this, we create an inverted index file containing image id and a score indicating the presence of characters of the vocabulary words in the image. Initial retrievals are obtained using the inverted index. We then re-rank the top-n initial retrievals by imposing constraints on the order and the location of characters from the query text. Figure 2 summarizes our indexing and retrieval scheme.

2.1. Potential Character Localization

Given a large collection of images or video frames, the first step of our retrieval pipeline is to detect potential locations of characters. We do not expect ideal character detection from this stage, but instead obtain many potential character windows, which are likely to include false positives. To achieve this, we train a linear SVM classifier with HOG features [7]. We then use a sliding window based detection to obtain character locations and their likelihoods. The character localization process is illustrated in Figure 3. Note that this is an offline step in our retrieval pipeline.



(a) (b)

Figure 2. *Summary of our indexing and retrieval scheme. (a) In the offline phase, we first detect the characters. We then compute a score indicating the presence of characters from the vocabulary words (vocabulary presence score), and create an inverted index file with this score and image id. In the online phase, user provides a query, which is searched on the indexed database to retrieve images based on the vocabulary presence score. The top-n retrievals are then re-ranked using our re-ranking schemes. (b) After character detection, an image I_m is represented as a graph G_m, where nodes correspond to potential character detections and edges model the spatial relation between two detections. The nodes are characterized by their character likelihood vector U, and the edges by their character pair priors V. This graph is used to prune false positive detections, and also to impose order and position constraints on the characters during the re-ranking phase. See Section 2 for details.*

For a robust localization of characters using sliding windows, we need a strong character classifier. The problem of classifying natural scene characters typically suffers from the lack of training data, e.g. [8] uses only 15 samples per class. It is not trivial to model the large variations in characters using only a few examples. Also, elements in a scene may interfere with the classifier, and produce many false positives. For example, the corner of a door can be detected as the character 'L'. To deal with these issues, we add more examples to the training set by applying small affine transformations to the original character images.[2] We further enrich the training set by adding many negative examples (non-characters, i.e. background). With this strategy, we achieve a significant boost in character classification.

We use a multi-scale sliding window based detector, which is popular in many applications [7,23,24]. Each window is represented by its top-left (x, y) position, width and height in the original scale. Let $\mathcal{K}$ be the set of all character classes, i.e. English characters (A-Z, a-z), digits (0-9) and a background class. Given a window i, we compute the likelihood, $P(l_i|hog_i)$, $l_i \in \mathcal{K}$, using Platt's method [16]. Here hog_i denotes the HOG features extracted from the window i. This results in a 63-dimensional vector for every window, which indicates the presence of a character or background in that window. We then perform character-specific non maximal suppression (NMS) to prune out weak windows. Further, since for a given query word, we wish to retrieve all the images where the query word appears either in upper or lower case, we transform the 63-dimensional vector to a 36-dimensional vector by taking the maximum between the upper and lower case likelihoods for every character and dropping the likelihood for background.

[2]Note that the use of affine transformations in training examples is shown to improve classification accuracy [14, 20].

2.2. Indexing

Once the characters are detected, we index the database for a set of vocabulary words. Consider a set of vocabulary words $\{\omega_1, \cdots, \omega_k\}$, which are given to us *a priori*. In a general setting, k can be as large as the number of words in English or all the words that we are interested in querying.

We first remove a few spurious character windows. To do so, we construct a graph, where each character detection is represented as a node. These nodes are connected via edges based on their spatial proximity. We then use contextual information, window width, size and spatial distance to remove some of the edges. In other words, edges between two neighbouring characters are removed if: (i) The width ratio of two neighbouring character windows exceeds θ_{width}, or (ii) The spatial distance between two character windows is more than θ_{dist}, or (iii) The height ratio of two neighbouring character windows exceeds θ_{height}. The thresholds $\theta_{width}, \theta_{dist}$ and θ_{height} are estimated from the training set. This may result in isolated nodes, which are discarded. This step essentially removes many false character windows scattered in the image.

Each node of this graph is described by a 36-dimensional vector U_i. Further, assuming these likelihoods are independent, we compute the joint probabilities of character pairs for every edge. In other words, we associate a 36×36 dimensional matrix V_{ij} containing joint probabilities of character pairs to the edge connecting nodes i and j (see Figure 2(b)).

Now, consider a word from the vocabulary $\omega_k = \omega_{k1}\omega_{k2}\cdots\omega_{kp}$, represented by its characters ω_{kl}, $1 \leq l \leq p$, where p is the length of the word. To index an image I_m for this word, we divide the image I_m into horizontal strips, each of height H. We then compute a

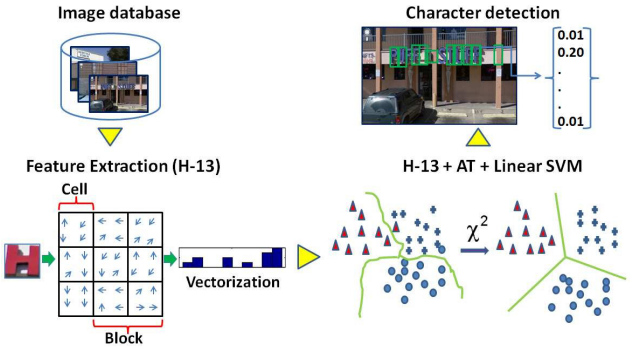


Figure 3. *Potential character localization. We compute HOG features at various scales for all the images. These features are then represented using the χ^2 kernel. A linear SVM trained on affine transformed (AT) training samples is used to obtain potential character windows. This results in a 63-dimensional vector for every window, which denotes the likelihood of every character/background class in that window.*

score denoting the presence of characters from the query in these horizontal strips. This score for an image I_m and a word ω_k, $S(I_m, \omega_k)$, is computed as the maximum over all the horizontal strips of the image. In other words, score $S(I_m, \omega_k)$ is given by:

$$\max_h \sum_{l=1}^{p} \max_j U_j(\omega_{kl}) = \max_h \sum_{l=1}^{p} \max_j P(\omega_{kl}|hog_j), \quad (1)$$

where j varies over all the bounding boxes representing potential characters whose top-left coordinate falls in the horizontal strip and h varies over all the horizontal strips in the image. To avoid the dominance of a single character, we modify the score in (1) as:

$$S(I_m, \omega_k) = \max_h \sum_{l=1}^{p} \min(\max_j P(\omega_{kl}|hog_j), \tau), \quad (2)$$

where τ is a truncation constant.

Once these scores are computed for all the words in the vocabulary and all the images in the database, we create an inverted index file [11] containing image id, the vocabulary word and its score. We also store the image and its corresponding graph (representing character detections) in the indexed database. These graphs and the associated probabilities are used in our re-ranking schemes, which we will describe in the following section.

2.3. Retrieval and Re-ranking

We use the inverted index file to retrieve the images and rank them based on the score computed in (2). This ensures that images containing characters from the query text have a high likelihood in a relatively small area (the horizontal strip of height H) get a higher rank. However, not all relevant images may be ranked well in this step, as it does not ensure the correct ordering and positioning of characters. To address this, we propose two methods to re-rank the results as follows.

Spatial ordering. Character spotting does not ensure that characters are spotted in the same order as in the query word. We address this by proposing a re-ranking scheme based on spatial ordering (RSO). Let $\psi_{total} = \{\sqcup\omega_{k1}, \omega_{k1}\omega_{k2}, \cdots, \omega_{kp}\sqcup\}$ be the set of all the bi-grams present in the query word ω_k, where $\sqcup$ denotes whitespace. We also construct a set $\psi_{present}$ containing the pairs of spatially neighbouring spotted characters. We now define the score of spatial ordering as $S_{so}(I_m, \omega_k) = \frac{|\psi_{present} \cap \psi_{total}|}{|\psi_{total}|}$, where $|\cdot|$ is the cardinality. The score $S_{so}(I_m, \omega_k) = 1$, when all the characters in the query word are present in the image, and have the same spatial order as the query word. We use this score to re-rank retrieval results.

Spatial positioning. The re-ranking scheme based on spatial ordering does not account for spotted characters being in the correct spatial position. In other words, these characters may not have uniform inter-character gap. To address this, we use the graphs representing the character detections in the images, the associated U vectors, and the matrix V to compute a new score. We define a new score characterizing the spatial positioning of characters of the query word in the image as $S_{sp}(I_m, \omega_k) =$

$$\sum_{l=1}^{p} \min(\max_i U_i(\omega_{kl}), \tau) + \sum_{l=1}^{p-1} \max_{ij} V_{ij}(\omega_{kl}, \omega_{kl+1}). \quad (3)$$

This new score is high when all the characters and bi-grams are present in the graph in the same order as in the query word and with a high likelihood. Additionally, higher value of new score ensures the correct spatial positioning of the characters. This is because the graph is constructed such that nodes representing characters spatially close to each other are connected. The retrieval results are then re-ranked based on the summation of this score and the score S_{so} obtained from spatial ordering. We refer to this scheme as re-ranking based on spatial positioning (RSP) in the paper.

2.4. Implementation Details

Character detection. We use an overlap threshold of 40% to discard weak detection windows in the non maximal suppression stage. The character classifiers are trained on the train sets of ICDAR 2003 character [3] and Chars74K [8] datasets. We harvest 48×48 patches from scene images, with buildings, sky, road and cars, which do not contain text, for additional negative training examples.

We then apply affine transformations to all the character images, resize them to 48×48, and compute HOG features. We analyzed three different variations [10] (13, 31 and 36-dimensional) of HOG. To efficiently train the classifier with a large set of training examples, we use an explicit feature map [22] and the χ^2 kernel. This feature map allows a significant reduction in classification time as compared to non-linear kernels like RBF.

Score computation. We divide the images into horizontal strips of height 30 pixels and spot characters from a set of character bounding boxes, as described in Section 2.2. The idea here is to find images where the characters of the vocabulary word have a high likelihood in a relatively small area. We set the truncation parameter $\tau = 0.2$ in (2) empirically, and retrieve an initial set of top-100 results with this score and re-rank them by introducing spatial ordering and positioning constraints.

3. Datasets

We evaluate our approach on three scene text (SVT, ICDAR 2011 and IIIT scene text retrieval) and two video (Sports-10K and TV series-1M) datasets. The number of images and queries used for these datasets are shown in Table 2.

Street view text [4] and ICDAR 2011 [17]. These two datasets were originally introduced for scene text localization and recognition. They contain 249 and 255 images respectively. We use all the unique ground truth words of these datasets as queries and perform text-to-image retrieval.

IIIT scene text retrieval dataset. The SVT and ICDAR 2011 datasets, in addition to being relatively small, contain many scene text words occurring only once. To analyze our text-to-image retrieval method in a more challenging setting, we introduce IIIT scene text retrieval (STR) dataset. For this, we collected data using Google image search with 50 query words such as Microsoft building, department, motel, police. We also added a large number of distractor images, i.e. images without any text, downloaded from Flickr into the dataset. Each image is then annotated manually to say if it contains a query text or not. This dataset contains 10K images in all, with 10-50 occurrences of each query word. It is intended for category retrieval (text appearing in different fonts or styles), instance retrieval (text imaged from a different view point), and retrieval in the presence of distractors (images without any text).

Video datasets. To analyze the scalability of our retrieval approach, we need a large dataset, where query words appear in many locations. In this context, we introduce two video datasets in this paper. The first one is from sports video clips, containing many advertisement signboards, and the second is from four popular TV series: Friends, Buffy, Mr. Bean, and Open All Hours. We refer to these two datasets as Sports-10K and TV series-1M respectively. The TV series-1M contains more than 1 million frames. Words such as *central*, *perk*, *pickles*, *news*, *SLW27R* (a car number) frequently appear in the TV series-1M dataset. All the image frames extracted from this dataset are manually annotated with the query text they may contain.

Datasets	**# queries**	**# images/frames**
SVT [17]	427	249
ICDAR [17]	538	255
IIIT STR	50	10K
Sports-10K	10	10K
TV series-1M	20	1M

Table 2. *Scene text datasets (SVT and ICDAR) contain only a few hundred images. We introduce an image (IIIT scene text retrieval) and two video (Sports-10K and TV series-1M) datasets to test the scalability of our proposed approach.*

Annotations are done by a team of three people for about 150 man-hours. We use 10 and 20 query words to demonstrate the retrieval performance on the Sports-10K and the TV series-1M datasets respectively. All our datasets are available on the project website.

4. Experimental Analysis

Given a text query our goal is to retrieve all images where it appears. We aim instance, i.e. text appearing in different view points, as well as category retrieval, i.e. text in different fonts and styles. In this section, we evaluate all the components of the proposed method to justify our choices.

4.1. Character classification results

We analyze the performance of one of the basic modules of our system on scene character datasets. Table 3 compares our character classification performance with recent works [8, 13, 24]. We observe that selecting training data and features appropriately improves the character classification accuracies significantly on the ICDAR-char [3], Chars74K [8], SVT-char [13] datasets. The reported accuracies are on the test set of the respective datasets. With respect to the computation time, linear SVM trained on 13-dimensional HOG (H-13) outperforms other HOG variants with only a minor reduction in performance. We use this combination in all our retrieval experiments.

Method	SVT	ICDAR	c74K	Time
FERNS [24]	-	52	47	-
RBF [13]	62	62	64	$30\mu s$
MKL+RBF [8]	-	-	57	$110\mu s$
H-36+AT+Linear	**69**	**73**	**68**	$20\mu s$
H-31+AT+Linear	64	73	67	$18\mu s$
H-13+AT+Linear	65	72	66	$\mathbf{8}\mu s$

Table 3. *A smart choice of features, training examples and classifier is key to better character classification. We enrich the training set by including negative examples, and many small affine transformed (AT) versions of the original training data from ICDAR and Chars74K(c74k). We then represent HOG features using an explicit feature map and train a linear SVM. Here H-36 is 36-dimensional HOG proposed in [7], while H-13 and H-31 are proposed in [10]. The time shown includes average time required per test sample for feature computation as well as classification. H-13 + AT + linear takes less than 50% time compared to other methods, with a minor reduction in accuracy, and hence used this for our character detection module. It is to be noted that [8] only uses 15 training samples per class.*

4.2. Retrieval results

We first evaluate our retrieval scheme on SVT, ICDAR 2011, and IIIT STR, one of our datasets. The retrieval performance is quantitatively evaluated using the well-known mean average precision (mAP) measure, which is the mean of the average precision for all the queries. The results are summarized in Table 4. We observe that the performance of our initial naive character spotting method is comparable to the baselines in Table 1. The re-ranking scheme improves the performance, and we achieve an mAP of 56.24% on SVT and 65.25% on ICDAR. Recall from Table 1 that the state-of-the-art localization and recognition based method only achieves an mAP of 23.32% on SVT. Reasonably high performance on IIIT STR, which contains instances (text in different viewpoints), categories (text in different fonts), and distractors (images without any text) shows that the proposed method is not only applicable to retrieve instances and categories of scene texts, but also robust to distractors.

We then evaluate the scalability of our proposed scheme on two large video datasets. We use precision computed using the top-n retrievals (denoted by P@n) as the performance measure. These results on video datasets are summarized in Table 5. The proposed re-ranking scheme achieves P@20 of 43.42% and 59.02% on Sports-10K and TV series-1M datasets respectively. Low resolution videos and fancy fonts appearing in advertisement boards make the Sports-10K dataset challenging, and thus the precision values are relatively low for this dataset.

Our indexing scheme allows us to retrieve images from a large dataset containing 1M images in about 3 seconds. The sliding window based character detection step and computation of index file are performed offline. They take around 9 seconds and 7 seconds per image.

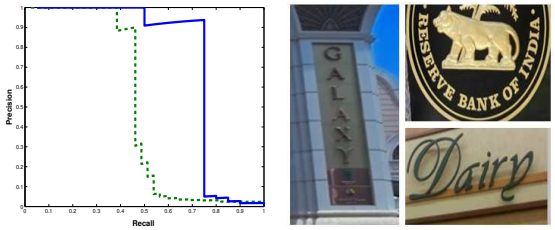
Dataset	Char. Spot.	RSO	RSP
SVT	17.31	46.12	**56.24**
ICDAR11	24.26	58.20	**65.25**
IIIT STR	22.11	36.34	**42.69**

Table 4. *Quantitative evaluation of text-to-image retrieval. We achieve a notable improvement in mAP with the proposed re-ranking schemes over baseline methods shown in Table 1. Another baseline we compare with uses character detections from [24] in combination with our spatial positioning re-ranking scheme, which achieves 52.12% mAP on SVT, over 4% lower than our result.*

Qualitative results of the proposed method are shown in Figure 4 for the query words *restaurant* on SVT, *motel* and *department* on IIIT STR. We retrieve all the occurrences of the query *restaurant* from SVT. The IIIT STR dataset contains 39 different occurrences of the word *motel*, with notable variations in font style, view point and illumination. Our top retrievals for this query are quite significant, for instance, the tenth retrieval, where the query word appears in a very different font. The query word *department* has 20 occurrences in the dataset. Few of these occurrences are on the same building. We observe that, overcoming the changes in the visual content, the relevant images are ranked high. Figure 5(a) shows precision-recall curves for two text queries: *department* and *motel* on IIIT STR. Our method achieves AP = 74.00 and 48.69 for these two queries respectively. Additional results are available on our project website. The method tends to fail in cases where almost all the characters in the word are not detected correctly or when the query text appears vertically. A few such cases are shown in Figure 5(b).

5. Conclusions

We have demonstrated text-to-image retrieval based on the textual content present in images and videos. The query-driven approach we propose outperforms localization and recognition pipeline based methods [9, 15]. We achieve a 30% improvement in mAP for text-to-image retrieval on SVT over previous methods. The benefits of this work over methods based on a localization-recognition pipeline [9, 15] are: (i) It does not require explicit localization of the word boundary in an image. (ii) It is query-driven, thus even in cases where the second or the third best predictions for a character bounding box are correct, it can retrieve the correct result. We showed that our method is robust, and scalable by analyzing it with three large image and video datasets.



(a) (b)

Figure 5. *(a) Precision-Recall curves for two queries on the IIIT scene text retrieval dataset. The blue (solid) and green (dotted) curves correspond to queries "department" and "motel" respectively. (b) A few failure cases are shown as cropped images, where our approach fails to retrieve these images for the text queries: Galaxy, India and Dairy. The main reasons for failure are: the violation of near horizontal assumption for scene texts (in case of Galaxy and India), or a stylish font (Dairy).*

Acknowledgements. This work is partly supported by MCIT, New Delhi. Anand Mishra is supported by the Microsoft Research India PhD fellowship 2012 award. Karteek Alahari is partly supported by the Quaero programme funded by the OSEO.

References

[1] http://textspotter.org.

[2] http://www.eng.tau.ac.il/∼talib/RBNR.html.

[3] http://algoval.essex.ac.uk/icdar/.

[4] http://vision.ucsd.edu/∼kai/svt/.

[5] D. Chen, J.-M. Odobez, and H. Bourlard. Text detection, recognition in images and video frames. *Pattern Recognition*, 2004.

[6] X. Chen and A. L. Yuille. Detecting and reading text in natural scenes. In *CVPR*, 2004.

[7] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

[8] T. E. de Campos, B. R. Babu, and M. Varma. Character recognition in natural images. In *VISAPP*, 2009.

[9] B. Epshtein, E. Ofek, and Y. Wexler. Detecting text in natural scenes with stroke width transform. In *CVPR*, 2010.

[10] P. F. Felzenszwalb, R. B. Girshick, D. A. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *TPAMI*, 2010.

[11] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[12] A. Mishra, K. Alahari, and C. V. Jawahar. Scene text recognition using higher order langauge priors. In *BMVC*, 2012.

Dataset	Char. Spot.		RSO		RSP	
	P@10	P@20	P@10	P@20	P@10	P@20
Sports	26.21	24.26	39.11	38.32	44.82	43.42
TV series	40.22	39.20	58.15	57.21	59.28	59.02

Table 5. *Quantitative analysis of retrieval results on video datasets. We choose 10 and 20 query words for Sports-10K and TV series-1M respectively. We use top-n retrieval to compute precision at n (denoted by P@n).*

[13] A. Mishra, K. Alahari, and C. V. Jawahar. Top-down and bottom-up cues for scene text recognition. In *CVPR*, 2012.

[14] M. Mozer, M. I. Jordan, and T. Petsche. Improving the accuracy and speed of support vector machines. In *NIPS*, 1997.

[15] L. Neumann and J. Matas. Real-time scene text localization and recognition. In *CVPR*, 2012.

[16] J. C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in Large Margin Classifiers*. MIT Press, 1999.

[17] A. Shahab, F. Shafait, and A. Dengel. ICDAR 2011 robust reading competition challenge 2: Reading text in scene images. In *ICDAR*, 2011.

[18] C. Shi. Scene text recognition using part-based tree-structured character detections. In *CVPR*, 2013.

[19] P. Shivakumara, T. Q. Phan, and C. L. Tan. A laplacian approach to multi-oriented text detection in video. *IEEE TPAMI*, 2011.

[20] P. Simard, B. Victorri, Y. LeCun, and J. S. Denker. Tangent prop - a formalism for specifying selected invariances in an adaptive network. In *NIPS*, 1991.

[21] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *ICCV*, 2003.

[22] A. Vedaldi and A. Zisserman. Efficient additive kernels via explicit feature maps. *TPAMI*, 2012.

[23] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. *CVPR*, 2001.

[24] K. Wang, B. Babenko, and S. Belongie. End-to-end scene text recognition. In *ICCV*, 2011.

[25] K. Wang and S. Belongie. Word spotting in the wild. In *ECCV*, 2010.

(a) Text query: *restaurant*

(b) Text query: *motel*

(c) Text query: *department*

Figure 4. *Text query example: Top-10 retrievals of our method on SVT and IIIT STR are shown. (a) Text query: "restaurant". There are in all 8 occurrences of this query in the SVT dataset. The proposed scheme retrieves them all. The ninth and the tenth results contain many characters from the query like R, E, S, T, A, N. (b) Text query: "motel". There are in all 39 occurrences of query in the IIIT STR dataset, with large variations in fonts, e.g. the first and the tenth retrievals. A failure case of our approach is when a highly similar word (hotel in this case) is well-ranked. (c) Text query: "department". The top retrievals for this query are significant. The fourth, sixth and seventh results are images of the same building with the query word appearing in different views. These results support our claim of instance as well as category retrieval.*