Video Text Detection and Recognition: Dataset and Benchmark

Phuc Xuan Nguyen
Department of Computer Science and Engineering
University of California, San Diego

Kai Wang
Google Inc.

Serge Belongie
Cornell NYC Tech
Cornell University

Abstract

This paper focuses on the problem of text detection and recognition in videos. Even though text detection and recognition in images has seen much progress in recent years, relatively little work has been done to extend these solutions to the video domain. In this work, we extend an existing end-to-end solution for text recognition in natural images to video. We explore a variety of methods for training local character models and explore methods to capitalize on the temporal redundancy of text in video. We present detection performance using the Video Analysis and Content Extraction (VACE) benchmarking framework on the ICDAR 2013 Robust Reading Challenge 3 video dataset and on a new video text dataset. We also propose a new performance metric based on precision-recall curves to measure the performance of text recognition in videos. Using this metric, we provide early video text recognition results on the above mentioned datasets.

Figure 1: Example frames in the YouTube Video Text dataset. The green bounding boxes indicate the annotated locations of text regions. Images in the top row are examples of scene text while the bottom rows show examples of overlay text.

1. Introduction

Text detection and recognition in unconstrained environments is a challenging computer vision problem. Such functionality can play valuable role in numerous real-world applications, ranging from video indexing, assistive technology for the visually impaired, automatic localization for businesses, and robotic navigation. In recent years, the problem of scene text detection and recognition in natural images has received increasing attentions from the computer vision community [1, 2, 21, 20, 18, 17, 5]. As a result, the domain has enjoyed significant advances on an increasing number of datasets of public scene text benchmarks [12, 4, 22, 21, 13, 10].

Even though the amount of video data available is rapidly increasing due to extensive use of camera phones and wearable cameras (e.g. Google Glass, GoPro and video-sharing websites such as YouTube), relatively little work has been done on extending text reading solutions to the video domain. The ICDAR 2013 Robust Reading Challenge 3 [7] introduced the first public video dataset for the purpose of providing a common benchmark for video text detection. We refer to this dataset as ICDAR-VIDEO for the remainder of the paper. Apart from the ABBYY baseline provided by the organizers, TextSpotter [15, 16] was the only participant in this challenge. To the best of our knowledge, both of these systems were originally designed for the task of text detection and recognition in images and were tweaked slightly to fit the competition format. Furthermore, while the ICDAR-VIDEO dataset provided both the position and string annotations for the words within each frame, only detection results were reported.

In this work, we focus on the problems of detection and recognition of text in videos. Figure 1 shows frames with text content in example videos. Similar to the image domain counterparts [18, 20], this paper focuses on the problem of text detection and recognition in videos given a list of words (i.e., a lexicon). In many applications, the lexicon availability assumption is reasonable. Consider the problem of assisting a blind person to shop at a grocery store: in this case the lexicon would be a list of products in the store or within a given aisle.

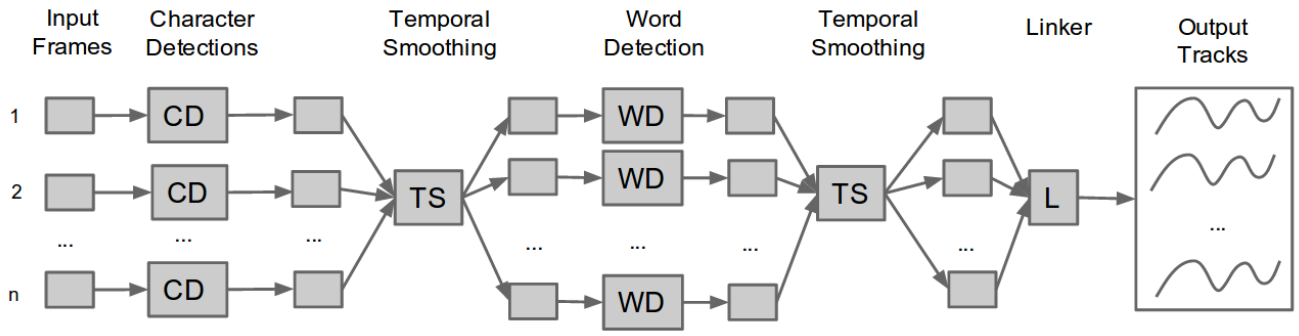


Figure 2: An overview of our system. Starting from the left, raw input frames are fed independently to the character detection module. This module returns a list of bounding boxes describing locations, character classes, and detection scores. These character bounding boxes are temporally smoothed to remove false positives. Next, we perform word detection using Pictorial Structures, rescore each word using global features, and perform non-maximum suppression. Word detections in each frame are then passed through another round of temporal smoothing to remove false positive words. Finally, we link the per-frame detections into tracks.

Contributions Our contributions are three-fold. (1) We extend Wang's solution [20] to the video domain by employing a different character detection method and including multiple modules that exploit temporal redundancy to improve the performance of the system. (2) We propose a precision/recall metric suited to the task of text recognition in video. (3) Finally, we introduce a new video text dataset.

2. System Overview

Our approach is based on an extension of Wang's solution [20] as the components of this pipeline are simple yet effective. The source code is publicly available[1]. Figure 2 shows the complete overview of our video text detection and recognition system. We describe each module in detail in the following sections.

2.1. Character Detection

The first step in our pipeline is to detect characters given an image frame. We perform multi-scale character detection via scanning-window templates trained with linear SVM and Histogram of Oriented Gradients (HOG) [3] features. For each character, we train a binary classifier with five rounds of hard negative mining. The initial round of training uses the character samples of the target class as positive training data and the other classes as negative training data. In subsequent rounds, we mine hard negative patches by running the previously trained model on images from the Flickr dataset [6] and add top-scoring detections to the negative sets. To ensure the quality of the negative pool, we manually examined each image in the Flickr dataset and removed ones containing text.

[1] http://vision.ucsd.edu/ kai/grocr/

Pre-processing Character Samples Character samples from public text datasets sometimes do not have tight bounding boxes and, in some cases, have the wrong labels. We manually go through each character image, remove samples with wrong labels, and recrop the images for a tight fit with the characters. Enforcing a tight bounding box gives the training data more consistent alignment and, hence, produces better templates.

Mixture Models We observe that there is great intra-class variability within each character class. Different "prototypes" of a letter can have very different appearances. Zhu et al. [24] suggest that we can grow the model complexity by adding mixture models to capture the "sub-category" structure of the data. The first step of training mixture models involves clustering the existing data into K different clusters. A classifier is then trained separately for each cluster. This technique often requires a large amount of training data as each cluster requires sufficient positive examples to train a good model. Fortunately, in recent years, many text datasets with character-level annotations have been published [4, 12, 13, 21]. Combining the character samples from these datasets gives a large set for training mixture models. For each character, we split the samples into 10 clusters using K-means. Similar to Zhu's [24] experiments, we append the aspect ratio to the HOG features, and use PCA to reduce the feature dimensionality prior to clustering.

2.2. Pictorial Structures and Word Rescoring

We follow Wang's Pictorial Structures [20] approach for constructing the word detections from character bounding boxes. We also use the same word rescoring technique,

but construct the training set differently. To construct the training set for the rescoring SVM, we randomly select 100 frames that contain text from each video in the training set. We run the system on this set and label each returned word positive if matched with a ground truth and negative otherwise.

2.3. Exploiting Temporal Redundancy

Lienhart [11] suggests in his survey that one can exploit temporal redundancy in video to remove false positives in individual frames and recover missed detections. In the following sections, we describe three modules—the temporal smoothing module, the linker, and the post-processing module—that leverage temporal properties to improve the performance further.

Temporal Smoothing Due to the local nature of scanning-window approaches, false positives are often pervasive, especially in a system that favors recall over precision. This problem was also observed in [14]. Removing character false positives at this step is crucial as it reduces the search space for the word detection step.

Given a bounding box detection at a given frame, we compute the overlap ratio between it and all detections in the preceeding and following N frames. If a sufficient amount of the neighboring frames contains detections that satisfy the overlap condition with the target frame, we keep the detection in the current frame. Otherwise we discard it as a false positive. More formally, let b_i^f be the i-th bounding box in frame f, $|b_i^f|$ be the number of pixels in this region, D_t be the number of detections in time t and α be a real number between 0 and 1, and define

$$I(b_i^f, t) = \begin{cases} 1 & \bigvee_{j=1}^{D_t} \left(\frac{|b_i^f \bigcap b_j^{f-t}|}{|b_i^f \bigcup b_j^{f-t}|} > \alpha \right) \\ 0 & otherwise \end{cases} \quad (1)$$

Next, we define a function H that takes an i-th bounding box at frame f, b_i^f, and returns 1 for a true positive or 0 for a false positive. The function is defined as

$$H(b_i^f) = \left(\sum_{j=1}^{N} I(b_i^f, j) \geq \beta N \right) \vee \left(\sum_{k=1}^{N} I(b_i^f, -k) \geq \beta N \right) \quad (2)$$

where β is a real number between 0 and 1.

This module has three hyperparameters: (1) N controls the number of frames to search forward and backward, (2) α specifies the overlap ratio condition, and (3) β controls the fraction of neighboring frames needed to decide whether the current detection is a true positive. We perform a grid search on a validation set to tune these parameters.

Linker When the results are temporally smoothed to remove false positives, we proceed to link the per-frame detections into tracks. For each detection in frame t, we search backwards in a buffer of 10 prior frames. We consider the following features:

- the overlap ratio between the current bounding box and the candidate in the prior frame,
- the edit distance between the word and the candidate word,
- the temporal distance between the current detection and the candidate, measured in number of frames.

We train a linear classifier to determine whether the bounding box detection in previous frames is a match for the current bounding box. We only allow one match per previous frame. If there is more than one, we choose the one with the highest score. We assign the current bounding box the identifier that is present in the majority of matched frames.

To establish a training set, we consider each detection in each frame and the set of candidate bounding boxes in the previous 10 frames. We label the bounding boxes with the same track identifier as positives and the rest as negatives. We feed these labels and the computed features to a standard SVM package[2].

Post Processing Due to motion blur and other artifacts, detections in some frames within a track might be missing, causing temporal fragmentations. To mitigate this effect, we linearly interpolate both the word scores and the word bounding boxes between the detected frames to recover missing detections. Finally, we remove all tracks with less than 10 frames.

3. Experiments

3.1. Dataset

ICDAR-VIDEO In the ICDAR 2013 Robust Reading Competition Challenge 3 [7], a new video dataset was presented in an effort to address the problem of text detection in videos. The dataset consists of 28 videos in total: 13 videos for the training set and 15 for the test set. The scenarios in the videos include walking outdoor, shopping in grocery stores, driving and searching for directions within a building. Each video is around 10 seconds to 1 minute long capturing scenes from real-life situations using different types of cameras.

To construct the lexicon for a video, we extract all ground truth words in the dataset to form a vocabulary. We then assign a lexicon for each video by taking a union of its

[2] http://www.csie.ntu.edu.tw/~cjlin/liblinear/

ground truth words and a random subset of 500 "distractor" words sampled from the vocabulary.

In the ICDAR-VIDEO dataset, the annotators assigned each word a quality, which can be "LOW", "MEDIUM", or "HIGH". During the competition, the "LOW" quality was not considered. More specifically, misses on the low quality words did not penalize the system and detections of the low quality words did not improve the score. For the sake of simplicity of the evaluation framework, we decided to include words with all qualities in our evaluation.

YouTube Video Text We introduce YouTube Video Text[3] (YVT) dataset harvested from YouTube. The text content in the dataset can be divided into two categories, overlay text (e.g., captions, songs title, logos) and scene text (e.g. street signs, business signs, words on shirt). Figure 1 shows examples of text content in the YVT dataset.

We downloaded a large number of videos by crawling YouTube. We split each downloaded video into 15-second segments and created Amazon Mechanical Turk tasks to filter out segments without text content. For each segment that contained text, we annotated the locations of text regions using VATIC [19]. We instructed the annotators to draw tight bounding boxes only for readable English words. Next, we asked the annotators to go through each frame in the ground truth track and type the word contained in this bounding box.

The dataset contains a total of 30 videos, 15 in the training set and 15 in the testing set. Each video has HD 720p quality, 30 frames per second, and 15-second duration. We constructed the lexicon in the same way as for the ICDAR-VIDEO dataset.

3.2. Performance Metrics

There are many evaluation frameworks proposed for multiple object tracking systems [8, 9]. In this paper, we use ATA metrics from the VACE framework [8] to measure the performance of the detection systems. We also propose a video precision-recall metric to measure the recognition performance.

VACE Metrics The Average Tracking Accuracy (ATA) in the VACE framework provides a spatio-temporal detection measure that penalizes fragmentations both in temporal and spatial dimensions.

For every frame t, a text tracking system outputs a set of bounding box detections $\{d_1^t, \ldots, d_n^t\}$. We denote the ground truth at frame t as $\{g_1^t, \ldots, g_n^t\}$. Using their unique identifiers, we can group these per-frame detections and ground truth into separate tracks, $\{D_1, \ldots, D_r\}$ and $\{G_1, \ldots, G_q\}$.

[3] http://vision.ucsd.edu/datasetsAll

The first step is to establish a one-to-one mapping between the ground truth and the system output tracks. We follow the mapping process described in [8].

Once the mapping between detection tracks and ground truth is established, the Sequence Track Detection Accuracy (STDA) is defined as

$$STDA = \sum_{i=1}^{N_M} \frac{\sum_t m(G_i^t, D_i^t)}{N_{G_i \cup D_i \neq 0}} \tag{3}$$

where N_M is the number of correspondences in the mapping M, $N_{G_i \cup D_i \neq 0}$ is the number of frames where either G_i or D_i exist and $m(G_i^t, D_i^t)$ takes the value 1 if overlap$(G_i^t, D_i^t) > 0.5$ and 0 otherwise.

The Average Tracking Accuracy is then calculated as,

$$ATA = \frac{STDA}{\left[\frac{N_G + N_D}{2}\right]} \tag{4}$$

where N_G and N_D are the number of ground truth tracks and the the number of detection tracks.

Video Precision and Recall Precision-recall curves are used widely in the literature of text recognition in the image domain. This metric demonstrates the tradeoff of the system and facilitates the selection of operating points. We propose to extend this to a sequence-level precision and recall performance metric. In particular, ground truths are matched at sequence-level similar to the computation for ATA. We, however, place additional restrictions. We define $m(G_i^t, D_i^t)$ to be 1 if the overlap ratio is greater than 0.5 and the words in frame t match (ignoring cases).

A ground truth track G_i and a detection track D_i are considered a match if and only if

$$overlap = \frac{\sum_t m(G_i^t, D_i^t)}{N_{G_i \cup D_i \neq 0}} > 0.5 \tag{5}$$

The threshold at 50% is arbitrary but reasonable. This extra restriction requires that the detected track must sufficiently fit the ground truth track at least half of the time. Once the one-to-one mapping is established, the matched detections are considered true positives, the unmatched detection tracks are false positives and the unmatched ground truth tracks are false negatives. We then use the conventional definitions of precision, and recall to evaluate the performance of the detections.

3.3. Character Detection

We begin with the evaluations of 4 different character detection models. Wang et al. [20] synthesizes characters with different fonts, backgrounds, rotation angles, and noises. (1) We train character models (SYNTH) from these synthetic data. (2) The second set of models is

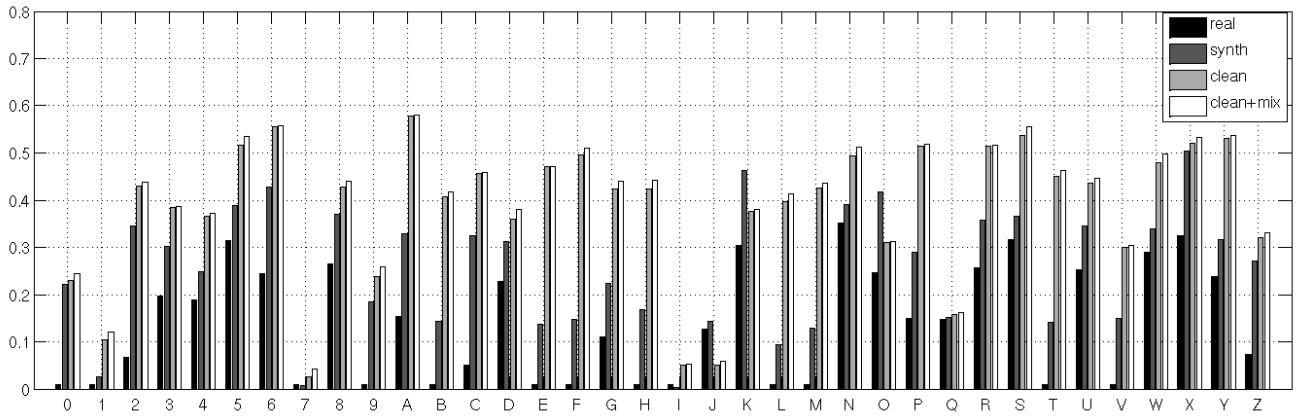


Figure 3: Character detection performance (F2-score) comparing different methods.

trained with unprocessed real characters obtained from public datasets (REAL). (3) The third set is trained with preprocessed data as described in section 2.1 (CLEAN). Finally, (4) we train mixture models by clustering preprocessed data (CLEAN+MIX). Real characters are collected from the SVT-CHAR [21, 14], Weinman's dataset [23], and the English characters of Chars74K [4]. For all character detection experiments, we benchmark the character results on the ICDAR 2003 Robust Reading [12] test set.

As character detection is an early step in the pipeline, we favor recall over precision as it is often easier to remove false positives than to recover false negatives. We use F2-score, a modified version of the F-score, defined as $F_\beta = (1+\beta^2).\frac{precision \times recall}{\beta^2 \times precision + recall}$, with $\beta = 2$.

Figure 3 shows the F2-score for each character detection methods. From this plot, we observe the following. (1) Models trained from "dirty" data (REAL) produce the worst results. This could be explained by the misalignments of the data and the labeling errors. (2) "Clean" real training data outperforms synthetic data. Even though synthetic data is different across fonts, angles, noises, and blurriness, this variation is not enough to describe the entire appearance space of real-scene characters. (3) In general, mixture models yield small performance improvements (1-3%).

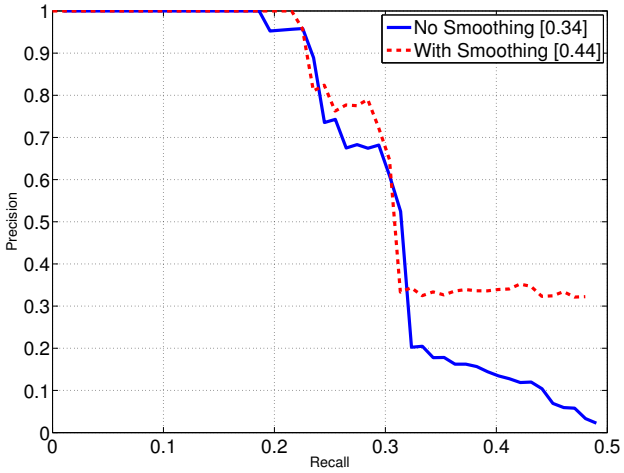


Figure 4: Character detection performance (F2-score) comparing the bounding box results with (red line) and without (blue line) temporal smoothing. The F2-score is in square bracket next to the method name.

3.4. Temporal Smoothing

We also evaluate the effectiveness of temporal smoothing in the task of removing false positives from the character detection step. Currently there is no public video text dataset that has character-level annotations. Since producing a video dataset with this level of annotation is expensive, we choose to evaluate the method on a smaller scale. We annotated a small set of consecutive frames from the ICDAR-VIDEO dataset at character-level. We selected 30 consecutive frames from 6 videos and annotated the bounding box locations for the character "A". Despite its relatively small size, this set makes possible a preliminary experiment to demonstrate the effectiveness of temporal smoothing at the character level. Figure 4 shows the precision and recall performance with and without temporal smoothing. This figures shows the effectiveness of temporal smoothing, especially at lower thresholds. This allows the character detection modules to operate at a lower threshold and achieve high recall without introducing an undue amount of false positives.

3.5. Word Detection and Recognition

Our main experiment consists of evaluating end-to-end word detection and recognition for both ICDAR-VIDEO and YVT datasets.

ICDAR-VIDEO Evaluation The first two pipelines we consider are ABBYY and TextSpotter. Even though the actual implementations of these methods are not published, we obtained the detection results by analyzing the Javascript structure of the ICDAR 2013 Robust Reading Competition website[4].

Next, we apply Wang's original implementation (PLEX) on each frame and group per-frame detections into tracks in a manner that similar to that of ABBYY baseline. More specifically, after obtaining the word detections for each frame, each detected word is assigned an identifier of a previously detected word with the best overlap (at least 0.5) in the buffer of 7 previous frames. Words are removed from detection unless there is a matching word detected in the immediate previous frame.

We also run PLEX with temporal enhancements, PLEX+T. This method uses temporal smoothing after the character detection and word detection step, links the per-frame detections using a trained linker and uses the post-processing as described in section 2.3.

Finally, we apply detection pipeline (DR) on every frame using ABBYY heuristic to connect the frames. This pipeline differs from PLEX at the character detection step. Instead of using models trained from synthetic data, DR uses mixtures models trained from processed real data. The last pipeline is DR+T where we use the temporal techniques to improve the performance of DR. Since PLEX and DR output a score for each word in every frame, we can compute a track score by averaging the scores of all words in the track.

Since ABBYY baseline and TextSpotter detections are produced without a lexicon, the performance results are provided mainly as references as opposed to strict comparisons.

Figure 6a shows the box plots for the ATA scores for each method. Figure 5a shows the video precision/recall curves. We generate the video precision-recall curves for PLEX, PLEX+T, DR, DR+T by varying a threshold on the track score. From these figures, we observe the following: (1) better local models lead to better detection and recognition performance as DR only differs from PLEX at the character detection step, however, final results differs significantly. (2) Temporal smoothing significantly improves the performance. With a relatively large lexicon, false positives are unavoidable. Furthermore, text in frames that are affected by motion blur and video artifacts is very hard to read (even for humans). Temporal smoothing and linear interpolation help to mitigate these effects.

[4]http://dag.cvc.uab.es/icdar2013competition/?ch=3

YVT Evaluation For the YVT dataset, we can only compare the performances of PLEX and DR because the original implementations of TextSpotter and ABBYY baseline are not publicly available. Figures 6b and 5b show the ATA metrics and the video precision/recall curves for the YVT dataset. Even though, DR still outperforms PLEX, the gap between PLEX and DR is smaller than in the ICDAR-VIDEO dataset. One possible explanation is the presence of overlay text. Overlay text often appears with standard fonts, no occlusion, and has a high level of contrast with the backgrounds. These appearance characteristics resemble the natures of synthetic training data. Again temporal smoothing improves the performance dramatically for both methods. Comparing performance on the YVT dataset and the ICDAR-VIDEO dataset reveals the relative difficulties between the two. This is expected as YVT's videos are of HD720p quality and contain overlay text.

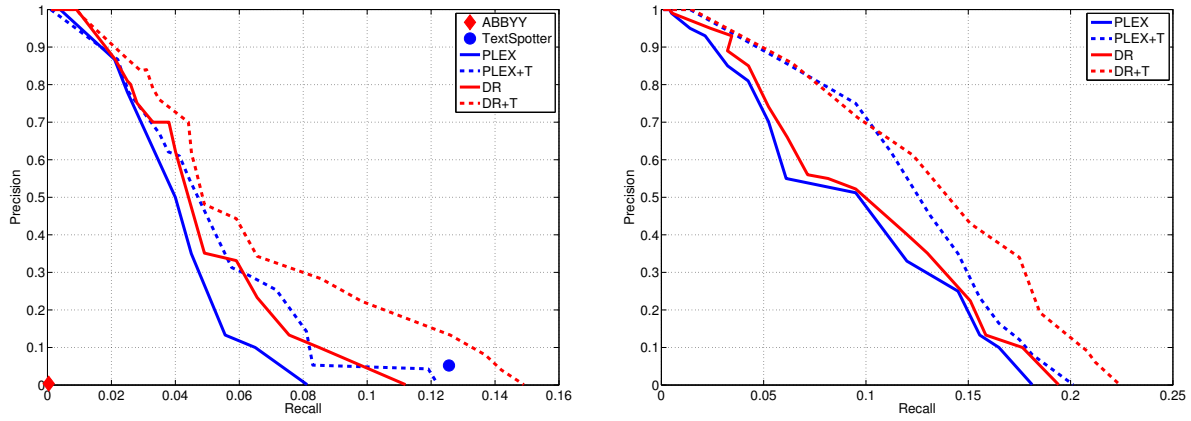
Figure 7 shows qualitative results of DR+T on example frames from both datasets. This figure offers insights into the successes and failures of the system. Failure cases are often caused by perspective distortions and unseen, challenging fonts. Phan et al. [18] has recently reported promising results in the problem of recognizing scene text with perspective distortions.

4. Conclusion

In this paper, we extend an end-to-end text detection and recognition solution to the video domain. Our work highlights the importance of local character detection models to the system as a whole. We also show the effectiveness of exploiting temporal redundancy to remove false positives. We propose a video precision and recall metric for benchmarking text recognition in video. By performing different detection and recognition experiments, we reveal the current state of text detection and recognition in video. Clearly, there is plenty of room for improvement in performance. In Figure 5, the best observed sequence recalls are around 15% and 23% for the ICDAR-VIDEO and the YVT datasets respectively. This means that less than one out of every four tracks is recognized correctly. We hope our datasets and performance results will serve as a baseline for future studies in text detection and recognition in the video domain.

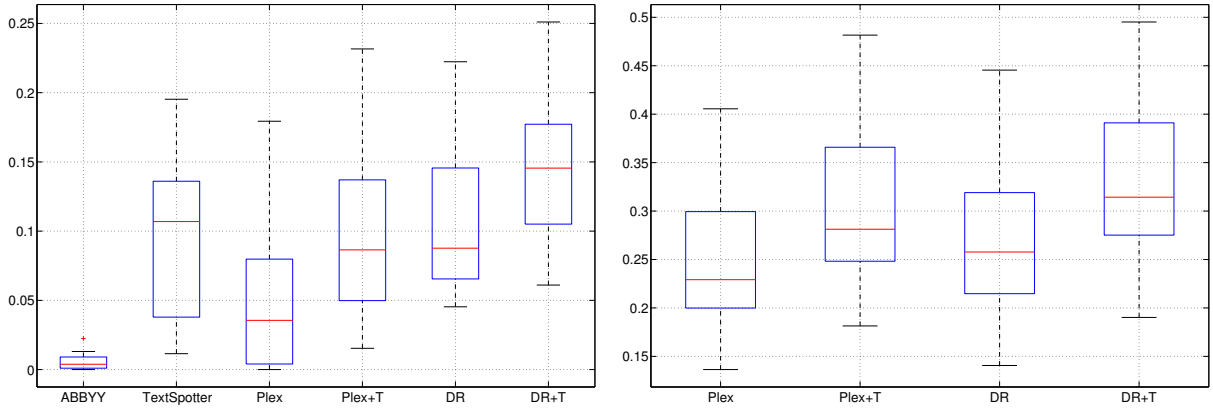
5. Acknowledgement

We thank Seung-Hoon Han and Alessandro Bissacco for valuable feedbacks during the project. Additional support was provided by Google Focused Research Award.



(a) ICDAR-VIDEO (b) YVT

Figure 5: Video precision-recall performance of the ABBYY baseline, TextSpotter, PLEX, and our detection-from-recognition pipeline (DR). Methods with temporal enhancements are denoted with +T.



(a) ICDAR-VIDEO (b) YVT

Figure 6: ATA metrics for the baseline methods and our detection-from-recognition pipeline (DR), see text for details.

References

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

[2] A. Coates, B. Carpenter, C. Case, S. Satheesh, B. Suresh, T. Wang, D. J. Wu, and A. Y. Ng. Text detection and character recognition in scene images with unsupervised feature learning. In *ICDAR*, 2011.

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

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

[5] W. Huang, Z. Lin, J. Yang, and J. Wang. Text localization in natural images using stroke feature transform and text covariance descriptors. In *ICCV*, 2013.

[6] M. J. Huiskes and M. S. Lew. The MIR Flickr retrieval evaluation. In *MIR*, 2008.

[7] D. Karatzas, F. Shafait, S. Uchida, M. Iwamura, S. R. Mestre, J. Mas, D. F. Mota, J. A. Almazan, and L. P. de las Heras. Icdar 2013 robust reading competition. In *ICDAR*, 2013.

[8] R. Kasturi, D. Goldgof, P. Soundararajan, V. Manohar, J. Garofolo, R. Bowers, M. Boonstra, V. Korzhova, and J. Zhang. Framework for performance evaluation of face, text, and vehicle detection and tracking in video: Data, metrics, and protocol. *PAMI*, 2009.

[9] B. Keni and S. Rainer. Evaluating multiple object tracking performance: the clear mot metrics. *EURASIP JIVP*, 2008.

[10] S. Lee, M. S. Cho, K. Jung, and J. H. Kim. Scene text extraction with edge constraint and text collinearity. In *ICPR*, 2010.

Figure 7: Selected results for the DR+T method. The detections are shown in green while the ground truth is shown in red.

[11] R. Lienhart. Video OCR: a survey and practitioners guide. In *Video mining*. 2003.

[12] S. M. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, and R. Young. ICDAR 2003 robust reading competitions. In *ICDAR*, 2003.

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

[14] A. Mishra, K. Alahari, C. Jawahar, et al. Scene text recognition using higher order language priors. In *BMVC*, 2012.

[15] L. Neumann and J. Matas. A method for text localization and recognition in real-world images. In *ACCV*, 2010.

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

[17] L. Neumann and J. Matas. Scene text localization and recognition with oriented stroke detection. In *ICCV*, 2013.

[18] T. Q. Phan, P. Shivakumara, S. Tian, and C. L. Tan. Recognizing text with perspective distortion in natural scenes. In *ICCV*, 2013.

[19] C. Vondrick, D. Patterson, and D. Ramanan. Efficiently scaling up crowdsourced video annotation. *IJCV*, 2012.

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

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

[22] J. J. Weinman and E. Learned-Miller. Improving recognition of novel input with similarity. In *CVPR*, 2006.

[23] J. J. Weinman, E. Learned-Miller, and A. R. Hanson. Scene text recognition using similarity and a lexicon with sparse belief propagation. *PAMI*, 2009.

[24] X. Zhu, C. Vondrick, D. Ramanan, and C. Fowlkes. Do we need more training data or better models for object detection?. In *BMVC*, 2012.