A Novel Video Caption Detection Approach Using Multi-Frame Integration

Rongrong Wang, Wanjun Jin, Lide Wu
Dept. of Computer Science and Engineering, Fudan University, Shanghai, P.R.China (200433)
{rrwang, jwj, ldwu}@fudan.edu.cn

Abstract
Captions in videos often play an important role in video information indexing and retrieval. In this paper, we present a novel video caption detection approach. We first apply a new Multiple Frames Integration (MFI) method to minimize the variation of the background of the image. A time-based minimum (or maximum) pixel value search is employed and Sobel edge map is used to determine the mode of search. Then block-based text detection is performed, i.e. a small window is used to scan the image and classified as text or non-text, using Sobel edges as features. We use a two-level pyramid to detect various text sizes. Finally, we present a new iterative text line decomposition method and accurate text bounding boxes are extracted from candidate text areas. Experimental result shows that the proposed approach achieves a high precision and recall.

1. Introduction
Text detection in videos is often used for video information indexing and retrieval, since text can provide concise and direct description of the objects or stories presented in videos, especially in news videos. It is helpful for people to get more information about the video.

In content-based information retrieval, video text detection has attracted much attention of many researchers. Current video text detection approaches can be classified into two categories. One category is detecting text regions in individual frames independently [1]. The other category is utilizing the temporality of video sequences [2,3]. The first category can be further divided into two kinds: connected component-based methods [4], which may have difficulties when texts are embedded in complex background or touch other objects, and texture-analysis-based methods [1-3], which can be very sensitive to font sizes and styles and also accurate boundaries of text areas are hard to be found.

Existing methods do solve some problems to a certain extent, but not perfectly. One of the key difficulties in text detection comes from the complexity of background. Multi-Frame Integration techniques such as frame averaging method [2] and the minimum pixel search [3] have been employed to reduce the influence of the complex background. The common characteristic of these techniques is: they all apply text detection on several consecutive frames first and use MFI to verify text areas or simplify the backgrounds of text boxes afterwards.

There are some deficiencies in these techniques: first, although the whole backgrounds of consecutive frames may vary widely, MFI may be not so effective to reduce the background complexity of text boxes, since text boxes are always too small; second, it is difficult to verify text areas with MFI because lots of false alarms always appear in several consecutive frames just like text areas; third, all these techniques need to apply text detection on all consecutive frames containing the same caption and it is time-consuming.

In this paper, we present a novel video caption detection approach using Multi-Frame Integration. Different from previous MFI methods, we perform MFI before text detection process, which can reduce the complexity of the whole background of the frame and make text detection much easier. Although our system uses texture features, it classifies text or non-text based on block. So it is not sensitive to languages and font sizes. Also our iterative text line decomposition method can find much more accurate text bounding boxes. Specially, we only focus on the stable superimposed text.

The rest of this paper is organized as follows. A detailed description of text detection algorithm is given in Section 2. Experimental results are shown in Section 3. Section 4 presents our conclusion and future work.

2. Text Detection Algorithm
Figure 1 shows the flow chart of our caption detection approach. Video streams are first decompressed into individual frames and grayscale frame images are taken as input. MFI is first applied and finally accurate text bounding boxes are extracted, as to be shown in this section.

2.1. Multi-Frame Integration
In real-life videos, one of the key difficulties in caption detection is due to widely varying complex backgrounds, which results in lots of false alarms. So although most of the text detection methods achieve a high recall, they always have a low precision.
Figure 1. Flowchart of the proposed method.

Existing techniques [2, 3] to resolve this problem are all based on the fact that for better understanding, the same caption always appears in several consecutive frames and at the same time complex backgrounds usually have movement, while the position of video captions is relatively stable across frames. Our system is also based on this fact. But different from previous techniques, we apply MFI before text detection process.

It can be found from real-life videos that captions always appear at least 2 seconds, i.e. in at least 60 consecutive frames. For the understanding of the video content, it is not essential to detect each caption in every frame. Shot is a clip that is recorded continuously without breaks, and usually is regarded as the basal unit in video information processing. So we first segment the video into shots using our Shot Boundary Detection System [6]. Then our caption detection system is performed every 30 frames within each shot. Before applying text detection on one frame, MFI is applied on this frame and its consecutive 29 frames. We select 30 frames for detecting and integrating because it guarantees that the same caption exists in consecutive 30 integrated frames at least once.

We exploit a MFI technique by using time-based minimum (or maximum) pixel value search, which is similar to the method in [3]. We do not choose the frame averaging method, because the grayscales of several consecutive frame images sometimes change widely.

For a frame cluster $C_i$ (from frame $i$ to frame $i+29$), we generate two images as follows:

\[
\text{MinImage}_i(x, y) = \min_{j \in [i:i+29]} (p_j(x, y))
\]

\[
\text{MaxImage}_i(x, y) = \max_{j \in [i:i+29]} (p_j(x, y))
\]

where $p(x, y)$ indicates the grayscale of the pixel of position $(x, y)$, and $j$ is the frame number.

In [3], MFI is applied as time-based minimum or maximum pixel value search based on white characters or black characters, for the objective of their MFI is to improve the quality of text boxes. Different with it, the objective of our MFI is to improve the quality of the whole image and reduce the difficulty of text detection. Our attention is much focused on the variation of areas outside text areas. So we do not make a choice between minimum pixel search and maximum pixel search based on the grayscale color of characters. In fact we can not know whether characters are white or black since our MFI is a pre-processing step. We will use the Sobel edge map to help us determine the mode of pixel search (see Section 2.2).

Figure 2 (a)(b) show two images generated by MFI. The next step will determine which image is chosen as the integrated image.

2.2. Edge Detection

In Section 2.1, for every frame cluster $C_i$, we generate two images: MinImage and MaxImage. Then we will employ Sobel edge detection on both MinImage and MaxImage respectively.
MaxImage, and judge whose Sobel edge points are much fewer. Finally we apply text block classification on the image with fewer edge points.

We choose four directional Sobel edge operators to extract the edges of an image. Figure 3 shows the Sobel operator. If the maximal gradient of current point is greater than the threshold, it is the edge point then. Here we set Sobel threshold as 110 experimentally.

With the help of Sobel edge map, we determine the mode of the time-based pixel value search. Figure 2 shows an example of MFI procedure. In this example, we choose MaxImage as our integrated image at last.

Although we only want to judge which image of both has more uniform color, the Sobel edge map will continue to be used in the classification step later (Section 2.3).

\begin{figure}[h]
\centering
\begin{tabular}{cccc}
-1 & -1 & 0 & 1 \\
0 & 0 & 2 & -2 \\
1 & 2 & 1 & 0 \\
\end{tabular}
\caption{Sobel operator. (a)(b)(c)(d) indicate Horizontal, Vertical, Left diagonal and Right diagonal respectively.}
\end{figure}

2.3. Text Block Classification

In the step of text classification, we use a MxN (20x10 pixels in our system) window to scan the integrated image and classify each window as text or non-text. The reason why we choose M>N not M=N or M<N is that video captions always align horizontally. If the edge density of the window is larger than a predetermined threshold, it will be classified as text; otherwise, it will be classified as non-text. In our experiment, the threshold is assigned to 0.70 empirically.

We do not move the window at 1 pixel step, because it is time-consuming and not essential either. Consider the trade-off between precision and speed, we move the window 10 pixels horizontally and 5 pixels vertically at a time.

If a single window is classified as text, all the pixels in this window are labeled as text and remain their original grayscale values of integrated image. The pixels in the window classified as non-text are set to 0 (black). Thus we get a label map with candidate text areas (see Figure 4 (c)).

The next task is to get the exact position of the text lines from the candidate text areas in the label map.

2.4. Iterative Text Line Decomposition

A text line decomposition procedure using Sobel edge maps is proposed in [5] to obtain tighter and more accurate bounding boxes of the text areas. A vertical and a horizontal decomposition procedure are performed for several iterations. In each procedure, a horizontal (or vertical) scan line crosses a candidate text area and is segmented to one or more line segments \( l_r \). Then if the number or the density of horizontal (or vertical) Sobel edge points in the top and bottom \( r \) lines of \( l_r \) is not large enough, the line segment \( l_r \) will be deleted from the candidate text area. At last each candidate area is expanded to its bounding rectangle.

But because of the complexity of background, even after several iterations of decomposition vertically and horizontally, some candidate areas cannot be expanded to their bounding rectangle (in Figure 4 (d) the tie area remains). Our system proposes an improved text line decomposition method.

**Step 1:** Apply the text line decomposition method of [5] on the candidate text areas in the label map.

**Step 2:** After each candidate area is decomposed by step 1 (see Figure 4(d)), we get its bounding rectangle and set all pixels in this rectangle to their original grayscale values of integrated image. Thus we get another label map with new candidate text areas (see Figure 4(e)).

**Step 3:** Repeat step 1 on the new map generated by step 2 and get the final text bounding boxes of all text areas (see Figure 4(f)).

Figure 4 shows an example of Iterative Text Line Decomposition procedure.
composition procedure. It can be seen that after three steps, the candidate text lines are much closer to the real text boxes.

Then some rules are applied to remove some false alarms. (1) The width and the height of the text box must be larger than \( \text{min\_text\_width} \) and \( \text{min\_text\_height} \); (2) the horizontal-vertical aspect ratio of the text box must be bigger than 0.75. The text box that does not satisfy one or more of above rules will be removed.

2.5. Handling Scale Problem

To facilitate the detection of various text sizes, we use a two-level pyramid to handle the scale problem. The first level is the original image and the second is the image generated by halving the resolution of the original image. The detection algorithms on each level are the same. Then we fuse the detection results of two levels. If the bounding boxes at one level do not overlap with the boxes at the other level, we simply merge them into the original image; otherwise, we merge the boxes by forming a bounding box which contains all of them.

3. Experiments

We have tested our method on ABC news videos at a solution of 352x240. The total length of these videos is about 30 minutes. The data contains many different sources, including TV business news, commercials, stock reports, weather reports, etc. Our testing data contains 133 video clips and each clip consists of several consecutive frames containing the same stable captions. The caption detection approach is applied on each clip and MFI is performed. The approach without MFI is also applied on the first frame of each clip for comparison.

The ground truth data is created manually, and is compared with the detected results by our proposed method. The comparison is listed as below.

<table>
<thead>
<tr>
<th>Method</th>
<th>Total Textboxes</th>
<th>Detected</th>
<th>False alarms</th>
</tr>
</thead>
<tbody>
<tr>
<td>MFI</td>
<td>384</td>
<td>355</td>
<td>39</td>
</tr>
<tr>
<td>Non-MFI</td>
<td>384</td>
<td>325</td>
<td>102</td>
</tr>
</tbody>
</table>

Table 1. Comparison of our text detection approach

From Table 1, we can see that our MFI technique can significantly reduce the false alarms and at the same time the detection rate is improved too. The reason is that the improvement of the quality of background can enhance the contrast of some low-contrast text lines and make text detection much easier.

Some detection results are shown in Figure 5. It can be seen that the text bounding boxes are quite tight and accurate.

4. Conclusion

A texture-based video caption detection approach using Multi-Frame Integration is presented in this paper. Specially we only focus on the stable superimposed text.

There are two highlights in our proposed method. First, we proposed a novel MFI technique for text detection, which is applied on video frames before text detection process. Experimental result shows that it can significantly reduce the false alarms and improve the recall of detection. Second, we propose a new iterative text line decomposition method, which can obtain tighter and more accurate text bounding boxes of text areas.

Acknowledgement

The research described in this paper has been supported by NSF of China (69935010).

5. References