FACE DETECTION DIRECTLY FROM H.264 COMPRESSED VIDEO WITH CONVOLUTIONAL NEURAL NETWORK

Shin-Shan Zhuang and Shang-Hong Lai

Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan

ABSTRACT

Human faces provide a useful cue in indexing video content. In this paper, we propose a novel face detection algorithm based on a convolutional neural network architecture that can rapidly detect human face regions in video sequences encoded by H.264/AVC. By detecting faces directly in the compressed domain, we use the discrete cosine transform (DCT) coefficients in H.264 intra coding as the feature vector for face detection, thus it is not necessary to carry out additional DCT transform during the encoding or decoding process. With the face detector inside the video encoding process, we can adjust the coding parameters adaptively and allocate more resources to the macroblocks corresponding to the face regions. Some experimental results of applying the face detector on the H.264 intra coded images are given to demonstrate the performance of the proposed algorithm.

***Index Terms*—**Face detection, neural network, *DCT*, video codec.

1. INTRODUCTION

H.264/AVC, the new video coding standard outperforms previous standards, such as MPEG-4 and H.263, in terms of coding efficiency and video quality. H.264 provides variable block size motion estimation and more intra prediction modes. The intra mode is introduced in H.264 to achieve better encoding performance for texture blocks.

Most of the previous face detection methods were operated on the spatial domain [1, 2, 3], and they generally share similar processing procedures and structures. There were also some previous works on face detection in compressed domains. Wang and Chang [5] proposed a system that can detect human face regions in MPEG video sequences. They analyzed the distribution of skin-tone colors in Cb-Cr plane and used the Bayesian decision rule to classify a color into skin-tone class or nonskin-tone class. Then, they applied some shape constraints of human faces on the binary mask of the detected skin color region. Finally, given a candidate face region, they computed the energy of the DCT coefficients in the corresponding luminance block and used some thresholds to classify the region as face or nonface. Luo and Eleftheriadis [6] proposed a face detection algorithm for JPEG images and MPEG videos that combines both color and texture information. After searching the skin color region, they also used the DCT coefficients as their feature vector and employed a multimodal Gaussian model as the classifier. Similar to the K-means algorithm, they cluster the face feature vectors into six clusters of Gaussian distribution and cluster the nonface feature vectors into eight clusters. According to these distances between a testing feature vector and each cluster by a logarithmic Gaussian distance, we can classify the testing image into the face or nonface class.

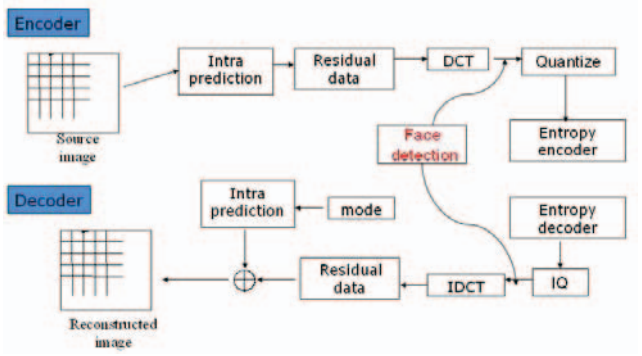

Fig. 1. Flow chart of our face detector inside the H.264 intra coding.

In this paper, we develop a feature-based face detection approach based on a convolutional neural network architecture. Different from the traditional face detection systems, our approach not only works on the compressed domain but also combines learning-based algorithm. In the compression domain, we can easily obtain the DCT coefficients during the encoding or decoding process. Therefore, we do not need to make any computational effort to obtain the feature vector. The flow chart of our face detection system for H.264 intra coding is shown in Fig. 1. The rest of this paper is organized as follows. The proposed face detection algorithm is described in details in section 2. Some experimental results on the H.264 intra coded images are given to demonstrate the performance of the proposed face detection algorithm in section 3. Finally, section 4 concludes this paper.

2. PROPOSED FACE DETECTION ALGORITHM

The proposed face detector is designed to locate multiple frontal faces of sizes between 64x64 pixels and 192x192 pixels in H.264 intra coded frames. The face images used for training are collected in a controlled environment with three conditions: 1) uniform light source onto the human faces, 2) face rotations are limited up to ±20 degrees, and 3) there is no occlusion to the face, including glasses.

2.1 Feature Vector for Face Detection

In the DCT domain, feature vectors are created directly from (block based) DCT coefficients. Illustrated from Fig. 2, we choose the first four coefficients in 4x4 transform block to generate the feature vector that represents the directional energy (horizontal, vertical, and diagonal edges) of the block in the spatial domain along with the DC coefficients. Since the size of our face model is 64x64 pixels (16x16 blocks), each one of the four corresponding DCT coefficients will create a 16x16 feature map, thus leading to the feature maps: MAP_{DC}, MAP_H, MAP_V, and MAP_D. In addition, we create another two feature maps: MAP_{MAX}, MAP_{MIN}. The map MAP_{MAX} is composed of the maximal value of the four parameters in a 4x4 block, and the map MAP_{MIN} is composed of the minimal value. In order to normalize these feature values, we rescale them to the range between 0 and 255 for each map. Comparing these six feature maps, we observe that the last four maps (MAP_V, MAP_D, MAP_{MAX}, MAP_{MIN}) give more consistent patterns on a variety of face examples. Thus, we only use these four maps as our feature vector instead of all the six maps.

DC	H		
V	D		

Fig. 2. The block structure corresponds to DCT coefficients

2.2 Convolutional Neural Network (CNN)

The CNN Architecture used in our face detector was inspired by the algorithm proposed by Garcia and Delakis [4], and the architecture chart is shown in Fig. 3(a). It consists of three layers, except the four input DCT maps of size 16x16 computed from a 64x64 input image.

Layer C1 is a convolutional layer with 14 feature maps. Each unit in the feature map is connected to a 4x4 neighborhood at identical locations in a subset of the DCT maps, illustrated in Fig. 3(b). The implementation corresponds to convolutions by 4x4 trainable kernels, followed by the addition of trainable biases. Here, outputs of different feature maps are fused in order to help in combining different features, thus extracting more complex information. Each of the four DCT maps provides inputs to two different feature maps of C1. This results in the first eight feature maps of C1. Each of the other six feature maps of C1 takes input from one of the possible pairs of different DCT maps. Therefore, layer C1 has 14 feature maps of size 7x7 and 334 (= 4*4*(4*2+6*2) + 14) trainable parameters.

Layers N1 and N2 contain classical neural units. These layers act as a classifier, and the DCT maps and Layer C1 are used as feature extractors. In layer N1, each of the 14 neurons is fully connected to all units of only one corresponding feature map of C1. The single neuron of layer N2 is fully connected to all the neurons of layer N1. The units in layers N1 and N2 perform the classical dot product between their input vector and the associated weight vector, to which a bias is added. This weighted sum is then passed through a hyperbolic tangent function to produce the state of the unit, with value between -1.0 and 1.0. The output of neuron N2 is used to classify the input as a nonface if its value is negative, or as a face if its value is positive. Layers N1 and N2 have 700(= 7*7*14+14) and 15(= 14+1) trainable parameters, respectively.

Concerning our learning strategy, all weights are computed through gradient-based learning, using a modified version of the backpropagation algorithm with momentum [8]. On the first iteration of the training loop, the network's weights are initialized randomly.

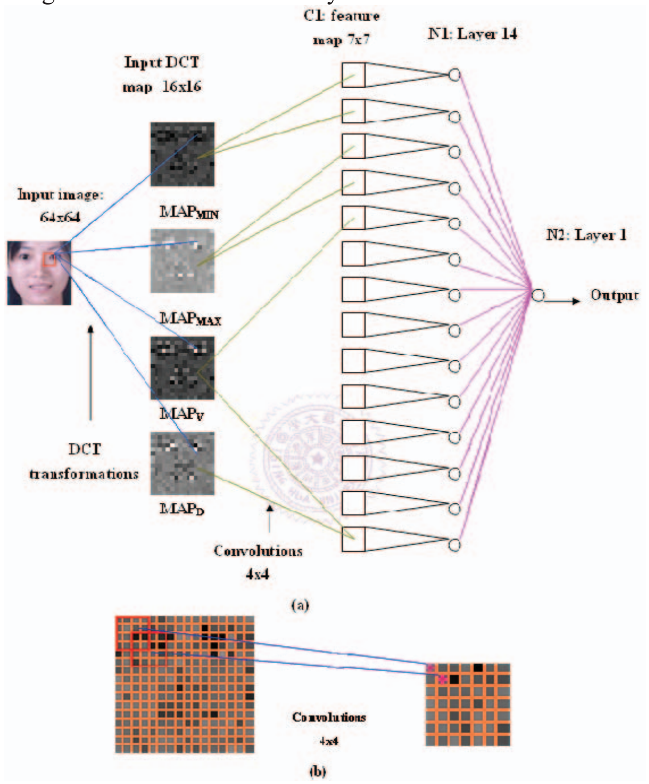

Fig. 3. (a)The three successive layers of CNN. (b)Receptive fields for convolution to generate a feature map of C1

2.3 Training Methodology

We built a training set by manually cropping 10,973 face image regions in a large collection of images collected from various sources over the Internet. Using manually labeled eye, mouth and ears positions, face images are cropped and rescaled to the size of 64x64 pixels. The aim of this procedure is to align theses faces to the same range approximately, while the distance from the mouth point to the eye-line (*Mouth2eye* in Fig. 4) is used to roughly preserve the original aspect ratio of the faces. In Fig. 4, the red dotted line is our borderline for cropping. We keep the Top_range and Bottom_range equal to (4/9)* *Mouth2eye* and (2/3)* *Mouth2eye*, respectively. In order to make more training data, we mirror these cropped faces. Therefore, we have totally more than 20000 faces in our training set.

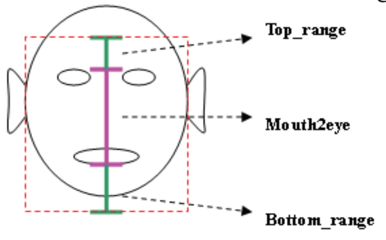

Fig. 4. Borderlines for cropping the face region.

Collecting a representative set of nonfaces is another problem. Practically, any randomly cropped image can serve as a nonface example, since the space of nonfaces can be seen as "the rest of the world." Because it is impossible to train the network with all possible non-face examples, nonface patterns were collected via an iterative bootstrapping procedure [4]. We iteratively re-train the system with an updated training set containing false alarms produced by applying the current face detector to a set of scenery images. In the bootstrapping process, an initial training set of 6,500 nonface patterns is built from nonface images, and the training set is updated with 3200 new non-face examples in each iteration.

2.4 Face Localization

To detect faces in a H.264 intra-coded image, we perform intra prediction and 4x4 DCT transform for the whole image first. Suppose the testing image is *H*-by-*W* blocks, and we compute the four DCT maps (MAP_V, MAP_D, MAP_{MAX} and MAP_{MIN}) of size *H*x*W* for face detection. Due to the block quantization problem mentioned above, to detect faces at multiple scales in the block-based DCT domain, we do not down-sample the image by 1.2 directly just like other face detection systems operating in the pixel domain. We choose to scan the DCT maps with the window sizes ranging from 16x16 to 48x48. Then we down-sample these windows to the face model size 16x16 to decide whether there is a face within the window or not. The procedures for our face detector in the DCT domain are illustrated in Fig. 5.

Before starting to scan the image block by block, we generate a score map of size *H*-by-*W*. We keep the windows with scores greater than a threshold *WinThr* (=0.0), and then we remove the windows with the corresponding scores less than *FaceThr* (=1.0~2.0). An example score map is depicted in Fig. 7(a). The intensities in the block indicate the scores. Fig. 7(b) shows the corresponding windows in Fig. 7(a), and Fig. 7(c) shows the results after we remove the windows with scores under the threshold *FaceThr*.

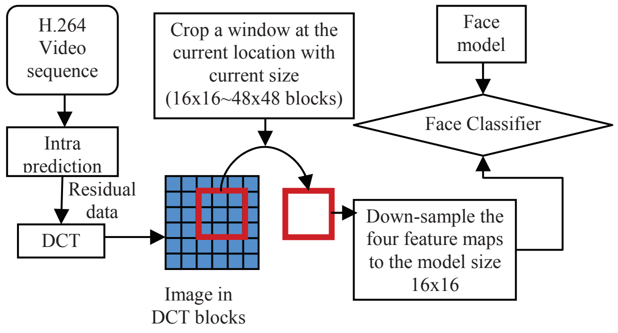
H.264 Video sequence
Crop a window at the current location with current size (16x16~48x48 blocks)
Face model
Face Classifier
Intra prediction
Residual data
DCT
Down-sample the four feature maps to the model size 16x16
Image in DCT blocks

Fig. 5. The face detection procedures in the DCT domain.

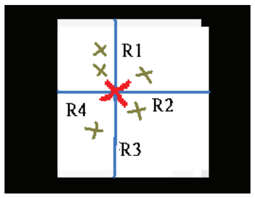

Fig. 6. The center of the *i*-th window for score grouping

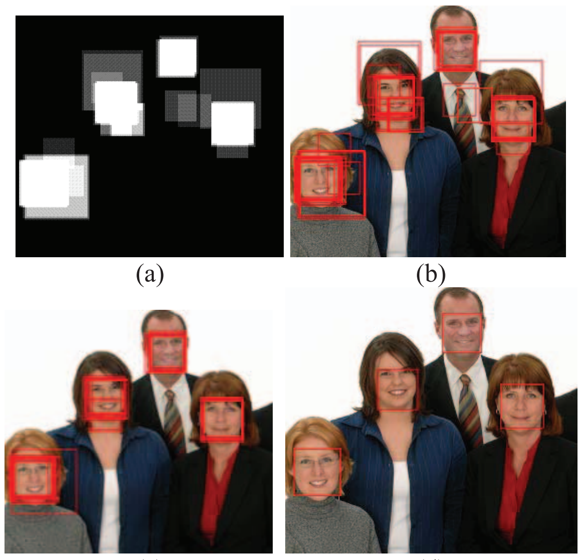

Fig. 7. (a)The score map, (b) all detected windows over the threshold *WinThr*, (c) the results after removing the windows with scores under the threshold *FaceThr*, and (d) detection results after grouping these detected windows.

We can see from Fig. 7(c) that there are generally multiple windows detected for a face. Therefore, we have to group these windows in order to give a specific location for each face in the image. Firstly, we make a *WinList* to record the size, score, central location, validation of these windows.

Suppose there are totally *N* windows detected in the testing image, then we apply the following steps:

```
for i=1…N  WinList[i]-> validation = true;
while !(all validation in WinList are false)
    pick up a window whose validation is false
    calculate MinDis
    merge all the windows with distance to WinList[i]->
    center within MinDis,
    calculate the average size, center and score of this
    group of windows and set their validations to false
end
```

We use the score map to calculate *MinDis*. Suppose we focus on the *i*-th window, Fig. 6 shows the score map with scores over *FaceThr* and points out the center of the *i*th window on the score map. Then we can calculate the distance from *WinList*[i]-> *center* to the top, right, bottom, and left boundary in the score map, indicated as *R1, R2, R3,* and *R4*, respectively. Finally, we set *MinDis* equal to the minimum value between *R1~R4*. The final results are shown in Fig. 7(d).

3. EXPERIMENTAL RESULTS

We implemented intra prediction and DCT transformation used in H.264 reference program on several test picture data sets and videos, and we used SATD criterion to make the mode decisions. In general, videos and images are compressed in YUV color channel, and our algorithm only works on the Y (luminance) channel.

We used two face image databases, including Yale, BioID databases, and three standard video sequences (Akiyo, Suzie and Miss America) as our testing data set. Because our face detection system does not contain training faces with glasses, we partition the testing databases into two groups, i.e. with and without glasses.

Because the proposed algorithm works on the H.264 intra-coded compressed domain, the block quantization problem and the limitation of the block resolution deteriorate the performance of our system inevitably. However, we still expect our face detector to have the same excellent performance as the pixel-domain face detector. Therefore, we compare the face detection performance between our system and Viola and Jones' system [9], which was implemented in the OpenCV software packages [7]. The comparison of the detection rate and false alarm is listed in Table 1. The reason why the detection rate on Suzie video sequence is not very high is because there are many non-frontal faces in the video. In general, we can preserve over 90 percent detection rate for faces with rotation angles within +/-20°.

4. CONCLUSION

In this paper, we proposed a new face detection algorithm on the compressed DCT domain for intra coded frames in H.264 video. The main novelty is the face detection on H.264 compressed domain in a learning-based framework. To sum up, this face detection work is novel because the face detector is performed on the DCT domain of the residual data produced by intra prediction instead of the original intensity data in the frame. Generally speaking, when we compress a video, we often use the information of the prediction error instead of the original intensity to achieve higher compression efficiency. From our experiments, we show that the proposed face detector can provide satisfactory results for face detection in the DCT compressed domain directly from the H.264 intra coded frames.

Table 1. Face detection results on the two testing face databases and three video sequences.

		Face #	**Detection rate%**		**False alarms**	
Database	**glasses**		OpenCV	Proposed	OpenCV	Proposed
Yale	without	129	100	86.04	0	8
	with	36	100	19.44	0	4
BioID	without	1037	96.43	91.90	282	54
	with	484	99.17	58.26	68	74
Sequence	**Frame #**		OpenCV	Proposed	OpenCV	Proposed
Akiyo	374	374	100	89.84	29	25
Suzie	374	374	72.19	71.39	21	13
Miss Am.	375	374	100	97.86	158	0

5. REFERENCES

[1] E. Osuna, R. Freund, and F. Girosi, "Training support vector machines: an application to face detection," IEEE Conf. Computer Vision and Pattern Recognition, pp. 130-136, 1997.

[2] H. Rowley, S. Baluja, and T. Kanade, "Neural network-based face detection," IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 20, No. 1, pp. 23-38, Jan. 1998.

[3] K.K. Sung and T. Poggio, "Example-based learning for view-based human face detection," IEEE Trans. Pattern Analysis Machine Intelligence, Vol. 20, No. 1, pp. 39-51, Jan.1998,.

[4] C. Garcia and M. Delakis, "Convolutional face finder: a neural architecture for fast and robust face detection," IEEE Trans. Pattern Analysis Machine Intelligence, vol. 26, no. 11, pp. 1408-1423, Nov. 2004.

[5] H. Wang and S.-F. Chang, "A highly efficient system for automatic face region detection in mpeg video," IEEE Trans. CSVT, 7(4), 1997.

[6] H. Luo and A. Eleftheriadis, "On face detection in the compressed domain," ACM Conf. Multimedia, pp. 285-294, Los Angeles, USA, 2000.

[7] OpenCV, http://www.sourceforge.net/projects/opencvlibrary

[8] J. Hertz, A. Krogh, and R.G. Palmer, Introduction to the Theory of Neural Computation. Reading, Mass.: Addison-Wesley, 1991.

[9] P. Viola and M. Jones, "Robust real-time face detection," International Journal of Computer Vision, Vol. 57, No. 2, pp. 137-154, May 2004