*International Journal of Internet, Broadcasting and Communication Vol.10 No.4 50-64 (2018)*
*http://dx.doi.org/10.7236/IJIBC.2018.10.4.50*

**IJIBC 18-4-9**

# Performance Improvement of Classifier by Combining Disjunctive Normal Form features

Hyeon-Gyu Min[1], Dong-Joong Kang[1*]

*[1]Dept. Mechanical Engineering, Pusan National University, Korea*
*[1*]Dept. Mechanical Engineering, Pusan National University, Korea*
*E-mail: min12@pusan.ac.kr, *djkang@pusan.ac.kr*

***Abstract***

*This paper describes a visual object detection approach utilizing ensemble based machine learning. Object detection methods employing 1D features have the benefit of fast calculation speed. However, for real image with complex background, detection accuracy and performance are degraded. In this paper, we propose an ensemble learning algorithm that combines a 1D feature classifier and 2D DNF (Disjunctive Normal Form) classifier to improve the object detection performance in a single input image. Also, to improve the computing efficiency and accuracy, we propose a feature selecting method to reduce the computing time and ensemble algorithm by combining the 1D features and 2D DNF features. In the verification experiments, we selected the Haar-like feature as the 1D image descriptor, and demonstrated the performance of the algorithm on a few datasets such as face and vehicle.*

***Keywords:*** *Object detection, ensemble learning, adaboost, 2D DNF cell classifier*

## 1. Introduction

Object detection [1], [2] is one of the most important tasks in the field of machine vision and has numerous applications [3]. In the past, object detection performance has been poor, but recently, the detection algorithms applying complex objects have been widely studied. In particular, object detection algorithms have been researched in more tangled environments, improving both accuracy and speed when the field of application is expanded.

Previous object detection algorithms perceived an object by comparing extracted features, however it was difficult to generalize such an algorithm. Because the features which represent an object can have a wide range of variations. Recently, the machine learning algorithm [4] has been developed which is capable of detecting novel types of data effectively.

Although a number of such learning algorithms are available, we have employed the Adaboost algorithm

Manuscript Received: Sep. 15, 2018 / Revised: Sep. 24, 2018 / Accepted: Sep. 30, 2018
Corresponding Author: djkang@pusan.ac.kr
Tel: +82-51-510-2356, Fax: +82-51-514-1118
Dept. Mechanical Engineering, Pusan National University, Korea

[5] in this paper. The Adaboost algorithm, which was initially used for face detection in real-time, constructs a strong classifier through a linear combination of weak classifiers by applying 1D features. We use the Haar-like feature [6] as 1D descriptor because it is simple and easily calculated. However, the 1D feature classifier has a reliable, in that the classification error is large when compared with the multi-dimensional feature classifier,

This paper proposes a 2D DNF (Disjunctive Normal Form) classifier. Its similar version has been applied for object tracking. It was introduced for online learning based tracking in our previous work [7]. The tracker solves the error drifting problem which is accumulated in each frame on the tracking algorithm.

Viola and Jones [8] presented a milestone paper about the field of face detection in which they proposed a Haar-like feature for detecting the face. In this paper, we also use the Haar-like feature to detect faces. While the Haar-like features are simple, Haar-like features has a relatively high accuracy. Viola and Jones also proposed the integral image for fast computing of the feature. This is useful for rapidly calculating the sum of pixels within rectangles. They also introduced the cascade method, which facilitates fast computing and increases detection performance. Finally, they proved that the Adaboost algorithm used with the Haar-like features has relatively better performance.

Among face detection algorithms there are two key issues: what features to extract, and which learning algorithm to apply.

Mita et al. [9] proposed joint Haar-like features, which improved the boosting algorithm. There are several modified versions of Adaboost available, such as Real Boost, and FloatBoost, which use extensions of the feature sets and allow various image patterns to be evaluated. So they proposed a new feature, called joint Haar-like feature, for detecting faces in images, based on the co-occurrence of multiple Haar-like features, to find a distinctive feature. The joint Haar-like feature can be calculated very fast, independently of image resolution, and has a robustness against the addition of noises and changes in illumination.

Another interesting recent work is [10], where the authors proposed a new weak classifier called Bayesian stump. Bayesian stump is also a histogram based weak classifier, however, the split thresholds of the Bayesian stump are derived from the iterative split and merge operations instead of being at equal distances and fixed.

Another well-known feature that is robust to illumination variations is the local binary patterns (LBP) [11], which is very effective for face recognition tasks.

Another popular complex feature for face/object detection is based on regional statistics such as histograms. Levi and Weiss [12] proposed local edge orientation histograms, which compute the histogram of edge orientations in sub-regions of test windows. These features are then selected by an Adaboost algorithm to build the detector.

The DNF was proposed in [7] and is used for solving the error drift problem of online tracking [13], [14].

When tracking different objects, the most suitable features to employ may not always be the same, and with this in mind, feature selection techniques were researched by [15]. The feature pre-selector chooses the feature with the best performance. Therefore, the time required for calculating features is reduced.

In this paper, we proposed the 2D DNF classifier to offline learn in order to detect an object of interest in a complex environment, using the Adaboost algorithm and the formulation of a novel type of weak classifier, and diversified features are applied to the DNF weak classifier. We could make a DNF feature for choose a

different 1D features. For example, The DNF is paired by EOH (edge orientation histograms), LBP (Local binary pattern), and Haar-like feature. Thus, we show and analyze the relative performance of proposed method for object detection with offline learning.

The remainder of this paper is organized as follows. In the next section, we give a brief definition of the adaboost and the DNF cell classifier. Section 3 describes the proposed selecting feature method and the procedure of constructing a strong classifier from 1D classifier and 2D DNF classifiers. Section 4 presents the experiments where the classifier is applied to some practical real images, and the results for the proposed classifier are compared with a 1D feature classifier. Finally, Section 5 outlines the conclusions.

## 2. Definition of weak classifier

The object detection method developed in this paper is a kind of supervised learning algorithm. The algorithm is trained on both positive images and negative images. After training, it can be used to distinguish object features from background features on new data. The positive images are interested object and negative images are background images. We chose Adaboost as the learning algorithm. Because the Adaboost algorithm makes an ensemble based strong classifier by combining a number of weak classifiers, it has good performance.

### 2.1 Adaboost

We define the feature vector as $\{\mathbf{x}_i, y_i\}_{i=1}^{N}$. $\mathbf{x}_i$ is a feature vector of the $i^{th}$ sample image in training data set and $y_i \in \{1 or -1\}$ is the label of each image. N is the sample number and the weak classifier function is given by $h_t(\mathbf{x})$ in (1).

$$h_t(\mathbf{x}) \begin{cases} +1, & \mathbf{x} \text{ is positive} \\ -1, & \mathbf{x} \text{ is negative} \end{cases} \tag{1}$$

Where $h_t(\mathbf{x})$, called weak classifier, is a mapping function that transforms a training image x into class label (1 or -1). The function $h_t(\mathbf{x})$ is built from training sample images including positive and negative examples.

The strong classifier defines a linear combination of weak classifiers as shown in (2).

$$H(\mathbf{x}) = \sum_{t=1}^{T} \alpha_t h_t(\mathbf{x}) \tag{2}$$

### 2.2 DNF cell classifier

A multi-dimensional feature classifier can improve the performance of the 1D feature classifier in a typical Adaboost algorithm, however, the classifier is complex and takes a great deal of learning time. In contrast, the 2D DNF cell classifier we have proposed provides the advantages of reduced calculation time and improved performance, because it extends the dimensional space by combining just a pair of 1D features and the feature pair could increase discriminative power than single feature.

Let $\{\mathbf{P}_f, y\}$ denote the $f^{th}$ cell sample and its label, where each element of $y$ is -1 or 1, $\mathbf{P}_f = [d_{h_i}, d_{h_j}]$ is

a 2D matrix composed of 1D weak classifiers, N is the total number of samples of the positive and negative types. Where $d_{h_i}$ and $d_{h_j}$ is feature value for $h_i$ and $h_j$ weak classifier, respectively. So $\mathbf{P}_{\mathrm{f}}$ is the matrix with two feature values per a sample in each column.

Define the 2D plane from two feature values and the plane $d_{\mathrm{h}_i} - d_{\mathrm{h}_j}$ is quantified into $m \times m$ bins. We vote the feature data in this plane.

For example, all of the sample data are voted on the plane quantified $5 \times 5$ bins in Fig. 1. The number of positive feature data and negative feature data is distributed differently in each of the bins. We define the positive bins as PosNum, and negative bins as NegNum for the bins in (3) ($|.|$ indicates the cardinality of the set).

We explained the training steps about 2D cell classifier. First, the feature data of $d_{\mathrm{h}_i}$ and $d_{\mathrm{h}_j}$ are extracted from sample images and vote them on the plane. Second, we quantified the surface likes in Fig. 1. Third, count the positive feature data and negative feature data in the each of the bins. If one of bins has positive feature data more than negative feature data, we save the index of the bins. They are defined in (3). PosNum( $\mathbf{P}_{\mathrm{f}}$ ) is the total number of samples satisfying both conditions that the pair value for samples in is involved in bin position (k,l) and its label is positive.

$$\begin{aligned} PosNum(\mathbf{P}_{\mathrm{f}}) &= \left|\mathbf{P}_{\mathrm{f}} \mid \mathbf{P}_{\mathrm{f}} \in bin(k,l) \cap \mathbf{P}_{\mathrm{f}} \in (y_{\mathrm{f}} = +1)\right| \\ \mathrm{NegNum}(\mathbf{P}_{\mathrm{f}}) &= \left|\mathbf{P}_{\mathrm{f}} \mid \mathbf{P}_{\mathrm{f}} \in bin(k,l) \cap \mathbf{P}_{\mathrm{f}} \in (y_{\mathrm{f}} = -1)\right| \end{aligned} \tag{3}$$

$$b_{kl} = \begin{cases} +1, & \mathrm{PosNum}(\mathbf{P}_{\mathrm{f}}) > \mathrm{NegNum}(\mathbf{P}_{\mathrm{f}}) + r \\ -1, & \text{otherwise} \end{cases} \tag{4}$$

In (4), $b_{kl}$ is the classification value for the bin position (k,l) of the 2D histogram.

If the number of positive sample data is larger than that of negative samples by r in $b_{kl}$, the $b_{kl}$ is classified as positive (+1); otherwise $b_{kl}$ is classified as negative (-1). Then we save the index of $b_{kl}$. The parameter r is a marginal constant

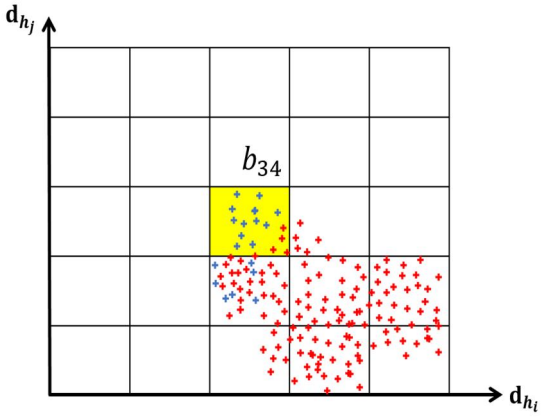



**Figure 1. Feature data distribution in a 2D plane; blue points are positive feature data and red points are negative feature data.**

In Fig. 1, we save the $b_{34}$ as a positive bin because it satisfies the positive condition of (4).

Summing up $b_{kl}$ for all bins gives a DNF classifier $\mathrm{h}_{\mathrm{DNF}}^{\mathrm{f}}(\mathbf{x})$ for $\mathrm{h}_i$ and $\mathrm{h}_j$ weak classifier in (5).

$$\mathrm{h}_{\mathrm{DNF}}^{\mathrm{f}}(\mathbf{x}) = \begin{cases} +1, & x \in \left( \bigcup_{1 \le k,l \le m} b_{kl} = +1 \right) \\ -1, & \text{otherwise} \end{cases} \tag{5}$$

Where $\bigcup_{1 \le k,l \le m} \mathrm{b}_{kl} = 1$ represents the union of all bins where the number of positive samples is larger than that of negative samples by at least r.

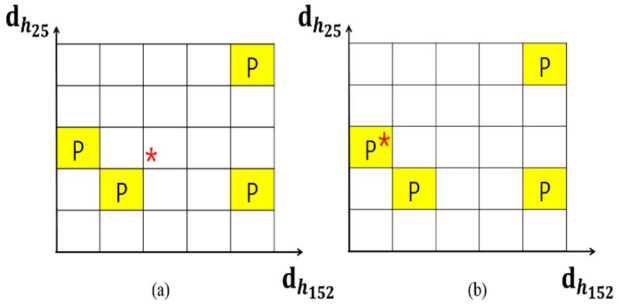



**Figure 2. Classifying the new input data(red star) by using the 2D DNF cell classifier. (a) Sample is assigned to negative; (b) Samples of positive assignment**

Figure 2 is an example showing the new data (red star) classified by using the 2D DNF cell classifier. The positive bin is represented as "P" on the basis of the learning result. The positive bins are decided by (4) with learning samples or two 1D weak classifier $h_{25}$ and $h_{152}$. If the new data is voted in the "P", it is classified as positive. Otherwise the data is classified as negative.

## 3. Methods for selecting feature and constructing ensembles

We propose a method for choosing features that reduces the learning time and building the strong classifier with 2D DNF feature that improves the classification performance. Using this method, we select features from among all of the 1D features, and generate the 2D DNF by combining the 1D features that were selected. As a result, it is possible to maintain the performance of the classifier even when the number of features has been increased, while keeping reasonably the computation time.

### 3.1 Selecting 2D DNF features

We can generate hundreds of thousands of Haar-like features in model image regions, and millions of 2D DNF can be created as combined pairs of 1D features. 2D DNF features are made by picking two different 1D features. For example, if the number of 1D features is 10,000, we can create ${}_{10,000}C_2$ of 2D features. However, the computing time for such feature combinations is very large, because the dimensions of the 2D

DNF cell classifier are higher than that of 1D feature classifier.

Before describing the strategy for selecting a feature, we first extract Haar-like features from an image and create the 1D classifier based on the extracted features. In conventional method using 1D feature, it finds a threshold to distinguish the positive from the negative samples. The classifier determines whether the new sample data is positive or negative by the threshold. In this paper, we define a 2D DNF feature classifier to $\mathrm{h}_{\mathrm{DNF}}(\mathbf{x})$ in (6).

$$\mathrm{h}_{\mathrm{DNF}}(\mathbf{x}_t) = \begin{cases} +1 & , \quad \mathbf{x}_t \; is \; positive \\ -1 & , \quad \mathbf{x}_t \; is \; negative \end{cases} \tag{6}$$

We train the classifier by applying the supervised learning algorithm that uses the label of the sample images. So, we calculate the classification error by comparing the label of the sample images with the result of 2D DNF feature classifies.

$$err_t = \sum_{i=1}^{N} w_i \left| \mathrm{h}_{\mathrm{DNF}}(\mathbf{x}_i) - y_i \right| \tag{7}$$

$$w_{t+1,i} = w_{t,1} \left( \frac{err_t^{1-e_t}}{1 - err_t} \right) \tag{8}$$

The result of the 2D DNF classifier, $\mathrm{h}_{\mathrm{DNF}}(\mathbf{x})$, is +1 or -1. $w_i$ represents the weight of each sample and is updated by the classification error. If the error is large, the weight of the sample image is large. Consequently, we select the feature depending on the weight.

We now introduce the strategy for selecting 2D DNF classifiers from among candidate pairs obtained from the selected 1D weak classifiers.

**Table 1. Strategy for selecting features**

Initial of weight $w_{i,1} = \frac{1}{N}$

for k=1: M

Normalize the weights.

$$w_{i,k} = \frac{w_{i,k}}{\sum_{i=1}^{N} w_{i,k}}$$

Find a weak 2D DNF classifier with minimum error from all candidates and save the error value for $\alpha_{DNF}$ :

$$err_k = \sum_{i=1}^{N} w_i \left| \mathrm{h}_{\mathrm{DNF}}(\mathrm{x}_t) - \mathrm{y}_t \right|$$

Update the weights

$$w_{t+1,i} = w_{t,i}\left(\frac{err_t^{1-e_t}}{1-err_t}\right)$$

end

N is the number of training sample images: we used 1,000 positive images and 1,000 negative images, and several types of the Haar-like features were used as 1D features.

### 3.2 Building the strong classifier

The 2D DNF is the normal form for 2D features by picking the different two of 1D features and not repeating the same 1D feature. This form of features will be able to classify indistinguishable features in 1-dimensional in the complex environment

Even though the 2D DNF classifier is composed of a paired combination of 1D feature classifiers, we found that an object detection algorithm which only uses a 2D DNF classifier does not have higher performance than a 1D feature classifier in offline learning. Therefore, in this paper, we propose a strong classifier employed by Adaboost combines a 1D feature classifier with a 2D DNF classifier, which makes its performance better than that of a 1D feature classifier. This strong classifier can classify objects which cannot be classified by a 1D feature classifier.

There are two possible methods for making a strong classifier with 2D DNF features.

In the first, when the error is obtained in the sample image, the algorithm chooses the 1D feature which is the minimum classification error for all test samples. Candidates for 2D DNF are generated from pairs of some chosen 1D features, and then the best 2D DNF are successively selected by the proposed method. In the second, all of the 1D features are combined to make the 2D DNF and then some 2D DNF features are selected.

In this paper, experiments were conducted using the first method. The reason for using the first method was to maintain good performance and to reduce the computing time for generating the 2D DNF. To save the computing time, reducing the total number of 1D and 2D DNF is important.

To maximize performance, we selected T as the number of 1D features and M as the number of 2D DNF features and made a linear combination of the two types of weak classifiers. The 2D DNF feature is instrument in improving the performance of the strong classifier. Combinations of 1D features and 2D DNF features are shown in (9).

Usually, the performance of 2D DNF classifier is better than that of 1D feature classifier when the positive and negative features agglomerate together to form larger clusters. The performance of 1D feature classifier is much better than that of 2D DNF classifier when the features are easy to separate between positive features and negative features. For this reason, we make the best use of 1D feature classifier for classify the positive features and combine the 2D DNF classifier with 1D feature classifier for reducing the errors.

$$\mathrm{H}(\mathbf{x}) = \sum_{t=1}^{T} \alpha_{\mathrm{f}} \mathrm{h}_t(\mathbf{x}) + \sum_{\mathrm{f}=1}^{M} \alpha_{\mathrm{f}} \mathrm{h}_{2D}^{\mathrm{f}}(\mathbf{x}) \tag{9}$$

## 4. Experiments and results

**(a) Face images and non-face images**

**(b) Car images and background images**

**Figure 3. Training sample images**

We implemented the proposed algorithm using the sample images provided by the MIT center for biological and computational learning and Visual Geometry Group in University of Oxford. The sample was face image and cars (rear) image data. Face image is a grayscale image of pixels and composed of 2,429 face images and 4,548 non-face images. We trained 1,000 face and non-face images, respectively, and evaluated the performance on 472 face images and 472 non-face images that had not been utilized in the training. And car image is converted to gray scale image and resized into pixels. Original images are 128 x 128 pixels. The car images are composed of 1,152 car images and background images respectively. We trained 500 car images and 500 background images and evaluated the 400 car and 400 background images that had not been used in the training. Fig. 3 shows a few samples for training the classifier

### 4.1 Feature extraction

The conventional Adaboost algorithm is based on several hundreds of thousands of produced Haar-like features. For the purpose of comparison in this paper, we simply used three of the elementary feature types, shifting the position and changing the scale. We show the elementary features in Fig. 4. The Haar-like features become different features depending on their shape and position. First, define the acting position and

scale in the training image for one among base shapes of Fig. 4. Then, a Haar-like feature is produced for the position and scale. If we change the position and scale, the different features are defined again

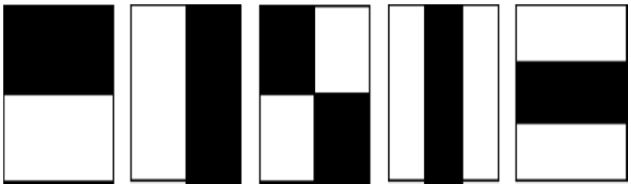


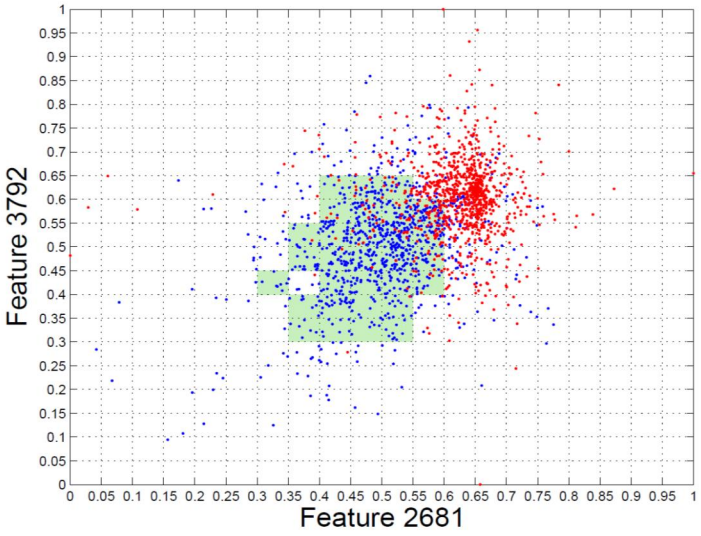
**Figure 4. Haar-like features in the experiment**

In face images, the default size of the Haar-like feature is pixels. We generated many kinds of features by changing the feature position within a pixel region and changing the scale into pixels. A total 22,899,528 2D DNF were possible from the combination of 1D features, because we used 6,768 1D features.

In car images, we give a same condition with experiment of face images. Therefore, we get the 47,682 1D features, and hundreds of thousands of 2D DNF features.

### 4.2 Experimental results

Figure 5 is a result of the learning model of 2D DNF cell classifier. Blue dots are the feature values of the positive images, red dots are the feature values of the negative images. Usually the distribution of positive and negative samples is mixed in the feature space, and it is difficult to distinguish the blue dots from red dots by the only 1D feature classifier, however, the mixed dots distribution can be better classified by using the 2D DNF cell classifier. And it shows the positive bins in a green color. The bins are labelled positive when the number of positive samples dropped in the bin position exceeds the number of negative samples by at least r. Each axis of the 2D feature region is normalized between 0 and 1 and each axis is divided into 20 grids.



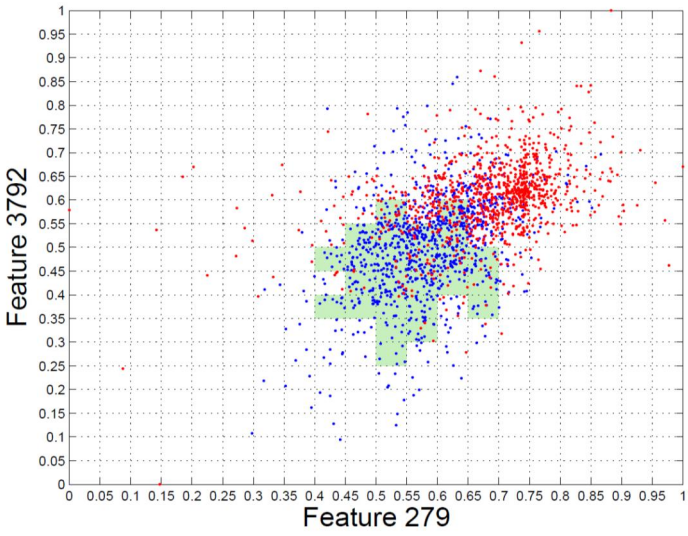



**Figure 5. 2D DNF distribution displays the positive bins.**

Figure 6 shows some selected features by the proposed method. Generally, the classifiers of the Haar-like features on the nose or around the eyes distinguish the positive and negative samples well. This can be seen from the features which have been classified as positive and negative, illustrated above. We define these features as important features. Even though it is clear that the classifier of all the features is better able to find interesting objects, however, for distinguishing between an object of interest and the background, it is not necessary to use all features. The classifier error composed of strong features may be bigger than to use all the features, but instead, learning speed is much faster.

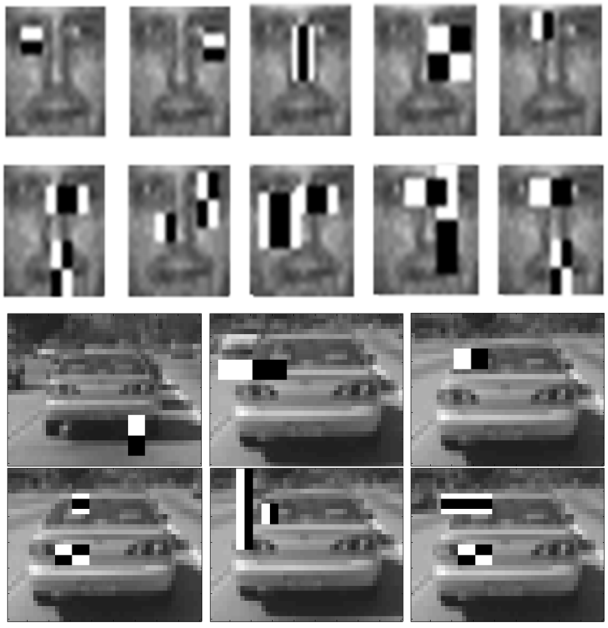


**Figure 6. The results of selection methods 1D features (first and third row) and 2D DNF features (second and fourth row)**

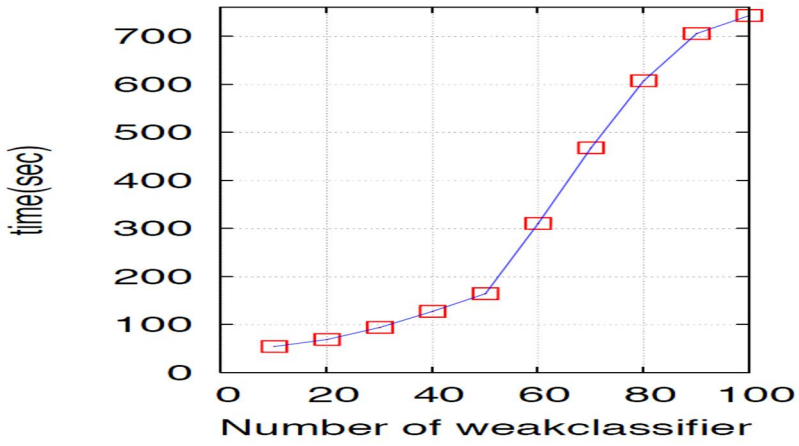
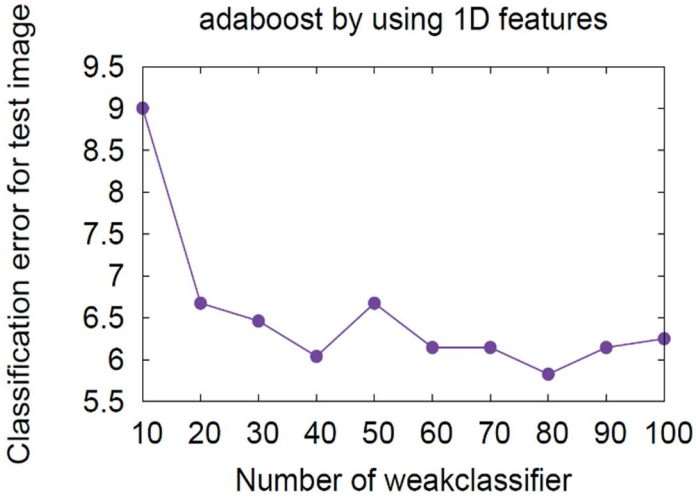


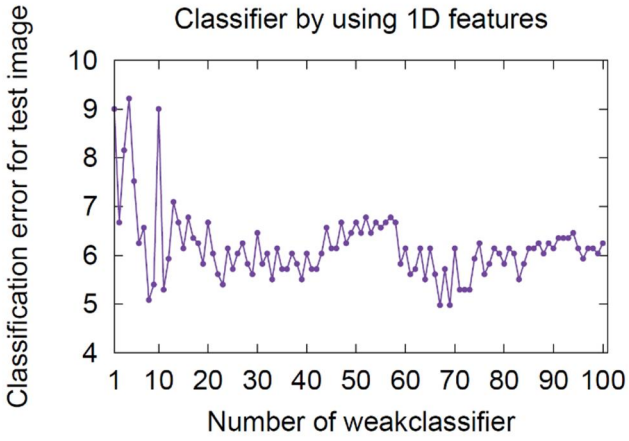

**Figure 7. The computing time for 2D DNF feature selection using the selection algorithm of Section 3.**

Figure 7 shows the processing time of the DNF classifier. For example, if we select 10 of the 1D features, we can make 45 combination pairs for the DNF feature. However, if 100 1D features are selected we can make 4,950 combination pairs for the DNF feature. The number of 2D DNF features increases exponentially. Creating a feature of 2D DNF for the 6,768 individual 1D features used in the actual experiment generated 22,899,528, and that requires a large memory. To learn about all the features in order to reduce the minimum classification error, hardware support is required and computing cost is too high. However, choosing a strong feature with the proposed method requires similar computing time for learning, without increasing computing resources, and detects objects better.

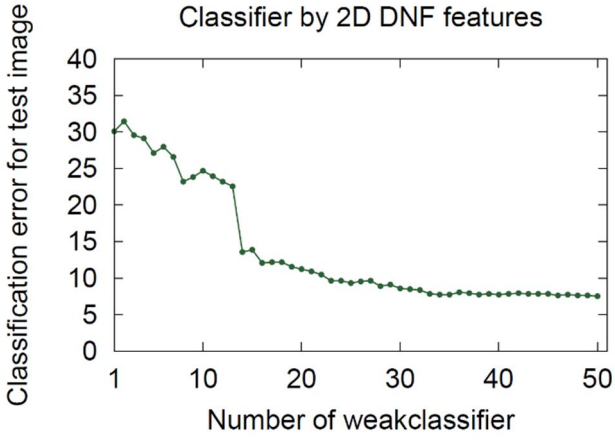
One of most important things for machine learning algorithm is to improve the classification performance even though it is not big. The proposed method enhances the performance of conventional Adaboost algorithm when it is not improve any more by additional combination of weak classifiers.



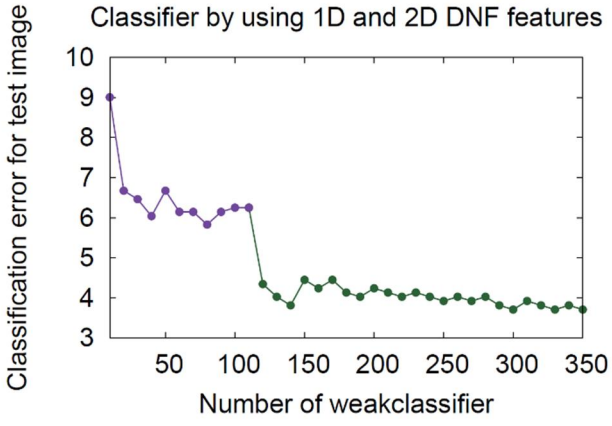

**(a) Classifier by using 1D features which is added 10 unit**




**(b) Classifier by using 1D features which is added one unit**




**(c) Classifier by using only 2D DNF features**




**(d) Classifier by using 1D features and 2D DNF features.**

**Figure. 8. Classification error for test images of face when the adaboost increase the number of weak classifier.**

Figure 8 shows the reduction of classification error for test images when we combine 2D DNF features to conventional Adaboost method.

First, we found the strong classifier by using 1D features only. Fig. 8(a) and 8(b) is result of applying the

Adaboost algorithm. Two figures show the classification error when we add the weak classifier by 10 unit and one unit, respectively. When we add the feature for generating the strong classifier one by one, the classification error value is reduced further and further, as shown in Fig. 8(a) and (b). In this paper, we use the 100 features selected by algorithm of section 3 in producing a strong classifier. After adding 100 weak classifiers, error is not reduced even if we add more features, so using only 100 weak classifier is enough to reduce the error with balancing the computing time.

In Fig. 8(c), the classifier is 2D DNF features with boosting algorithm, as shown in (12). Since it is generated by linear combination of weak classifiers, the graph was similar to result of performance for 1D classifier which was shown that the more weak classifiers were combined, the better performance of classifiers became. In Fig. 8(d), additional performance improvement for conventional methods is achieved when we combine 1D feature and 2D DNF. The blue line shows classification error of classifier by using 1D features and the red line shows classification error of the strong classifier which is linearly combined with additional 2D DNF classifier generated by 1D features. The strong classifier with 2D DNF features shows error of classification about 7%, however the strong classifier combined of 1D feature and 2D DNF features shows error of classification about 4%.

To evaluate the classifier, we tested the performance of the proposed method on 472 positive images and 472 negative images that were not involved in training. As the number of features increased, classification error was reduced. However after joining more than 100 1D features, the performance of the classifier did not improve any more.

We could generate a 4,950 of 2D DNF features from first 100 1D weak classifiers among the strong classifier. However, the 2D DNF cell classifier using the selection algorithm in Section 4 used a total of 300 features of 100 1D and 200 2D classifiers, because it was enough to reduce the error in this experiment.

As shown in Table 2, the proposed method reduced the object detection error by about 2.437% for face and by 3.875% for car, respectively, when compared to the one that uses only 1D features for classification. That corresponds to a 39.7% and 47.7%, respectively, relative error reduction when compared to the conventional Adaboost performance using only 1D features.

The SVM classifier shows an error of detection 9.957% about face and 5.875% about car. The SVM classifier is better than adaboost algorithm. However, the method combining the 2D DNF classifier shows the minimum error when compares the SVM and adaboost.

**Table 2. Comparison of classification error.**

| A kind of classifier | Face | Car |
|---|---|---|
| SVM | 9.957% | 5.875% |
| 1D feature classifier (Conventional Adaboost) | 6.144% | 8.125% |
| 1D+2D features classifier (2D DNF classifier combined) | 3.707% | 4.250% |

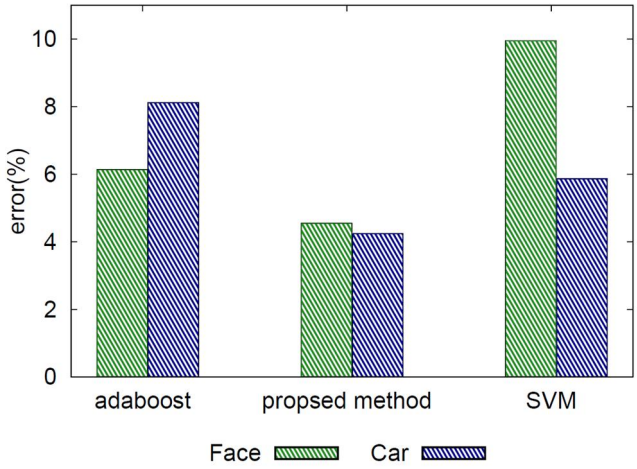



**Figure 9. Error comparison of the proposed method with adaboost and SVM method.**

## 5. Conclusion

A learning algorithm which uses the existing classifier of 1D features has the advantage of being simple, and the learning time is short, but the performance in a complex environment is not enough. Therefore, in this paper, we created a new classifier by combining the 2D DNF and 1D features, and showed that performance was improved compared with the existing classifier of 1D features. Furthermore, as we reduced the number of features by selecting features based on the classification error, the learning time of classifier could be reduced. For future work, we can use other kinds of features, such as local binary patterns, and local edge orientation histograms, to enhance the performance of object detection in a more complex environment. The multi-dimensional combination of different types of features might increase learning and detection performance even more.

## Acknowledgement

This work This work (Grants No. C0138997) was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government(MSIT) (No.2016R1A2B4007608), and National IT Industry Promotion Agency (NIPA) grant funded by the Korea government (MSIT) (No.S0602-17-1001).

Business for Cooperative R&D between Industry, Academy, and Research Institute funded Korea Small and Medium Business Administration in 2014.

## References

[1] Pedro F. Felzenszwalb, Ross B. Girshick, David McAllester, and Deva Ramanan, “Object Detection with Discriminatively Trained Part-Based Models”, *In IEEE Trans. PAMI*, vol. 32, no. 9, 2010.

[2] H. Grabner and H. Bischof, “On-line boosting and vision”, *In IEEE Conference on CVPR*, vol. 1, pages 260–267, 2006.

[3] Dong-Wook Kim, Woo-Youl kim, Jisang Yoo and Young-Ho Seo, “A Fast and Accurate Face Tracking Scheme by using Depth Information in Addition to Texture Information”, *Journal of Electrical Engineering & Technology*, vol. 9, no. 2, pages 707-720, 2014.

[4] Y. Bai and M. Tang, “Robust Tracking via Weakly Supervised Ranking SVM”, In *IEEE Conference on CVPR*, pp1854-1861, 2013.

[5] Oscar Martinez Mozos, Cyrill Stachniss, and Wolfram Burgard "Supervised Learning of Places from Range Data using Adaboost," *In IEEE Robotics and Automation*, pp. 1730-1735, 2005.

[6] Rainer Lienhart and Jochen Maydt, "An Extended Set of Haar-like Features for Rapid Object Detection," *In IEEE Proc. Conference on Image Processing*, vol. 1, pp. I-900-I-903, 2002.

[7] Zhu Teng, and D.J. Kang, "Disjunctive Normal Form of Weak Classifier for Online Learning based Object Tracking," *In Proc. Conference on VISAPP*, pp. 138-146, 2013.

[8] Paul Viola and Michael Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features," *In IEEE Proc. Conference on CVPR*, vol. 1, pp. I-511-I-518, 2001.

[9] Takeshi Mita, Toshimitsu Kaneko, and Osamu Hori, "Joint Haar-like Features for Face Detection", *In IEEE Conference on ICCV*, vo1. 2, pp 1619-1626, 2005.

[10] Rong Xiao, Huaiyi Zhu, He Sun, and Xiaoou Tang, "Dynamic Cascades for Face Detection", *In IEEE Conference on ICCV*, pp.1-8, 2007.

[11] Timo Ojala, Matti Pietikainen, and Topi Maenpaa, "Multiresolution Gray-Scale and Rotation Invariant Texture Classifiecation with Local Binary Patterns", *In IEEE Trans. PAMI*, vol. 24, no. 7, pp. 971-987, 2002.

[12] Kobi Levi and Yair Weiss, "Learning Object Detection from a Small Number of Examples: the Importance of Good Features", *In IEEE Proc. Conference on CVPR*, vol.2, pp. II-53 - II-60, 2004.

[13] Vo Quang Nhat and Gueesang Lee, "Illumination Invariant Object Tracking with Adaptive Sparse Representation", *International Journal of Control, Automation, and Systems*, vol. 12, no. 1, pp.195-201, 2014.

[14] D. Wang, H. Lu, and M.-H. Yang, "Online Object Tracking with Sparse Prototypes", *In IEEE Trans. Image Processing*, vol. 22, no.1, pp.314-325, 2013.

[15] Robert T. Collins, Yanxi Liu, and Marius Leordeanu, "Online Selection of Discriminative Tracking Features", *In IEEE Trans. PAMI* , vol. 27, no. 10, pp.1631-1643, 2005.