2013 Second IAPR Asian Conference on Pattern Recognition

Classifier Introducing Transition Likelihood Model Based on Quantization Residual

Yuji Yamauchi
Chubu University
Email: yuu@vision.cs.chubu.ac.jp

Takeo Kanade
Carnegie Mellon University
Email: tk@cs.cmu.edu

Hironobu Fujiyoshi
Chubu University
Email: hf@cs.chubu.ac.jp

***Abstract*—Binary codes that are binarizations of features represented by real numbers have recently been used in the object recognition field, in order to achieve reduced memory and robustness with respect to noise. However, binarizing features represented by real numbers has a problem in that a great deal of the information within the features drops out. That is why we focus on quantization residual, which is information that drops out when features are binarized. With this study, we introduce a transition likelihood model into classifiers, in order to take into consideration the possibility that a binary code which has been observed from an image will transition to another binary code. This enables classifications that consider transitions to the desired binary code, even if the observed binary code differs from the actually desired binary code for some reason. From the results of experiments, we confirmed that the proposed method enables an increase in detection performance while maintaining the same levels of memory and computing costs as those for previous methods of binarizing features.**

I. Introduction

There is intense research under way into the highly accurate detection of pedestrians from images, in order to implement driving assistance for drivers and security systems for reassurance and safety. Since there are large variations in which the pedestrians that are the detection object can be seen, due to various different causes such as their orientation and pose, it is necessary to have features that enable the extraction of elements that are as far as possible common from many images of pedestrians. For that reason, there have been many proposals for features that are intended to be robust with respect to such factors[1], [5], [12], [3]. The most successful approach is a Histogram of Oriented Gradients (HOG) features proposed by Dalal, et al. [1]. HOG features are a high-dimensional features in which gradient information obtained from each pixel is represented as a gradient orientation histogram in a local region. HOG features that enable the perception of pedestrians shapes have a high detection performance, despite their comparative simple processing, and are being used in many pedestrian detection methods [14], [2], [10].

Another recent approach that is popular has the objective of producing mobile terminals that are aiming towards the commercial viability of people detection techniques [4] and running on hardware such as Field-Programmable Gate Arrays (FPGAs) [7]. To enable operation on such a device, it is becoming essential to have features that use little memory and also enable highly accurate detection. Binarized features have been proposed as one such method [6], [11], [13]. In general, a feature is represented as feature vectors with real numbers as elements, but each of the features described above consist of a number of encodings that have been binarized by some sort of benchmark, and combined into a single feature. Yamauchi et al. have reduced the memory used for representing features by subjecting HOG features to threshold value processing then binarizing in accordance with the magnitude relationships of two histograms [13]. In addition, Mu et al. have used a Local Binary Pattern (LBP) [8] as a feature in pedestrian detection [6], to represent magnitude relationships with neighboring pixels. Such binarized features also have the advantage of being robust with respect to noise.

However, the process of quantizing a features into binary form creates a problem in that a great deal of the information within the features drops out. If information divided into two classes is included in the missing information, the classification capability deteriorates when a binarized features are is used. It is reported in [13] that simply binarizing a HOG features that is represented by real numbers results in a deterioration in detection performance.

In this study, we focus on the information that drops out during the binarization of the feature. This missing information is called a quantization error that is defined as the difference in values before and after quantization, but in this study we define the missing information as quantization residual since the values after the quantization are 0 / 1. A quantization residual indicates the difference between a real number value and a threshold value, when the values of a feature represented by real numbers are subjected to threshold value processing, by way of example. One characteristic of quantization residuals is that binary encoding that is stable and less likely to invert is obtained when the quantization residual is larger. If the quantization residual is small, on the other hand, unstable binary encoding in which inversions can easily occur are obtained. Binary codes that is expressed by combining a number of binary encodings is a discrete variable that forms another feature by simply inverting one binary code. For that reason, it can happen that even when features are similar when represented by real numbers, they are not observed to be the same binary code when binarized. Essentially, since it is necessary to extract elements that are common to the detection objects, in order to detect pedestrians highly accurately, such representations of features are not suitable.

That is why, in this study, a transition likelihood model that represents the relationships between binary code based on quantization residuals is introduced into classifiers. A transition likelihood model represents the degree of likelihood of an observed binary code x transitioning to another binary code x'. This study uses transition likelihood distributions created

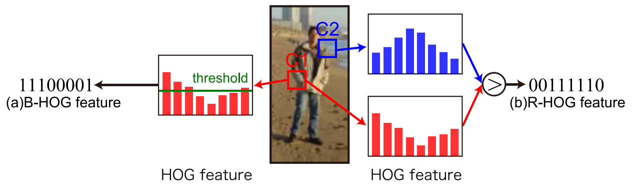


Fig. 1. Binary codes. (a)B-HOG feature, (b)R-HOG feature.

on the basis of binary code and quantization residuals obtained from training samples, as a transition likelihood model. This predicts transitions from an observed binary code to another binary code, in accordance with the transition likelihood distribution created during the classification. By introducing this binary code transition prediction into classifiers, it becomes possible to take into consideration transitions to the originally obtained binary code, even if the observed binary code differs from the actually desired binary code for some reason.

II. HOG FEATURES AND BINARIZATION

In this study, we use a HOG feature[1] that enables perception of the human form, and also binary code that are binarized forms of that feature. In this case, we discuss a binarized HOG feature and a relational HOG feature [13], as a HOG feature and a binary code.

A. Binarized HOG

A binarized HOG (B-HOG) feature is a feature that has been proposed in order to reduce the memory for a HOG feature in which elements are represented by real numbers. The B-HOG feature is obtained by subjecting gradient orientation histograms $\boldsymbol{V}_c$ in eight orientations within a certain cell c to threshold value processing as shown in Eq. (1), to obtain binary encoding, as shown in Fig. 1(a).

$$x_c^B(n) = \begin{cases} 1 & \text{if} \quad v_c(n) > th \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

We compute the binary code for all 8 orientations in accordance with Eq. (1), and take the thus-obtained binary coded $\boldsymbol{x}_c^B = \{x_c^B(1), x_c^B(2), \cdots, x_c^B(8)\}$ as the B-HOG feature.

B. Relational HOG

B-HOG features provide a valid method for reducing memory, but they have a problem in that they necessitate an optimal threshold value. A relational HOG (R-HOG) feature by which binarization is done by comparing two gradient orientation histograms, as shown in Fig. 1(b), has been proposed[13] as a binarization method that does not necessitate a threshold value. An R-HOG feature is binarized by comparing the magnitudes of gradient orientation histograms $\boldsymbol{V}_{c_1}^H$ and $\boldsymbol{V}_{c_2}^H$ in eight orientations obtained from two cell regions, as shown in Eq. (2).

$$x_{c_1,c_2}^R(n) = \begin{cases} 1 & \text{if} \quad v_{c_1}(n) > v_{c_2}(n) \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

The binary code $\boldsymbol{x}_{c_1c_2}^R = \{x_{c_1c_2}^R(1), x_{c_1c_2}^R(2), \cdots, x_{c_1c_2}^R(8)\}$ obtained in this manner forms the R-HOG.

C. Problems with previous methods

The amount of memory for representing a feature can be reduced to $1/64$ by binarizing a feature represented by real numbers(double :8byte). However, the reduction in the amount of memory for representing the feature inevitably leads to a large reduction in the feature representation capability. This is because information that is valid for classification is comprised in the quantization residual that is information that drops out during the binarization. In fact, it has already been reported in [13] that detection performance deteriorates when HOG features represented as real numbers are binarized by threshold value processing. In addition, since the binary code that are used in previous pedestrian detection methods are discrete variables, a completely different feature is represented if just one code is different. For that reason, it can happen that the same binary code cannot be observed, even if features that are similar appear during the representation by real numbers.

III. PROPOSED METHOD

Fig. 1 shows the procedure of classification based on transition likelihood model. Our approach introduces into classifiers a transition likelihood model created based on quantization residuals, which have not been used at all previously. The transition likelihood model outputs a transition likelihood that expresses the possibility of an observed binary code transitioning into another binary code. Since the proposed method makes it possible to represent relationships between binary code even though they are discrete variables, by considering these transition likelihoods during classification, it can output classification results that are even more reliable.

In this section, we first define the quantization residual of a binary code and discuss the transition likelihood distribution created on the basis of quantization residuals as a transition likelihood model. We then discuss classifiers into which the binary code transition likelihood model has been introduced.

A. Quantization residual

The proposed method focuses on quantization residuals, which are the amounts of information that drop out when a feature represented by real numbers is binarized.

B-HOG feature
The quantization residual of a B-HOG feature is the difference between the gradient strength $v_c(n)$ of a gradient orientation histogram in a orientation n and a threshold value th, as shown by Eq. (3):

$$q_c^B(n) = v_c(n) - th \tag{3}$$

We obtain quantization residuals $\boldsymbol{q}_c^B = \{q_c^B(1), q_c^B(2), \cdots, q_c^B(8)\}$ in all 8 orientations from Eq. (3).

R-HOG feature
The quantization residual of an R-HOG feature is the difference between two gradient orientation histograms $v_{c_1}(n)$, $v_{c_2}(n)$, as shown by Eq. (4):

$$q_{c1,c2}^R(n) = v_{c_1}(n) - v_{c_2}(n) \tag{4}$$

We obtain quantization residuals $\boldsymbol{q}_{c1,c2}^R = \{q_{c1,c2}^R(1), q_{c1,c2}^R(2), \cdots, q_{c1,c2}^R(8)\}$ in all 8 orientations from Eq. (4).

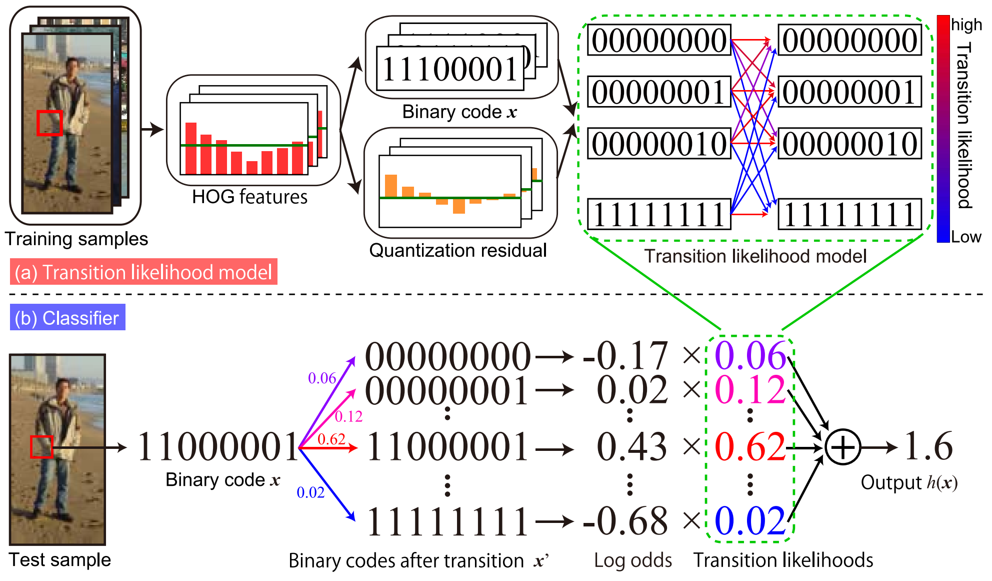


Fig. 2. Process of the proposed method. (a) Training transition likelihood model based on binary codes and quantization residuals from training samples. (b) Classifier based on the transition likelihood model

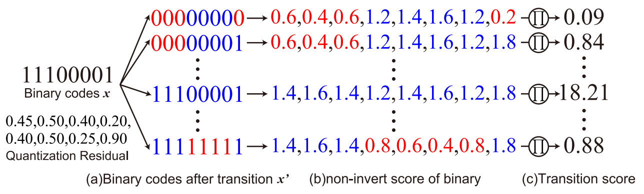


Fig. 3. Flow to compute the transition score of binary code based on quantization residuals. Blue represents that a binary code does not invert. Red represents to invert of a binary code.

B. Transition likelihood distribution of binary code

We create a binary code transition likelihood distribution in order to represent how possible it is that a binary code will transition to another binary code . The flow for calculating the transition scores of binary code are shown in Fig. 3.

1) Binary code transition score: To calculate the binary code transition score, we first obtain the degree of non-inversion z of the binary code. As shown in Fig. 3, when we have assumed that a binary code $\boldsymbol{x}$ that is observed from a certain training sample transitions to another binary code $\boldsymbol{x}'$, we determine the degree of non-inversion from whether or not there is a transition for each binary code $\boldsymbol{x}(n)$ that makes up the binary code $\boldsymbol{x}$ and that quantization residual. In this study, we consider the following two points on the calculation of the degree of non-inversion of binary code:

1) Whether or not the binary code inverts
2) Magnitude of the quantization residual

Taking into account the above two points, our considerations are divided into four patterns. First of all, if there is no inversion of the binary code and the quantization residual is large, the degree of non-inversion increases, but if the quantization residual is small, the degree of non-inversion decreases. If there is inversion of the binary code and the quantization residual is large, the degree of non-inversion decreases, but if the quantization residual is small, the degree of non-inversion increases. In the calculation of the degree of non-inversion of this study, we use a concave function $F(q)$ and a convex function $\bar{F}(q)$ in which one-dimensional functions are combined, as shown in Fig. 4(a). We calculate the degree of non-inversion of the binary code by using a concave function $F(q^i(n))$ and a convex function $\bar{F}(q^i(n))$, as shown in Eq. (5), from a binary code $x^i(n)$ observed from a sample i and whether or not there is a transition in a binary code $x'(n)$ when we considered transitions to a certain binary code $\boldsymbol{x}'$.

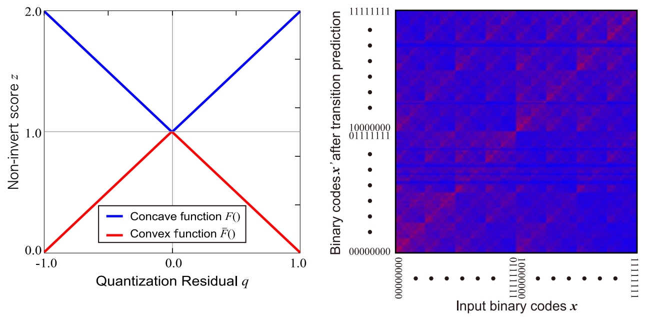
$$z(x^i(n),x'(n),q^i(n))=\begin{cases}F(q^i(n)) \text{ if } x^i(n)=x'(n)\\ \bar{F}(q^i(n)) \text{ otherwise}\end{cases} \tag{5}$$

The degree of non-inversion of binary code obtained from Eq. ((5)) is high when there is no transition in the binary code (codes indicated by the blue characters in Fig. 3(b)) and low where there is a transition (codes indicated by the red characters). In addition to the presence or absence of transitions of the binary code, a transition score corresponding to the value of the quantization residual is output.

2) Transition scores of binary code: We obtain a transition score $e(\boldsymbol{x}'|\boldsymbol{x}^i)$ for the binary code on the assumption that the binary code $\boldsymbol{x}$ transitions to the binary code $\boldsymbol{x}$', from the thus-obtained degree of non-inversion z of the binary code. The transition score of the binary code is obtained by taking the sum total of the degrees of non-inversion of the binary encoding, as shown in Eq. (6), to take into account synchronism of the binary encoding.

$$e(\boldsymbol{x}'|\boldsymbol{x}^i,\boldsymbol{q}^i)=\prod_{n=1}^{8} z(x^i(n),x'(n),q^i(n)) \tag{6}$$

This is obtained from all of the training samples I, to create a transition likelihood distribution E for the binary code by



(a) Non-invert score

(b) Transition likelihood distribution

Fig. 4. (a) Non-invert score of a binary code is computed from concave function $F()$ and convex function $\bar{F}()$. (b) Visualization of transition likelihood distribution $E(\boldsymbol{x}'|\boldsymbol{x})$ of binary code.

summing them as shown by Eq. (7):

$$E(\boldsymbol{x}'|\boldsymbol{x}) = \sum_{i=1}^{I} e(\boldsymbol{x}'|\boldsymbol{x}^i, \boldsymbol{q}^i)\delta[\boldsymbol{x}^i - \boldsymbol{x}'] \tag{7}$$

$\delta[\cdot]$ is the Kronecker delta function, which outputs 1 when $\boldsymbol{x}^i - \boldsymbol{x}' = 0$ and 0 otherwise. We create a transition likelihood distribution E as described above for each feature.

An example of transition likelihood distributions that we have created is shown in Fig. 4(b). We used a B-HOG feature and the threshold value th used in the binarization of the HOG feature was 0.09, in a similar manner to [13]. This transition likelihood distribution has higher transition likelihoods as codes become more similar to the input binary code. For example, if the input binary code is {00000000}, the binary code after transition where the Hamming distance is 0 is naturally the most likely. When the Hamming distance increases, on the other hand, transitions in the binary code are unlikely to occur, so it is clear that the transition likelihood is less. However, transition likelihoods assume values that are different from the Hamming distances because they are determined on the basis of the quantization residuals of the training samples. For that reason, when transitions of the binary code are likely to occur, in other words, when there are many samples with small quantization residuals within the training samples, the proposed method is better at representing transitions between binary code that are likely to occur in practice than with Hamming distance.

C. Transition prediction based on quantization residual

In this section, we discuss classifiers into which are introduced a transition likelihood model created on the basis of quantization residuals obtained from training samples (Fig. 2(b)).

1) Real AdaBoost classifier: In our research, we use Real AdaBoost [9], which enables highly accurate and fast classifications, as the statistical training method. A strong classifier $H(\boldsymbol{x})$ that is trained by the Real AdaBoost algorithm is represented by linear linking of weak classifiers $h_t(\boldsymbol{x})$, as shown in Eq. (8):

$$H(\boldsymbol{x}) = \sum_{t=1}^{T} h_t(\boldsymbol{x}) \tag{8}$$

where T is the total number of weak classifiers that are combined and t is the number of each weak classifier. Subsequently, since the number of the weak classifier is irrelevant, we express $h(\boldsymbol{x})$ as a general weak classifier. The weak classifiers $h(\boldsymbol{x})$ are determined by the log ratio of the probability of occurrence of positive class or negative class, as shown in Eq. (9).

$$h(\boldsymbol{x}) = \frac{1}{2}\ln\frac{W_+(\boldsymbol{x})}{W_-(\boldsymbol{x})} \tag{9}$$

where the probability density function $W_\pm$ of the binary code is created by summing the weighting w_i of the training samples as shown by Eq. (10):

$$W_+(\boldsymbol{x}) = \sum_{\boldsymbol{x}^i=\boldsymbol{x} \wedge y_i=+1} w_i, \quad W_-(\boldsymbol{x}) = \sum_{\boldsymbol{x}^i=\boldsymbol{x} \wedge y_i=-1} w_i \tag{10}$$

2) Classifiers using transition likelihood distributions: Since this study considers the transition from the observed binary code $\boldsymbol{x}$ to another binary code $\boldsymbol{x}'$, we introduce the transition likelihood model into Eq. (11) and define each weak classifier $h(\boldsymbol{x})$ as shown by Eq. (11).

$$h(\boldsymbol{x}) \triangleq \frac{1}{2}\sum_{\boldsymbol{x}'\in\mathcal{X}}\left(P(\boldsymbol{x}'|\boldsymbol{x})\ln\frac{W_+(\boldsymbol{x}')}{W_-(\boldsymbol{x}')}\right) \tag{11}$$

where $P(\boldsymbol{x}'|\boldsymbol{x})$ represents the probability of the binary code $\boldsymbol{x}$ transitioning to the binary code $\boldsymbol{x}$', $\mathcal{X}$ means universal set of binary codes x. However, in practice the observed binary code does not transition to another binary code so $P(\boldsymbol{x}'|\boldsymbol{x})$ cannot be obtained. That is why in this study, we substitute a transition likelihood distribution $E(\boldsymbol{x}'|\boldsymbol{x})$ that represents the possibilities of transitions between binary code, as shown by Eq. (12). The transition likelihood distribution $E(\boldsymbol{x}'|\boldsymbol{x})$ can simulate the representation of the transition probability $P(\boldsymbol{x}'|\boldsymbol{x})$ of binary code that cannot be observed in practice.

$$P(\boldsymbol{x}'|\boldsymbol{x}) \approx \frac{E(\boldsymbol{x}'|\boldsymbol{x})}{\displaystyle\sum_{\boldsymbol{x}'\in\mathcal{X}} E(\boldsymbol{x}'|\boldsymbol{x})} \tag{12}$$

If the binary code $\boldsymbol{x}$ is not observed during this time, we consider that there is also no transition between binary code. In such a case, both the denominator and numerator of Eq. (12) are zero, so $P(\boldsymbol{x}'|\boldsymbol{x}) = 0$. As described above, classification can be done by taking into account the possibility of an observed binary code transitioning into another binary code, from Eq. (11) and Eq. (12). However, since Eq. (11) is considered for transitions to all of the binary codes, the log ratio of the probability of occurrence of a binary code that differs greatly from the observed binary codeg also has an effect on the weak classifiers. Since the log ratio is obtained during this process even when the probability of occurrence of the probability density function $W_\pm$ is equivalent to substantially zero, an extremely large or small value is added, which has an adverse effect on the value of the weak classifier $h(\boldsymbol{x})$. To resolve that problem, we subject the transition likelihoods to threshold value processing as shown in Eq. (13), to suppress the effects on the classification results of binary- codes with low possibilities of transition from the observed binary code.

$$E'(\boldsymbol{x}'|\boldsymbol{x}) = \begin{cases} E(\boldsymbol{x}'|\boldsymbol{x}) & \text{if } \ E(\boldsymbol{x}'|\boldsymbol{x}) > \epsilon \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

This threshold value ϵ creates a model that does not take account of transitions of the binary code if it is set to an extremely large value. In this study, we set $\epsilon = 0.005$ from preliminary experiments. Weak classifiers using the binary code transition likelihood distribution that we propose (Eq. (11)) seem at first glance to have larger computational costs when compared with the Real AdaBoost weak classifiers (Eq. (9)) that are generally used. However, the same computational costs are achieved by previously computing outputs for each binary code that is observed in practice, and saving them as a look-up table.

IV. Evaluation Experiments

In our experiments, we compare B-HOG and R-HOG features, which are features that have been binarized as proposed in [13], and classifiers in accordance with the proposed method. Binary codes transition prediction model based on the quantization residuals of training samples is introduced into these classifiers. We use the INRIA person dataset[1] as the dataset used in these evaluation experiments.

A. Experiment results

A DET curve of the results of the experiments is shown in Fig.5. First of all, if the HOG feature , B-HOG feature , and the proposed method based on the B-HOG feature are compared, the proposed method, the HOG feature , and the B-HOG feature are shown to be in decreasing order of detection performance. The detection rate of the B-HOG feature , which is a HOG feature that has been subjected to threshold value processing where the FPPW is 10^{-3}, deteriorated by approximately 2.9% in comparison with the HOG feature. When the proposed method based on the B-HOG feature where the FPPW is 10^{-3} was compared with the HOG feature and B-HOG feature, the detection rate rose by approximately 5.4% and 8.4% respectively. From the above, we see that the detection performance deteriorated when the HOG feature was simply binarized by threshold value processing, but we were able to obtain detection performances that exceeded those of the HOG feature by introducing binary code predictions into the classifiers, in accordance with the proposed method. In addition, this tendency produced similar results even when the binarization method was different. A comparison of the R-HOG feature where FPPW was 10^{-3} and the proposed method based on the R-HOG feature demonstrated an increase in detection performance of approximately 3.0%.

B. Discussion

We have confirmed from the results of the evaluation experiments that weak classifiers into which a binary code transition likelihood model has been introduced contribute to an increase in detection performance, irrespective of the method used to binarize the feature. In this paragraph, we discuss to what degree is the detection performance of each weak classifier raised by the introduction of binary code transition predictions.

The results of miss-classification rates of general Real AdaBoost weak classifiers (Eq. (9)) and the miss-classification rates of weak classifiers into which transition predictions have been introduced (Eq. (11)) are plotted on two-dimensional

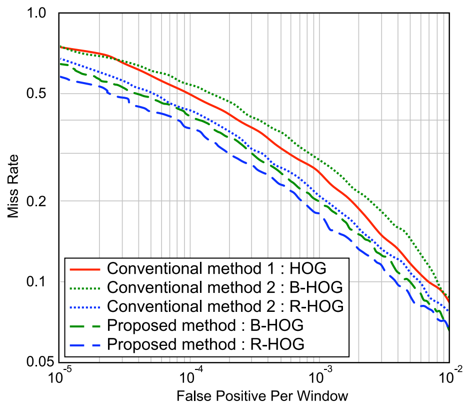


Fig. 5. DET curves of experiment.

graphs in Fig. 6. All of the plotted points represent miss-classification rates when the same feature is used. If there is no change in the performance of a weak classifier, it is plotted on the red line indicating $y = x$. If the detection performance is higher with the proposed method, the point is plotted to the right and below the red line. From Fig. 6, it is clear that the detection performance of a large number of weak classifiers is increased by the proposed method. When based on the B-HOG feature, the detection performance for approximately 96.2% of the weak classifiers can be raised, and when based on the R-HOG feature, the detection performance for approximately 88.3% of the weak classifiers can be raised. The classification rate for each individual weak classifier increases by approximately 5.0% at a maximum. However, since Real AdaBoost, which is an ensemble training method in which large numbers of weak classifiers are combined, is used as the training method, the final classifiers trained by the proposed method can achieve an even higher detection performance.

From the above results, we have confirmed that the proposed method enables pedestrian detection that is more accurate than that of previous methods. With previous methods that use binary code, binary code that are discrete variables are handled as mutually independent values. For that reason, if the observed binary code has been observed to be another binary code for some reason, only the observed binary code will be considered during the classification, so it is possible that that classification results will vary widely. The proposed method, on the other hand, forms a framework which enables predictions of how likely the observed binary code will transition into all the other binary codes. Since the binary code transitions are based on the transition likelihood distribution created from the binary code obtained from training samples and the quantization residuals, the possibilites of binary code transitions that occur readily in practice are considered. For that reason, even if a binary code has been observed to be another binary code for some reason or other, that adverse effect can be restrained.

V. Conclusions

In this paper, we proposed a method of effective utilization of quantization residuals, which have not been used previously

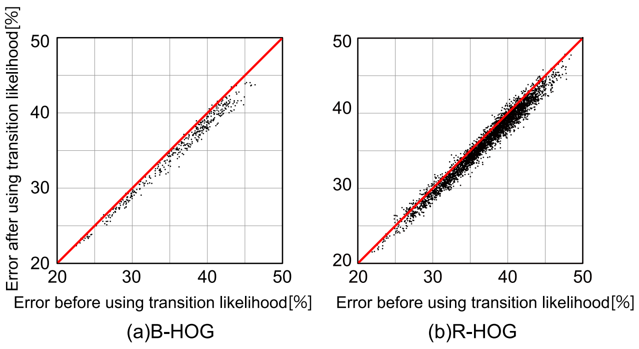


Fig. 6. The graph of error computed from Eq. (9) and Eq. (11). The ploted points are represented equal error rate. (a) B-HOG feature base. (b) R-HOG feature base.

in the pattern recognition field. The proposed method introduces a transition likelihood model into classifiers, in order to consider the possibility of a binary code that has been observed from an image transitioning into another binary code. The proposed method is capable of outputting highly reliable classification results since it can consider the possibility of an observed binary code transitioning into another binary code. The results of experiments show that the proposed method enables an increase in detection performance while maintaining the same levels of memory and computing costs as those for previous methods of binarizing features.

The approach that makes effective use of quantization residuals in accordance with the proposed method is implemented by Boosting-based classifiers in this study, but we consider it will be possible to expand it into other classifiers such as Random Forest and SVM. We will examine expansion into such training methods in the future.

References

[1] N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. In *Computer Vision and Pattern Recognition*, volume 1, pages 886–893, 2005.

[2] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object Detection with Discriminatively Trained Part Based Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2009.

[3] C. Hou, H. Z. Ai, and S. H. Lao. Multiview Pedestrian Detection Based on Vector Boosting. In *Asian Conference on Computer Vision*, pages 210–219, 2007.

[4] B. Leibe, K. Schindler, N. Cornelis, and L. V. Gool. Coupled Object Detection and Tracking from Static Cameras and Moving Vehicles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(10):1683–1698, 2008.

[5] K. Levi and Y. Weiss. Learning Object Detection from a Small Number of Examples: the Importance of Good Features. In *Computer Vision and Pattern Recognition*, volume 2, pages 53–60, 2004.

[6] Y. D. Mu, S. C. Yan, Y. Liu, T. Huang, and B. F. Zhou. Discriminative local binary patterns for human detection in personal album. In *Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[7] V. Nair, P.-O. Laprise, and J. J. Clark. An FPGA-Based People Detection System. *EURASIP Journal on Advances in Signal Processing*, 2005(7):1047–1061, 2005.

[8] T. Ojala, M. Pietikainen, and D. Harwood. A comparative study of texture measures with classification based on featured distributions. *Journal of the Pattern Recognition*, 29(1):51–59, 1996.

[9] R. E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336, 1999.

[10] W. R. Schwartz, A. Kembhavi, D. Harwood, and L. S. Davis. Human Detection Using Partial Least Squares Analysis. In *International Conference on Computer Vision*, 2009.

[11] X. Wang, T. X. Han, and S. Yan. An HOG-LBP Human Detector with Partial Occlusion Handling. In *International Conference on Computer Vision*, 2009.

[12] B. Wu and R. Nevatia. Detection of Multiple, Partially Occluded Humans in a Single Image by Bayesian Combination of Edgelet Part Detectors. In *International Conference on Computer Vision*, pages 90–97, 2005.

[13] Y. Yamauchi, C. Matsushima, T. Yamashita, and H. Fujiyoshi. Relational HOG Feature with Wild-Card for Object Detection. In *Workshop on Visual Surveillance(in conjunction with ICCV2011)*, 2011.

[14] Q. Zhu, S. Avidan, M. C. Yeh, and K. T. Cheng. Fast Human Detection Using a Cascade of Histograms of Oriented Gradients. In *Computer Vision and Pattern Recognition*, pages 1491–1498, 2006.