2014 International Joint Conference on Neural Networks (IJCNN)
July 6-11, 2014, Beijing, China

Traffic Sign Recognition using a Novel Permutation-based Local Image Feature

Tian Tian, Ishwar Sethi and Nilesh Patel

***Abstract*— Traffic sign recognition (TSR) is an essential research issue in the design of driving support system and smart vehicles. In this paper, we propose a permutation-based image feature to describe traffic signs, which has an inherent advantage of illumination invariance and fast implementation. Our proposed feature LIPID (local image permutation interval descriptor) employs interval division and zone number assignment on order permutation of pixel intensities, and takes the zone numbers as the descriptor. A comprehensive performance evaluation on German Traffic Sign Recognition Benchmark (GTSRB) dataset is carried out, which reveals the great performance of our proposed method. Experiment results exhibit that our feature outperforms some state-of-the-art descriptors, showing a potential prospect in TSR applications.**

I. Introduction

Traffic Sign Recognition (TSR) is being studied as an important component of advanced driver assistance systems [1][2]. Recognition of traffic signs works for warning drivers about potential dangers and inappropriate behaviours, playing a valuable role in protecting driving safety and avoiding traffic accidents. General TSR system involves the following basic aspects: capture videos or images containing traffic signs within the visual field, detect signs in the captured images, classify and recognize what the detected sign is, and track the motion of signs if needed. Many researchers have attained achievement in their designs of TSR systems, such as Zaklouta *et al*. [3], Baro *et al*. [2] and Bahlmann *et al*. [4]. The implementation of TSR refers to various aspects of computer vision and machine learning, including shape and color analysis, feature extraction and image representation, as well as multi-class classification. As an important application of artificial intelligence, TSR is being developed with the aims of high accuracy and real-time operation.

State-of-the-art algorithms mainly focus on fast and effective representation algorithms of traffic signs and efficient classification methods. Currently, a number of image features have been applied in the detection and recognition of traffic signs. For example, Haar-like features and histogram of oriented gradient (HOG) features were applied in traffic sign detection and recognition in the work of Bahlmann *et al*. and Xie *et al*. [4][5]. Besides the above traditional features, more image features which are specially designed for real-time implementation have been employed in this task. Ruta *et al*. proposed a Color Distance Transform (CDT) which enables robust discrete-color image comparisons in real time [6]. Zheng *et al*. introduced the scale and rotation invariant BRISK features in their TSR system for sign representation and recognition [7]. As the popularity of affordable mobile devices, research findings are brought out of laboratory and put into practical tests, followed by a higher requirement of better detection and recognition accuracy and faster computation speed.

In this paper, we propose a novel permutation-based image descriptor and implement it in the traffic sign recognition experiments on German Traffic Sign Recognition Benchmark (GTSRB) dataset. As we know, TSR applications always handle with image sequences captured under different lighting conditions and require real-time computing. Our feature just meets the demands of TSR tasks, with the advantages of low computation complexity and inherent invariance to monotonic illumination changes. Our experiments involve several state-of-the-art image features, and employ three widely used supervised learning methods for sign recognition. Evaluation results show that our proposed image feature achieves performance as good as HOG and reveals a best computation speed among the all.

II. Related work

A. Image descriptors

Image representation through local descriptors is always a research interest in computer vision, for different descriptors have been widely used in numerous applications including object matching, recognition, image retrieval and motion tracking. Generally, distinctiveness and computation speed are two primary requirements of designing image features, however, they are usually two trade-off aspects.

State-of-the-art algorithms have reached good performance, aiming applications with either requirements in precision or speed. Lowe's SIFT (Scale Invariant Feature Transform) [8] is well-known for its excellent invariance to various affine deformation. Though SIFT exhibits great performance in many applications, the high complexity and computational cost impose restrictions on its development. Bay *et al*. proposed SURF (Speed-up Robust Feature) [9] as an alternative to SIFT, which performs comparably with SIFT meanwhile maintaining most desirable properties. Based on the work of Papageorgiou *et al*., Viola and Jones adapted the idea of using Haar wavelets and developed the so-called

Tian Tian is now a PhD candidate of Automation School, Huazhong University of Science and Technology, Wuhan, Hubei, China (email: tiantian@hust.edu.cn)

Ishwar Sethi and Nilesh Patel are with Department of Computer Science and Engineering, Oakland University, MI, USA (email: {isethi, npatel}@oakland.edu)

This work was supported by China Scholarship Council and was completed in Oakland University.

978-1-4799-1484-5/14/$31.00 ©2014 IEEE

Haar-like features [10][11]. They were used in the first real-time face detector and famous for the fast computation using integral images. HOG (Histogram of Oriented Gradients) was first described for the problem of pedestrian detection by Dalal and Triggs [12]. HOG describes an image by counting occurrences of gradient orientation in localized portions of an image, since it improves accuracy in pedestrian detection, it has been soon introduced to many other detection and recognition fields.

With the development of affordable image sensors and mobile devices, applications requiring low computation complexity and real-time implementation continue to spring up. Therefore, image descriptors of non-floating points are paid more attention. BRIEF (Binary Robust Independent Elementary Feature) [13] proposed by Calonder *et al*. has led the development of binary descriptors. it uses binary strings by simply comparing the intensities of pixel pairs located in fixed positions to construct a descriptor. I-BRIEF [14] and D-BRIEF [15] then improved original BRIEF on robustness and distinctiveness. ORB (Oriented Fast and Rotated BRIEF) [16], BRISK (Binary Robust Invariant Scalable Keypoint) [17] and FREAK (Fast Retina Keypoint) [18] further develop BRIEF on invariance properties by improving keypoint detection and utilizing more effective sampling patterns. In order to resist to monotonic brightness changes, permutation method is then introduced into construction of descriptors. MROGH (Multisupport Region Order-Based Gradient Histogram), MRRID (Multisupport Region Rotation and Intensity Monotonic Invariant Descriptor) [19] and RATMIC (Resistant to Affine Transformation and Monotonic Intensity Change) [20] are all histogram-based features which implement intensity order in pooling strategy to achieve resistance to intensity changes. Ziegler *et al*. proposed LUCID (Locally Uniform Comparison Image Descriptor) [21] on the study of permutation distances and Kendall's Tau metric, which has a form of integer strings. It sorts the intensities of pixels and directly employs the order permutation as the image feature. In the development of image features, researchers are continuously aiming at designing description method of better precision and faster implementation.

B. Multi-class classifiers

Multi-class classification is a problem that we have to face in machine learning. Compared to binary classification, multi-class problems require more complex classifier structures or schemes. In most solutions, this problem decomposes trivially into a set of unlinked binary problems, which can be solved naturally using the existing techniques for binary classification. In addition, some classifiers are congenitally designed to deal with multiple classes. During the past decades, many algorithms were carried out with different inspirations, the outstanding ones of which have been applied in numerous practical fields of machine learning.

Support Vector Machine (SVM) proposed in the work of [22] has achieved great performance on 2-class classification problems. However, the original SVMs were designed only for binary classification, which does not satisfy the practical requirements. How to effectively extend SVM for multi-class classification is drawing researchers' attention. Currently there are two primary types of approaches for multi-class SVM [23]. One is by constructing and combining several binary classifiers while the other is by directly considering all data in one optimization formulation. The oldest implementation for multi-class SVM is probably the one-against-all method, which construct k SVM models for each class. And the ith class is trained with all of the instances in this class with positive labels and all the others with negative labels. Another major method is the one-against-one method with the idea first introduced in [24] and the first implementation on SVM in [25]. This method constructs $k(k-1)/2$ classifiers where each one is trained on data from two classes. A third algorithm is the directed acyclic graph SVM (DAGSVM) proposed in [26]. One advantage of using a DAG is some analysis of generalization can be established, which are not available in one-against-all and one-against-one methods yet. In addition, the test time of DAG method is less than the one-against-one method. Besides, methods by considering all data at once with a decomposition implementation can be found in [27]. Experiments and discussions on large problems show that one-against-one method and DAG method may be more suitable for practical use, and the latter one operates faster with an additional analysis result. A widely used implementation of SVM, LIBSVM (a library for SVM) released by Chih-Chuang Chang and Chih-Jen Lin [28], employs this method and provides probability estimates.

Artificial Neural Networks are computational models inspired by animal's central nervous systems that are capable of machine learning and pattern recognition. They are usually presented as systems of interconnected neurons which can compute values from inputs by feeding information through the network. Multi-layer Feedforward Neural Networks provide a natural extension to multi-class problem [29]. Instead of using one neuron in the output layer to produce binary output, N neurons could be employed to give out codes of multiple labels. The output codeword corresponding to each class can be chosen as follows [30]: For one-per-class coding, each output neuron is designated the task of identifying a given class, and the output code for that class should be 1 at this neuron and 0 for the others. When testing an unknown example, the neuron providing the maximum output is considered the class label for that example. For distributed output coding, each class is assigned an unique binary codeword from 0 to $2^N - 1$, where N represents the number of output neurons. When we test an unknown example, the output codeword is compared to each class codeword, and the nearest codeword is considered the wining class. Usually, the codeword distances are computed with Hamming distance, which measures the number of bit dissimilarity between two binary strings. It is obvious that the first way requires more neurons than the second one under the same circumstance of class numbers, nevertheless the first approach may have relatively better robustness. When the number of classes is high, the second method should be

an economical choice.

Decision Trees are a very powerful classification technique. Two widely known algorithms for building decision trees are Classification and Regression Trees [31] and ID3/C4.5 [32]. The tree splits training data based on the values of available features to produce a good generalization. The split decision is made at each node based on the feature that gives the maximum information gain. Each leaf node corresponds to a class label, so this algorithm can naturally handle multi-class classification problems. On the basis of Decision Trees, Random Forests were developed to gain better performance. Random Forests are an ensemble learning method for classification which employ multiple decision trees at training step and output the class that is the mode of classes output by individual trees. This algorithm was developed by Breiman and Cutler [33] as their trademark. Their method combines Breiman's bagging idea and the random selection of features introduced by Ho *et al*. [34], which enables constructing a collection of decision trees with controlled variation. Random Forest is a quite strong classifier that can handle multi-class problems well. It has been widely used in the applications of data-mining and object recognition.

C. Traffic sign recognition

Designing smarter vehicles which aims to minimize the number of driver-based wrong decisions or accidents has become a research hotspot in automotive technology nowadays [35]. As one of the most important issues, Traffic Sign Recognition (TSR) is developed to warn drivers about traffic signs and increase driving safety. In the future, these assistant driving systems may even take control of the vehicle instead of human being under some circumstances. Broad TSR includes the whole process from capturing continuous images of driving horizon, detection of traffic signs in captured images, recognition of detected signs to tracking of recognized signs, while the narrowly defined one just refers to the recognition step. TSR belongs to object detection and recognition problem, nevertheless has some unique characteristics. Traffic signs are one type of artificial objects, with fixed colors, shapes and patterns. Thus, techniques concerning color segmentation, feature extraction including shapes and patterns, and various classification methods are all involved in TSR tasks.

In consideration of reducing computational cost on classifying road signs, Barnes and Zelinsky adapted the fast radial symmetry detector to the image stream from a camera mounted in a car to eliminate almost all non-sign pixels, and applied normalized cross-correlation to classify the signs [36]. Their method is suitable for circular signs only, and achieves good performance on changes of lighting conditions. Bahlmann *et al*. designed a system for traffic sign detection, tracking and recognition using color, shape and motion information [4]. Signs are detected with a set of Haar wavelet features obtained from Ada-Boost training, once detected, they are tracked within a temporal information propagation framework followed by a classification performed using Bayesian generative modeling. Ruta *et al*. utilized novel image representation and discriminative feature selection algorithms in the traditional three-stage framework of TSR [6]. Baro *et al*. employed a boosted detector cascade trained with a novel evolutionary version of Adaboost in the detection of signs, which allows the use of large feature spaces. Moreover, classification is implemented by an Error-Correcting Output Code (ECOC) framework [2]. Zaklouta *et al*. proposed a real-time traffic sign recognition system in three stages, consisting of a segmentation, a detection and a classification phase [3]. They combine color enhancement with adaptive thresholds in sign region detection which is implemented by an efficient linear Support Vector Machine (SVM) with Histogram of Oriented Gradients (HOG) features. And tree classifiers including K-D Tree and Random Forest are used in the identification of sign contents.

For assessing the performance of current machine learning algorithms on TSR, Stallkamp *et al*. presented a publicly available traffic sign dataset with more than $50,000$ images of German road signs in 43 classes [37]. The data was considered in the second stage of the German Traffic Sign Recognition Benchmark held at IJCNN 2011 and a competition within numerous state-of-the-art algorithms was carried out on this dataset. Recorded results show that Convolutional Neural Networks (CNNs) won the first place with particularly high classification accuracy in the performance. Committee of CNNs algorithm even outperforms human performance, however, at a very large cost of computation. Besides, Linear Discriminant Analysis (LDA) on HOG feature also gains good results. This competition shows the great performance and potential of computer algorithms on TSR versus human.

III. The permutation-based local image descriptor

In this section, we depict a simple but effective local image descriptor called LIPID (local image permutation interval descriptor). It is a permutation on zone numbers of image pixel intensities, preserving good properties on monotonic brightness changes and timings. LIPID is developed on the basis of LUCID [21] yet it improves the robustness of LUCID meanwhile maintaining enough distinctiveness. The construction from permutation rather than bit sampling and comparison enables a desirable illumination invariance with no speed sacrifice.

A brief introduction of LUCID is given as follows: Let p_1 and p_2 denote two $n \times n$ image patches, and the computation of descriptor construction and matching with the generalized Hamming distance can be done in three lines of Matlab commands:

$$\begin{aligned} &[sortp1, order1] = \text{sort}(p1(:)); \\ &[sortp2, order2] = \text{sort}(p2(:)); \\ &distance = \text{sum}(order1 \ = order2); \end{aligned} \tag{1}$$

where $order_1$ and $order_2$ are the order permutation representations for p_1 and p_2 respectively.

From the construction of LUCID, it is clear that the whole process includes three key steps: vectorization of

image patch, sorting pixel intensities and recording order permutation. The idea of building LUCID is quite simple, and the permutation-based feature has brought in some unique and satisfying properties. However, its shortcoming is also obvious: LUCID is very sensitive because the order permutation can be easily disrupted by the intensity changes of individual pixels. We discover the direct use of order indices as descriptor leads to the unrobustness, and a feasible modification can be applied on descriptor building to improve the performance. Thus, we apply a interval division and zone number assignment on the order permutation and adopt the zone indices instead of order indices as our final feature.

Let p denote an image patch of gray scales and we first rearrange this patch into a vector. This vector is then sorted in ascending order according to the intensities of pixels, and the order indices which indicate the previous position of elements in the sorted vector are recorded. The above steps are quite similar to what is done in LUCID. And then, we construct our descriptor with the help of order indices instead of directly using them. The orderly elements in the sorted vector are divided into several zones, and all the elements in each interval is assigned a zone number. With the order indices indicating their original position in the patch vector, each element in the patch vector gains a zone number accordingly. Namely, each pixel is assigned a number according to which zone its intensity order belongs to within the patch. Each zone number reflects how bright a pixel is in the patch, with the measurement of an approximate position in the sorting permutation. Fig. 1 demonstrates how our feature is extracted with a simplified example. And the complete algorithm is as following:

1) Convert an image patch into grayscale one, and vectorize it to gain a vector V.
2) Sort V to obtain a sorted vector O and an order indices vector I.
3) Divide O into n zones and assign a zone number for each interval.
4) Tag each element in V with a corresponding zone number according to I which indicates the original positions.
5) Collect and arrange the zone numbers tagged on V to get the feature vector.

From the construction of LIPID descriptor, it is true that the two main parameters, the size of image patch and the number of dividing zones, should be chosen pairwise to enable an integer of each interval. Therefore, we choose powers of two to be the number of zones, and accordingly, multiples of zone number for the patch pixel numbers. Generally in point to point matching, the more zones we divide, the better accuracy we will achieve, meantime the less retrieved matching pairs we will get. And larger patch size helps improve representation accuracy, with an increase of feature vector dimension. As empirical results, 4, 8 or 16 should be proper choices of zone number, and an appropriate patch size will gain balance between accuracy and time and space consumption.

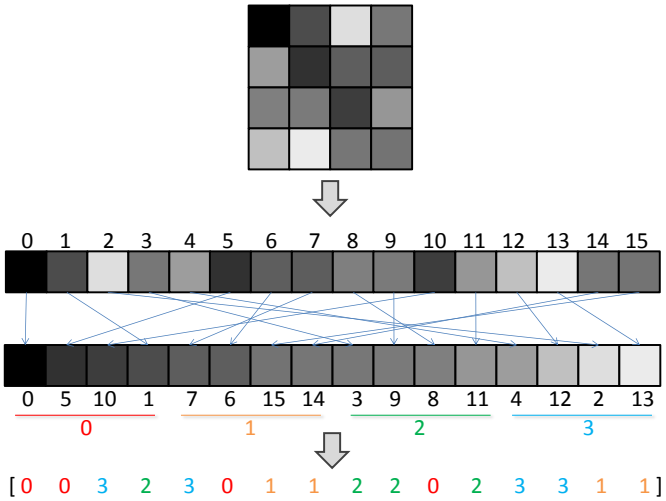

Fig. 1. A simplified example of constructing LIPID feature.

IV. Performance Evaluation

A. Experiments setup

Our proposed feature is tested on the German Traffic Sign Recognition Benchmark (GTSRB) dataset introduced by Stallkamp *et al* [37]. This dataset contains up to 26640 different images for training and more than ten thousand images for testing. These images were created from video captured while driving in Germany, which include 43 different categories of German traffic signs. The regions of interest (ROI) in both training and test images are also given together with class IDs for the sake of locating traffic signs in non-square images. Though image frequency of each class vary from 0.5% to 5.6% in the training set, an adequate amount of at least 150 images for one class guarantees the requirement of experiments. Fig. 2 shows a collection of random representatives of every category and the standard images of all categories are shown in Fig. 3. Another reason for choosing GTSRB as our experiment dataset is it provides several state-of-the-art pre-calculated feature sets which allow different scholars to conveniently run their tests. In the experiments we use raw images and our proposed features together with the pre-computed Haar-like, HOG and Hue histograms features this dataset offers to show a fair and verifiable result. Besides, LUCID and BRIEF as fixed-point feature representatives are also added for comparisons.

We employ Support Vector Machine (SVM), Neural Network (NN) and Random Forest to train the features and do the recognition respectively. On one hand, traffic sign recognition is a multi-class classification problem with unbalanced class frequencies, supervised learning should be an ideal choice to ensure the recognition accuracy as much as possible. On the other, the above-mentioned approaches have been validated to achieve good performances in many classification applications. Our experiments are implemented by Matlab R2009a on a PC with 2.93GHz i7 CPU and Windows 7 Professional 64-bit OS. Indeed, employment of

Fig. 2. Representatives of the 43 classes of traffic signs in GTSRB dataset.

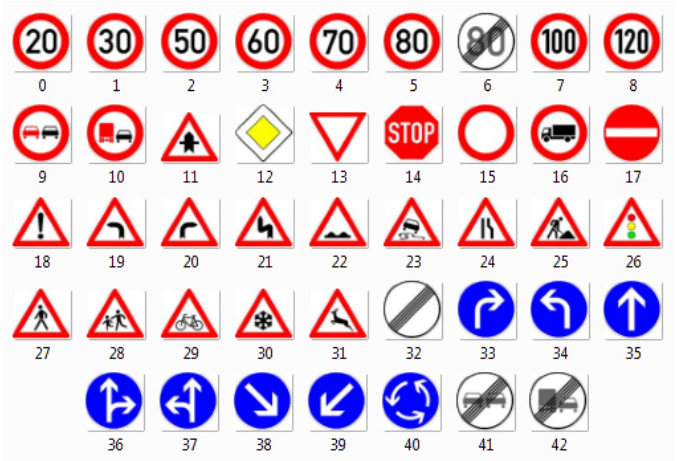

Fig. 3. All standard 43 signs in GTSRB dataset.

the whole dataset must be more convinced, however, due to the numbers of features and training methods and the limit of computing power, we have to strategically sample part of the training and test sets to reduce the tremendous time consumption in the experiments. Since the images of each category are all captured from a continuous video, the changes of adjacent frames are minor. If we sample a subset from each training category, it won't hurt the final result and conclusion. As for the test dataset, images are originally from different classes randomly. We only need to sample it at a same rate in order to keep the accordance of numbers between training and test dataset. On the basis of the above analysis, every 1 from 10 images are picked up in each class to form a subset of the original dataset, as thus, we obtain 2664 images for training and 1256 ones to do the tests.

B. Implementation details

We notice that the rich information mainly concentrates in the center of signs. Most signs have a border (see Fig. 3), and the patterns offering primary and discriminative information always locate in the center. If we don't specially use color information, the removal of borders will not result in a loss of data, moreover, the dimension of feature will be reduced doubtlessly. Since the pre-provided HOG and Haar-like features don't take colors into account, our proposed feature employs gray-scale images as well. Besides, the background in region of interest (ROI) is still quite busy for most images, and the disturbance of background cannot be neglected especially for those signs taking up less area in ROI such as triangle signs. In view of all the above, we design a center square patch as well as a cross-shape one containing 5 sub-patch for our feature extraction. Fig. 4 illustrates where our feature LIPID is extracted. The left figure shows the contours of all traffic sign shapes, including round, upward triangle, downward triangle, octagon and diamond, and the right one indicates the patch locations with the lines of dashes. The red dash square represents the center-located patch, while the magenta ones show the cross-shape 5 patches. Actually, the center square patch takes the center pixel as the only keypoint around which the LIPID feature is extracted, while the cross-shape patches use five keypoints located in the center, left, right, top and bottom.

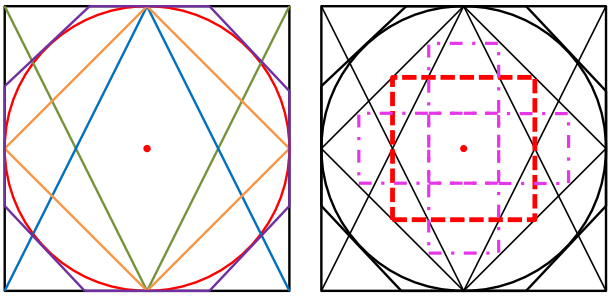

Fig. 4. All shape contours of traffic signs in GTSRB dataset and the image patches we apply for LIPID feature extraction.

In accordance with the pre-computed features, we also rescale all the images into a standard size of 40×40. Noting that the feature dimension and number of zones should be chosen pairwise, the size of patch (number of sampling points) must corresponds to a given zone number. For the one keypoint center patch, the side length is set to be 20 when the zone number is 8, producing a feature with dimension of 400. And for the 5 keypoints design, we choose either 4 or 8 zones, accordingly, the patch size is 10×10 and 12×12. The former one's dimension is 500, while the latter one has a dimension of 720 with a slight overlap between the center patch and the others. To facilitate brief expressions in the following text, we name the above three features LIPID1-8, LIPID5-4 and LIPID5-8. For LUCID, we use the center one patch of 20×20, and BRIEF employs a 256 length of bit comparison.

Because the benchmark dataset provides more than one selections of parameters for most pre-calculated features, we need to choose one or several representative groups for each kind. The first 1568 dimension HOG feature, and the combination of horizontal and vertical edge 6×6 haar-like feature (1024 dimension for each orientation) are selected. Histogram of hue feature just has one choice, and we also supplement resized raw images as a contrast.

C. Training and test performance

In this section, we start the experiments with Support Vector Machine, followed by Neural Network and Random Forest. Recognition performance using each learning method is given, as well as the timings of training and test process.

Support Vector Machine (SVM) is a supervised learning model for two-group classification problems [22]. To reduce a multi-class problem into multiple binary classification problems, SVM can be modified for multi-class classifications. An efficient implementation of SVM is LIBSVM, a library for SVM released by Chih-Chuang Chang and Chih-Jen Lin [28]. It offers multi-class classification, cross validation for model selection and probability estimates, and also supports various kernels and language interfaces. We use its matlab interface with all default parameters, expect for c (cost) and g (gamma in kernel function). To find an optimum group of c and g, we use cross validation to search for it in a given range (both c and g in the range of $(2^{-8},2^{8})$). Table I gives the results of searching best c and g. By the limit of running time and pre-defined searching range, these given parameters may be only local optimum values and far from the ones that achieve best performance. It is explicit that this search procedure takes a considerably long time, which does depend on the dimension of feature. Fig. 5 and Fig. 6 shows the recognition rates and timings of different features, each is obtained with the respective best parameters shown in Table I. It is obvious that traffic sign recognition with HOG exhibits a best performance up to 92%, and BRIEF, LIPID5-4 and LIPID1-8 show accectable results which are much better than the others. Haar-like feature, hue histogram feature, LIPID5-8 and LUCID perform similarly as raw images, which show a weak discrimination with SVM. As for time consumption, BRIEF has the minimum of time for its shortest feature vector. LIPID shows comparable training time with HOG and less test time than HOG. Raw images and Haar-like feature have the most time elapse yet produce worst results.

TABLE I
SEARCH FOR OPTIMUM PARAMETERS OF SVM.

Feature	Dimension	Best c	Best g	Time(sec)
Raw image	1600	1	0.1	71707.7
Haar-like	2048	1	0.1	98341.8
HOG	1568	16	0.0118	59937.6
Hue-hist	256	1	0.1	10069.1
LIPID1-8	400	1.7411	0.0039	19620.3
LIPID5-4	500	3.0314	0.0039	24657.5
LIPID5-8	720	1	0.1	35712.1
LUCID	400	1	0.1	19272.0
BRIEF	256	16	0.0068	11639.5

Neural Network is then employed in our experiments as well. Though there have been numerous types of networks and various methods for training, we choose a very basic one that has been widely used for its reasonable structure and operating speed. Here, a two-layer feed-forward network with sigmoid hidden and output neurons is selected, and the network is trained with scaled conjugate gradient

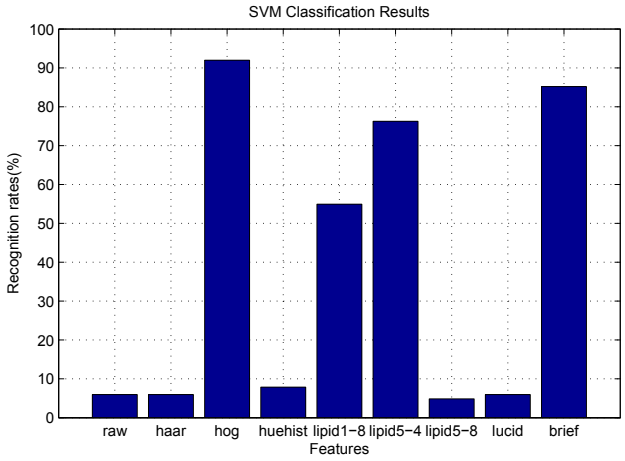

Fig. 5. Recognition rates when using SVM.

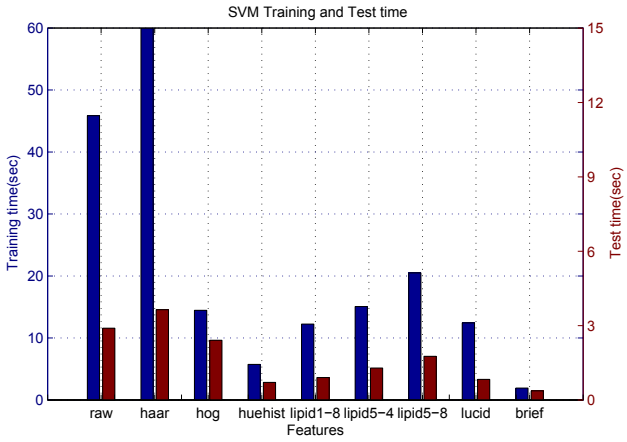

Fig. 6. SVM Timings. Blue and red bars indicate training and test time respectively.

backpropagation. Training set is randomly divided into a training subset of 50% samples, a validation subset of 25% and a testing subset of 25% as well. Fig. 7 shows the training and test results with this two-layer feed-forward network. As the number of hidden neurons varies, accuracy rate alters. Overall, HOG and LIPID1-8 reveal the best performance, and all LIPID features have a better accuracy than Haar-like, LUCID and hue histogram features. Timings of Neural Network is given by Fig. 8. It can be seen that LIPID features tend to adopt more hidden neurons in the construction of network. Though the training time of LIPID grows remarkably with the increase of vector dimension, the test time remains on a relatively low level.

Random Forest is an ensemble learning method for classification. It operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes output by individual trees. Random Forest is actually a framework rather than a particular model. With the integration of weak classifiers, this framework becomes strong enough to solve multi-class classification problems. A basic parameter of Random Forest is the number of decision

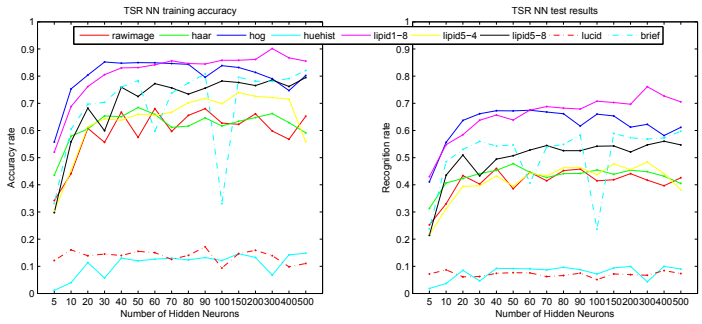

Fig. 7. Traffic sign recognition accuracy rates with a two-layer feed-forward network. The left subfigure shows the output results of trained net with the input of training dataset, which reflects the training accuracy, and the right one shows the recognition results of the test data.

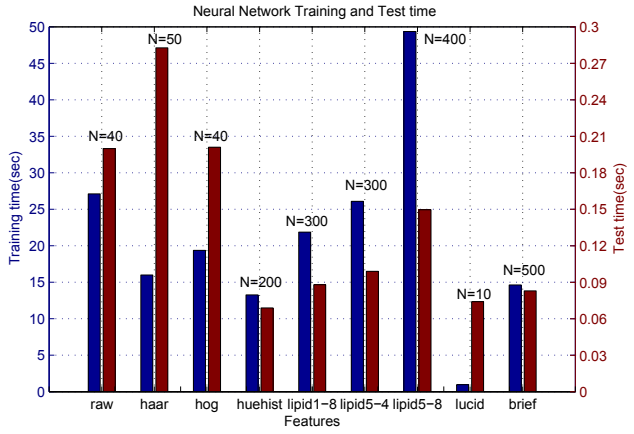

Fig. 8. Neural Network training and test time on different features. Training time is presented by blue bars while test time uses brownish red ones. Results are computed under respective number of neurons that performs best recognition rate, which is also marked above each bar.

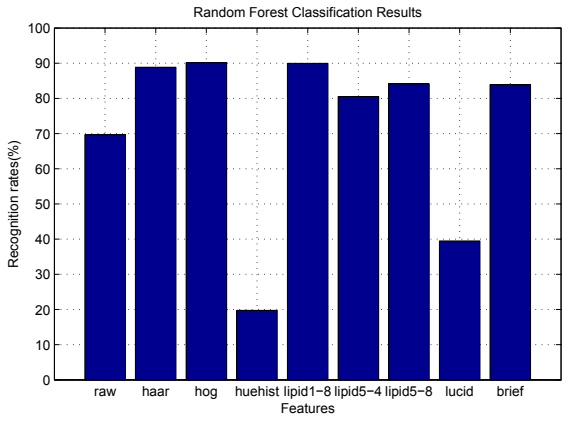

Fig. 9. Random Forest recognition rates on GTSRB.

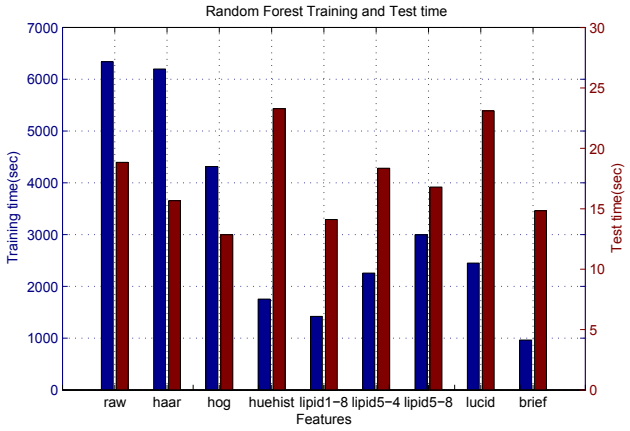

Fig. 10. Random Forest training and test time on different features. Training time is presented by blue bars with scale axis on the left, while test time uses brownish red bars with scale axis on the right.

trees constructed in the framework for predicting responses. In our experiments, we construct 100 decision trees in the Random Forest. Fig. 9 shows the recognition results using Random Forest. Both rates of HOG and LIPID1-8 reach up to 90%, and all the others exceed raw images except for hue histogram and LUCID. Fig. 10 demonstrates the timings using Random Forest for traffic sign recognition. Similarly, training and test time is shown in different colors and scales in order to be drawn in one figure. In general, BRIEF, LIPID and HOG seem to require less time in the overall process.

The real-time performance is a vital requirement in traffic sign recognition applications. Feature extraction of candidate signs should be of low complexity and high computation speed. On the basis of this demand, we implement timing tests on extraction of different features. Tests are implemented on all images of test dataset and an average time consumption of feature extraction per image is given in Table II, in which it is obvious that LIPID presents the fastest speed. Haar-like feature doesn't perform as fast as we imagine because the relatively small size of test images couldn't make use of the advantage of integral image. Besides, Haar-like feature has multiple templates for different orientations and feature types, so the dimension of it is larger than the others. Actually, computation of LIPID can even be optimized in the sorting step, using a non-comparison sorting algorithm to speed up. With a stable non-comparison sorting for integers such as Pigeonhole Sorting [38], we're able to achieve linear time efficiency in this key step in our method. In general, LIPID outperforms the other features on timing tests, showing a great potential in real-time applications.

TABLE II
TIMINGS OF FEATURE EXTRACTION.

Feature	Haar	HOG	Huehist	LIPID	LUCID	BRIEF
Time(ms)	6.08	4.35	0.40	0.34	0.37	0.36

From the confusion matrix of our experiments, we find that the recognition of signs with large monochrome in the center such as sign No.12 in Fig. 3 is the weakness shown with larger frequency of our feature. On account of neglecting color information and sampling merely in the center, this shortcoming is clear yet promising to be improved in the future work. To sum up, our proposed descriptor performs competitively with HOG and notably better than the others

in experiments of traffic sign recognition on GTSRB dataset, and outperforms all the involved features in computing speed tests.

V. Conclusions

In this paper, we present a permutation-based local image feature and implement traffic sign recognition on German Traffic Sign Recognition Benchmark dataset. With an interval division and zone number assignment on sorted pixel intensities, this feature LIPID shows a good distinctiveness and robustness at a low computation requirement. Compared to other pre-calculated and fixed-point features involved in the experiment, our feature almost achieves the best performance meanwhile reveals the least time cost. In the application of traffic sign recognition, our feature satisfies the demands of illumination changes and real-time processing, and offers an alternative to state-of-the-art features. Although the confusion cases occur more frequently on signs with large monochrome in the center, classification by color beforehand or designing better sampling method will help to eliminate missort. In the future work, this feature will be further developed to be resistant to image occlusion, noise and distortion that frequently occur in traffic sign recognition. Combined with new classification methods and schemes, it is promising to achieve great performance with affordable computation requirements. Moreover, it is expected to be applied in traffic sign detection and other applications requiring illumination variance and fast speed.

References

[1] U. Handmann, T. Kalinke, C. Tzomakas, M. Werner, and W. Seelen, "An image processing system for driver assistance," *Image and Vision Computing*, vol. 18, no. 5, pp. 367–376, 2000.

[2] X. Baró, S. Escalera, J. Vitrià, O. Pujol, and P. Radeva, "Traffic sign recognition using evolutionary adaboost detection and forest-ecoc classification," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 10, no. 1, pp. 113–126, 2009.

[3] F. Zaklouta and B. Stanciulescu, "Real-time traffic sign recognition in three stages," *Robotics and Autonomous Systems*, 2012.

[4] C. Bahlmann, Y. Zhu, V. Ramesh, M. Pellkofer, and T. Koehler, "A system for traffic sign detection, tracking, and recognition using color, shape, and motion information," in *Intelligent Vehicles Symposium, 2005. Proceedings. IEEE*. IEEE, 2005, pp. 255–260.

[5] Y. Xie, L.-f. Liu, C.-h. Li, and Y.-y. Qu, "Unifying visual saliency with hog feature learning for traffic sign detection," in *Intelligent Vehicles Symposium, 2009 IEEE*. IEEE, 2009, pp. 24–29.

[6] A. Ruta, Y. Li, and X. Liu, "Real-time traffic sign recognition from video by class-specific discriminative features," *Pattern Recognition*, vol. 43, no. 1, pp. 416–430, 2010.

[7] Z. Zheng, H. Zhang, B. Wang, and Z. Gao, "Robust traffic sign recognition and tracking for advanced driver assistance systems," in *Intelligent Transportation Systems (ITSC), 2012 15th International IEEE Conference on*. IEEE, 2012, pp. 704–709.

[8] D. G. Lowe, "Object recognition from local scale-invariant features," in *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, vol. 2. Ieee, 1999, pp. 1150–1157.

[9] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *Computer Vision–ECCV 2006*. Springer, 2006, pp. 404–417.

[10] C. P. Papageorgiou, M. Oren, and T. Poggio, "A general framework for object detection," in *Computer Vision, 1998. Sixth International Conference on*. IEEE, 1998, pp. 555–562.

[11] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, vol. 1. IEEE, 2001, pp. I–511.

[12] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1. IEEE, 2005, pp. 886–893.

[13] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," in *Computer Vision–ECCV 2010*. Springer, 2010, pp. 778–792.

[14] J. Liu and X. Liang, "I-brief: A fast feature point descriptor with more robust features," in *Signal-Image Technology and Internet-Based Systems (SITIS), 2011 Seventh International Conference on*. IEEE, 2011, pp. 322–328.

[15] T. Trzcinski and V. Lepetit, "Efficient discriminative projections for compact binary descriptors," in *Computer Vision–ECCV 2012*. Springer, 2012, pp. 228–242.

[16] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: an efficient alternative to sift or surf," in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 2564–2571.

[17] S. Leutenegger, M. Chli, and R. Y. Siegwart, "Brisk: Binary robust invariant scalable keypoints," in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 2548–2555.

[18] A. Alahi, R. Ortiz, and P. Vandergheynst, "Freak: Fast retina keypoint," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 510–517.

[19] B. Fan, F. Wu, and Z. Hu, "Rotationally invariant descriptors using intensity order pooling," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 34, no. 10, pp. 2031–2045, 2012.

[20] Z. Huang, W. Kang, Q. Wu, and X. Chen, "A new descriptor resistant to affine transformation and monotonic intensity change," *Computer Vision and Image Understanding*, 2013.

[21] A. Ziegler, E. Christiansen, D. Kriegman, and S. J. Belongie, "Locally uniform comparison image descriptor," in *Advances in Neural Information Processing Systems*, 2012, pp. 1–9.

[22] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[23] C.-W. Hsu and C.-J. Lin, "A comparison of methods for multiclass support vector machines," *Neural Networks, IEEE Transactions on*, vol. 13, no. 2, pp. 415–425, 2002.

[24] S. Knerr, L. Personnaz, and G. Dreyfus, "Single-layer learning revisited: a stepwise procedure for building and training a neural network," in *Neurocomputing*. Springer, 1990, pp. 41–50.

[25] U. H.-G. Kreßel, "Pairwise classification and support vector machines," in *Advances in kernel methods*. MIT Press, 1999, pp. 255–268.

[26] J. C. Platt, N. Cristianini, and J. Shawe-Taylor, "Large margin dags for multiclass classification." in *NIPS*, vol. 12, 1999, pp. 547–553.

[27] J. Weston and C. Watkins, "Multi-class support vector machines," Citeseer, Tech. Rep., 1998.

[28] C.-C. Chang and C.-J. Lin, "Libsvm: a library for support vector machines," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 3, p. 27, 2011.

[29] M. Aly, "Survey on multiclass classification methods," *Neural Netw*, pp. 1–9, 2005.

[30] T. G. Dietterich and G. Bakiri, "Solving multiclass learning problems via error-correcting output codes," *arXiv preprint cs/9501101*, 1995.

[31] L. Breiman, *Classification and regression trees*. CRC press, 1993.

[32] J. R. Quinlan, *C4. 5: programs for machine learning*. Morgan kaufmann, 1993, vol. 1.

[33] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[34] T. K. Ho, "The random subspace method for constructing decision forests," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 20, no. 8, pp. 832–844, 1998.

[35] Aydin, "Traffic sign recognition," Ph.D. dissertation, MIDDLE EAST TECHNICAL UNIVERSITY, 2009.

[36] N. Barnes and A. Zelinsky, "Real-time radial symmetry for speed sign detection," in *Intelligent Vehicles Symposium, 2004 IEEE*. IEEE, 2004, pp. 566–571.

[37] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition," *Neural networks*, vol. 32, pp. 323–332, 2012.

[38] P. E. Black, *Dictionary of algorithms and data structures*. National Institute of Standards and Technology, 2004.