Traffic Sign Recognition – How far are we from the solution?

Markus Mathias, Radu Timofte, Rodrigo Benenson, and Luc Van Gool

***Abstract*— Traffic sign recognition has been a recurring application domain for visual objects detection. The public datasets have only recently reached large enough size and variety to enable proper empirical studies. We revisit the topic by showing how modern methods perform on two large detection and classification datasets (thousand of images, tens of categories) captured in Belgium and Germany.**

We show that, without any application specific modification, existing methods for pedestrian detection, and for digit and face classification; can reach performances in the range of $95\% \sim 99\%$ of the perfect solution.

We show detailed experiments and discuss the trade-off of different options. Our top performing methods use modern variants of HOG features for detection, and sparse representations for classification.

I. Introduction

TRAFFIC SIGN RECOGNITION (TSR) gets considerable interest lately. The interest is driven by the market for intelligent applications such as autonomous driving [1], advanced driver assistance systems (ADAS) [2], mobile mapping [3], and the recent releases of larger traffic signs datasets such as Belgian [3] or German [4] datasets.

TSR covers two problems: traffic sign detection (TSD) and traffic sign classification (TSC). TSD is meant for the accurate localization of traffic signs in the image space, while TSC handles the labeling of such detections into specific traffic sign types or subcategories.

For TSD and TSC numerous approaches have been developed. A recent survey of such methods and existing datasets is given in [5].

A. Contributions

The contribution of this work is three-fold:

- We show that top performance can be reached using existing approaches for pedestrian detection [6] and face recognition [7], [8], without the need to encode traffic sign specific information.
- We provide an extensive evaluation on two large Belgian and German traffic sign benchmarks for both detection and classification.
- We show on these datasets detection and classification results in the range $95\% \sim 99\%$ of the perfect solution.

Markus Mathias and Radu Timofte contributed equally to this work.

Markus Mathias, Radu Timofte, and Luc Van Gool are with VISICS, ESAT-PSI / iMinds, University of Leuven, Belgium (email: {first-name.lastname}@esat.kuleuven.be).

Rodrigo Benenson is with Max-Planck-Institut für Informatik, Saarbrücken, Germany (email: benenson@mpi-inf.mpg.de).

Luc Van Gool is also with Computer Vision Lab, D-ITET, ETH Zurich, Switzerland.

Fig. 1. Top Image: a typical image from BTSD together with the detected traffic signs. Bottom images: Each block shows true positives from the 3 superclasses (rows) spotted by our `ChnFtrs` detector. From left to right, the first block contains true positives from the GTSD, the second from BTSD, and the third examples of false positives from either dataset.

II. Traffic Sign Datasets

In this paper we evaluate traffic sign detection and classification on four datasets: the German Traffic Sign Detection Benchmark (GTSD) [9][1], the German Traffic Sign Recognition Benchmark (GTSC) [4] and the Belgium Traffic Sign Dataset with its split for Detection (BTSD) [3] and for Classification (BTSC) [10][2].

The choice of the datasets is motivated by their large amount of annotations, diversity of the content and classes, the availability of a split for benchmarking separately TSD and TSC. The GTSD recently was subject to a competition, making it easier to compare various approaches. No other dataset surveyed in [5] has a comparable size or number of classes.

To assess detection performance we use GTSD and BTSD. Both are split in 3 main categories (or superclasses) based on their shape and color:

(M) mandatory: round, blue inner, white symbols.
(D) danger: (up) triangular, white inner, red rim.
(P) prohibitory: round, white inner, red rim.

Table I shows the training/testing split and the number of traffic signs per superclass for the detection task. GTSD

[1] `http://benchmark.ini.rub.de`
[2] `http://homes.esat.kuleuven.be/~rtimofte`

TABLE I

DETECTION BENCHMARKS WITH (M)ANDATORY, (D)ANGER, AND (P)ROHIBITORY SUPERCLASSES.

	Training				Testing			
	images	annotations			images	annotations		
GTSD	600	M 113	D 154	P 370	300	M 50	D 62	P 161
BTSD	5 905 +16 045	M 1 026	D 765	P 891	3 101 +583	M 570	D 795	P 580

TABLE II

CLASSIFICATION BENCHMARKS, NUMBER OF INSTANCES FOR TRAINING AND TESTING

	Training	Testing	Classes	Diversity
GTSC	26 640 + 12 569	12 630	43	$\sim$ 30 images/sign
BTSC	4 591	2 534	62	$\sim$ 3 images/sign

provides fully annotated images for testing. The BTSD dataset provides only partially annotated positive images, thus additional 583 images without traffic signs are used for assessing the false positive rates.

For classification GTSC provides annotations for 43 different classes, BTSC distinguishes 63 classes. The mentioned detection superclasses cover most of the classification classes, but not all of them.

Table II shows the number of samples in Belgian and German datasets (BTSC and GTSC) used to benchmark our classification approaches. Both come with their own split into training/validation and testing data. GTSC has more image samples than BTSC, but a smaller number of traffic sign classes, and a fewer number of physically distinct traffic signs captured by these images. GTSC contains around 30 images per each sign tracked into the video sequence, while BTSC contains on average 3 images per each sign as seen by different cameras at 3 different time moments.

There is no direct mapping between the class labels in BTSC and GTSC. For instance, the speed limit signs form a single class in BTSC, while they spans several classes in GTSC.

In section III we describe the detection methods evaluated in section IV. Section V describes the classifications methods we evaluate in section VI. Finally in section VII we present the results obtained when joining detection and classification, and offer some concluding remarks in section VIII.

III. TRAFFIC SIGN DETECTION

Traffic sign detection is a classic instance of rigid object detection. It is currently accepted that histogram of oriented gradients (HOG) is an effective way to capture shape information. The integral channel features detector first introduced by Dollar et al. [11], builds upon these features to provide excellent quality for the task of pedestrian detection, significantly improving over the traditional HOG+linear SVM combination [12]. More recently, variants of the integral channel features detector have shown to reach high speed (50 fps) [13], [14], and top quality, improving over most existing methods for pedestrian detection [6] (while still using a single rigid template per candidate detection window).

In this paper we show we can use the integral channels features approach to reach top performance for traffic signs detection, without any application specific changes or special features.

A. Integral Channel Features Classifier

The integral channel features classifier (`ChnFtrs`) is a family of boosted classifiers based on discrete Adaboost. We use the same basic setup as described in [11]. The weak learners used for boosting are depth-2 decision trees, where each node is a simple decision stump, defined by rectangular region, a channel, and a threshold. As channels we use the proposed 6 orientation channels, 1 gradient magnitude channel, and the 3 channels of the LUV color space (see Figure 2 first row). Our final classifier is a weighted linear combination of boosted depth-2 decision trees.

B. Detection using different aspect ratios

The training and testing annotation bounding boxes are tight around the traffic signs. The bounding boxes of traffic signs rotated around the gravity axis have a width to height ratio smaller than the frontal ones. Since the correct perspective rectification is not available, for training we align the example by stretching the annotated traffic sign to our squared model window. During testing, besides evaluating the model at different scales, we also consider different aspect ratios. In the baseline setup, we use 50 scales and 5 ratios resulting in 250 image sizes.

C. Training setup

While the two datasets are different in the number of traffic signs and their sizes, we trained all models using the same setup. We perform in total 4 rounds of training with increasing numbers of weak learners (50, 100, 200 and 400). Each round adds 2 000 hard negatives to the training dataset, randomly sampled in round 1, and via bootstrapping in further rounds. We use a model size of 56×56 pixels.

As the number of possible stumps is large (all possible rectangles in 10 different channels), we train the detector by using a randomly sampled subset. The feature pool is generated by randomly sampling 80 000 candidate rectangles. Figure 2 shows the model of the "danger" superclass trained on the GTSD. Training takes roughly 45 minutes per superclass.

D. Testing setup

We perform testing using our own GPU enabled implementation of the integral channel features detector. Our code is based on the open source release of [14]. Each detector template is applied exhaustively at all image positions in a scale space of 50 scales and 5 ratios. We set the scale space search range according to the specifications of the datasets (on GTSD 16×16 to 128×128 pixels, on BTSD 16×16 to 256×256 pixels). We search for traffic signs from ratio 0.8 to 1.2 ($^{\text{width}}/_{\text{height}}$).

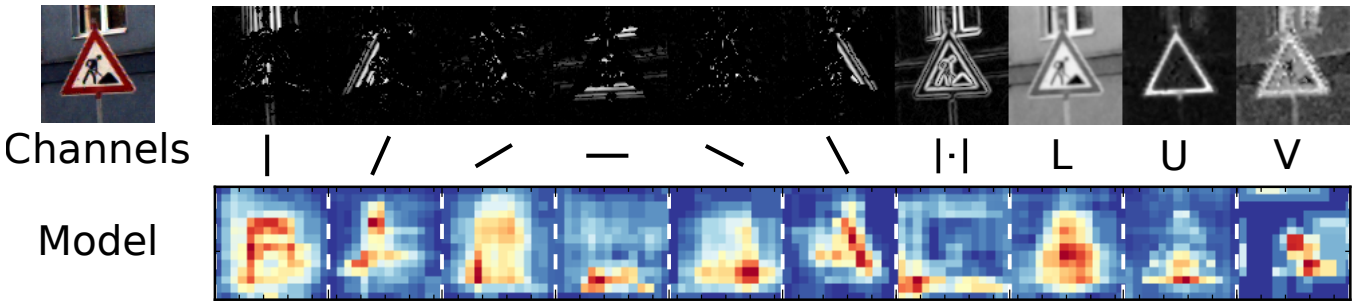


Fig. 2. Features picked during training in the 10 feature channels for the "danger" class. From left to right, 6 orientation channels, gradient magnitude, L channel, U channel, V channel. (Heatmap normalized per channel)

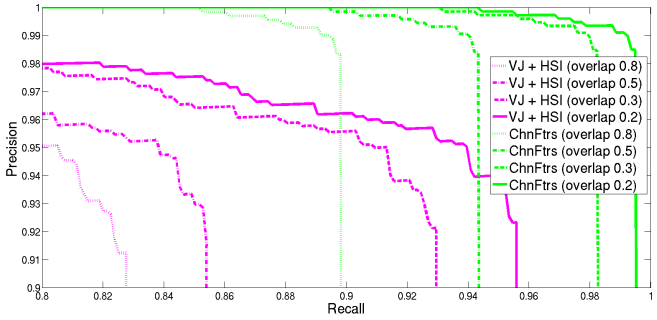


Fig. 3. Precision/Recall curves on BTSD for `ChnFtrs` and `VJ + HSI` detectors, prohibitory super-class at different overlap ratios.

IV. Detection Experiments

Table III summarizes the detection performance by comparing the area under curve (AUC) of different detectors on GTSD and BTSD on all three superclasses. We trained and evaluated the proposed detector in two different configurations. "Ours Baseline" is trained using the setting described in preceeding sections. By using these reasonable settings, we are able to train our detector fast (45 minutes per superclass) and the testing setup is very similar to the one described in [14], theoretically allowing to use the same approach for high speed detections. Currently, the detector runs at around 2.5 Hz on an Intel Core i7 870 with a Nvidia GeForce GTX 470 GPU.

GTSD: The second configuration ("Ours Competition") is optimized to reach the highest possible detection quality for the GTSD competition (at expenses of training and testing speed). The sliding window searches over 100 scales and 15 ratios and runs at 0.35 Hz. Further we followed the recommendations of [6], we globally normalize training and testing images (illumination normalization); for the mandatory class we additionally trained a multi-scale model to reach our best competition result (48×48 and 96×96 pixels).

The method "Ours Competition" reaches perfect detection scores in two of the three classes (one other competitor matches that performance, see Table III). The average overlap of our detections with the ground truth annotations lies above 87% for all superclasses.

BTSD: In the BTSD dataset our detector also shows good localization of the bounding boxes (see Figure 3), still reaching 89.8 AUC for an required overlap of 80%. The jumps in recall can mostly be explained by data mislabeling. Signs like the "Zone 30" sign, as shown in Figure 4(b), belong to the "danger" class, but are annotated as a whole and not only the circular red sign (that our classifier detects), this creates artificial false positives. The lower performance obtained when evaluating a detector trained on one dataset and tested on the other shows that there is a significant dataset bias (see Table III).

Figure 4 shows further peculiarities of the datasets. The blue sign showing bikes and/or pedestrians are not considered part of the "mandatory" class, despite corresponding to the semantic category, and thus counted as false positives by our detector (Figure 4(c)). The BTSD dataset contains challenging samples like strong perspective views, occlusions, rotated and damaged traffic signs (Figure 4(a)). The few traffic signs that our detector misses are mostly instances of the aforementioned categories.

The `VJ+HSI` detector (in figure 3) was introduced in [3], used in [2], and serves as a reference point. It uses a segmentation step based on a learned set of color-based thresholding methods which rapidly prune the regions of interest, leaving an average of 3 thousands candidates per 2 megapixels image. The regions of interest are the input for a Viola-Jones cascade trained using Haar-like features extracted on HSI color channels. We train over BTSD one specialized cascade per superclass (as in [3]), no traffic sign is used as negative.

V. Traffic Sign Classification

The classification of traffic signs is a classic case of supervised rigid objects classification. Face and digit image classification are closely related tasks since these two categories usually present limited variation in pose and appearance. Many of the methods used on these two domains also apply to traffic sign classification.

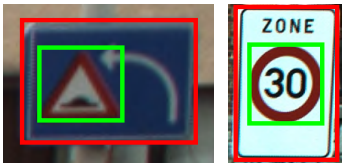
Our base pipeline consist of three stages: features extraction, dimensionality reduction, and classification. Each of them is detailed in the following subsections. We will consider a large number of feature representations, dimensionality reduction techniques, and classification methods that had success for digits and faces. Section VI presents empirical

TABLE III

DETECTION RESULTS ON GTSD AND BTSD BENCHMARKS FOR 0.5 OVERLAP (AUC IN [%]).

Team	Method	GTSD			BTSD		
		M	D	P	M	D	P
Ours Baseline	`ChnFtrs`, trained on GTSD	91.91	100	99.34	92.55	95.76	84.60
Ours Baseline	`ChnFtrs`, trained on BTSD	76.38	78.5	91.06	97.96	97.40	94.44
Ours Competition	`ChnFtrs`, trained on GTSD	*96.98*	*100.00*	*100.00*	94.79	96.40	86.51
[3]	`VJ+HSI`, trained on BTSD	61.12	79.43	72.60	92.32	95.91	84.27
wgy@HIT501[9]	HOG+LDA+SVM	100.00	99.91	100.00	–	–	–
BolognaCVLab[9]	MSER+HOG+SVM	95.76	98.72	99.98	–	–	–
LITS1[9]	HOG+SVM	92.00	98.85	100.00	–	–	–
wff[9]	HOG+CNN	97.62	99.73	−−	–	–	–

(a) difficult conditions



(b) unexpected annotations: red box refers to annotation, but the detector usually detects the greeen box.

(c) unclear class: these signs do not belong to the "prohibitory" class

Fig. 4. Failure cases due to: difficult conditions, unexpected bounding box annotations or unclear class belongings.

evaluations of the different combinations and describe the trade-offs of selected combinations.

A. Features extraction

Since traffic signs are designed to be of use for color blind people, the most discriminative features for classification are the inner pattern and the shape, the color is less important. In preliminary experiments we tested with color-dependent features but the results were unsatisfactory when compared with the grayscale based features and came at the price of increased computational cost. Color matters more for detection than for classification.

In our classification experiments we consider the following features:

- **I**: grayscale values of the cropped traffic sign images rescaled to 28×28 pixels. The I features are 784-dimensional.
- **PI**: the pyramid of histograms of oriented gradients (HOG) features with their optimal parameter settings as in [15], where top results were achieved for handwritten digit and face classification. PI provides a 2172-dimensional descriptor.
- **HOG1, HOG2, HOG3** : HOG features as precomputed for GTSC [4]. Three settings are provided, the difference among them is given by the number of HOG cells and the extraction grid. HOG1 and HOG2 are 1568-dimensional, while HOG3 is 2916-dimensional.

B. Dimensionality Reduction

Speed-wise, it is beneficial to reduce the dimensionality of the features. We consider LDA for discriminatively projecting the original feature representations to lower manifolds. We also inspect using SRLP and INNLP that provide embeddings which reveals different structural affinities in the data than the direct Euclidean distances.

All the features are l_2-norm normalized to sum to 1, before and after projection or before classification.

In the following we shortly review the dimensionality reduction techniques used. The reader is referred to the original works for more details.

1) Linear Discriminant Analysis (LDA) [16], [17]: is an embedding technique, which maximizes the inter-class variance while minimizing the intra-class variance. The LDA projection thus tries to best discriminate among classes. The solution can be obtained solving an eigenvector problem. By construction, LDA can lead to an embedding space with a number of dimensions less than the number of classes.

2) Sparse Representation based Linear Projection (SRLP) [10]: is a variant of Locality Preserving Projections (LPP) [18]. LPP itself is a linear approximation of the nonlinear Laplacian Eigenmap [19] method aiming at preserving the local affinities from the original space into the embedding. The algorithmic procedure employs an adjacency graph construction, setting edge weights, and finally solving a generalized eigenvector problem formulation. SRLP differs from LPP in that it preserves the weights from the sparse representation (see Section V-C.2) as affinity measures in the linear embedding space.

3) Iterative Nearest Neighbors based Linear Projection (INNLP) [8]: is another LPP variant and goes similarly to SRLP, the INN coefficients (see Section V-C.3) replacing those from the sparse representations in the graph construction.

In the experiments of section VI we consider the following combinations:

LDA I, LDA PI, LDA HOGx : Linear Discriminant Analysis (LDA) projections of the original image feature representation (I, PI, HOG1, HOG2, or HOG3). Note that we use regularized LDA [20] (as in [10]).

SRLP I, SRLP PI, SRLP HOGx : Sparse Representation-based Linear Projections (SRLP) of the original image feature representation. We use the regularized version of SRLP as introduced by [10] with supervised learning, that is, in training, the sparse representations are computed over samples sharing the same class with the projected one.

INNLP I, INNLP PI, INNLP HOGx : Iterative Nearest Neighbors-based Linear Projections (INNLP) of the original image feature representation. We use the supervised and regularized settings as originally introduced by [8].

Merged features : combining different features is found beneficial, and we use here the direct merging of the feature representations as concatenation of l_2 normalized compounding features.

C. Classification

We consider the following classification methods for our experiments:

*1) Nearest Neighbor Classifier (***NN***):* picks the label of the training sample which provides the minimum least squares error to the query.

The following two classifiers use also the least squares but are not constrained to one training sample, they linearly combine all the training samples in a least squares sense. One is using the regularization to promote sparsity of the coefficients (SRC), the other imposes the weights and searches for the best combination (INNC).

*2) Sparse Representation-based Classifier (***SRC***) [7]:* uses the so-called sparse representation (SR) which aims at minimizing the number of non-zeros coefficients in the linear decomposition of a query sample over a set of training samples. This, in practice, is obtained solving an l_1-regularized least squares formulation. In all our experiments, we use the Feature Sign algorithm [21] and fix its regulatory parameter to 0.05.

The Sparse Representation-based Classifier (SRC) [7] decision is taken based on the corresponding residuals to each class training samples. For speed, we directly use the corresponding class coefficients in magnitude from the representation, as in [10].

*3) Iterative Nearest Neighbors Classifier (***INNC***) [8]:* is based on a recently introduced (sparse) representation, the Iterative Nearest Neighbors (INN) representation, obtained by approximately solving a constrained least squares formulation. The coefficients from the INN representation are imposed and sum up to 1, and their decaying rate is controlled by a regulatory parameter, $\lambda \in (0, 1)$. We keep the settings from [8] and set $\lambda = 0.05$ which leads to a number of NN iterations, $K = 62$, for the INN procedure. For an input query, INNC sums up the coefficients from the INN representation corresponding to each class and assigns the query to the class with the maximum such sum.

4) Support Vector Machines Classifiers: Support Vector Machines (SVM) are a very popular technique for out of the box classification [22], [23]. The kernels $k(\mathbf{x}, \mathbf{y})$ we use here are the Linear - $\mathbf{x} \cdot \mathbf{y}$, Intersection Kernel- $\min(\mathbf{x}, \mathbf{y})$, Polynomial - $(\mathbf{x} \cdot \mathbf{y} + 1)^5$, and Radial Basis Function $\exp(-\|\mathbf{x} - \mathbf{y}\|^2)$. The classifiers are named **LSVM**, **IKSVM**, **POLYSVM**, and **RBFSVM**, accordingly.

We train one-vs-all classifiers using LIBSVM [24] (with parameter $C = 10$) and LIBLINEAR [25] (with parameters $C = 10$, $B = 10$). As in [15], the test sample is associated with the class with the highest posterior probability estimated using the sample's margin.

TABLE IV
TRAINING TIMES FOR THE PROJECTIVE METHODS ON GTSC

Features	raw	LDA	SRLP	INNLP
I	none	∼3s	∼1day	∼2h
PI	none	∼11s	∼1day	∼5.5h
HOG1	none	∼6s	∼1day	∼4h
HOG2	none	∼6s	∼1day	∼4h
HOG3	none	∼17s	∼1day	∼11h

VI. CLASSIFICATION EXPERIMENTS

In the subsequent sections we evaluate different combinations of various features and feature representations combined with different classifiers. As will be seen, many of these combinations provide comparable quantitative results while greatly differing in training and testing times. Therefore we first provide a comparison of timings for the different training and evaluation pipelines.

A. Training time

All provided times are rough estimates based on standard C/C++ SVM libraries and unoptimized Matlab codes, they serve as reference. The classification experiments were carried out on a Quad-Core AMD Opteron 8360 SE machine with 64GBytes of RAM.

Features computation is the first stage of our pipeline. Other than using the image intensities directly, in this paper we experiment with different variants of HOG features (PI, HOGx). We work with cropped and normalized image patches of 28×28 grayscale pixels, and for each of them the feature computation takes less than 5ms with our Matlab scripts. The features computation step has to be performed both for training and testing.

As a second stage we generate different feature projections. The training times for the projections on the GTSC are summarized in Table IV. While LDA operates in the range of a few seconds, INNLP and SRLP are considerably slower in training the projections, depending on the dimensionality of the raw features. INNLP is up to 12 times faster than SRLP.

When training the SVM based classifiers, the feature dimension plays an important role (see Table V). While training IKSVM, POLYSVM or RBFSVM over thousand dimensional features (I, PI, HOGx) is very time consuming

TABLE V

PERFORMANCE AND RUNNING TIMES OF DIFFERENT CLASSIFIERS REPORTED ON GTSC (TIMES FOR THE CLASSIFIER ONLY, DISREGARDING FEATURES COMPUTATION AND PROJECTION).

Feature	Projection	Classifier	Accuracy	Training time	Testing time
I,PI,HOGs	INNLP	INNC ($K = 62$)	$\mathbf{98.53}\%$	none	$\sim$ 10m
		INNC ($K = 14$)	98.27%	none	$\sim$ 3m
	SRLP	SRC	98.50%	none	$\sim$ 9h
PI,HOGs	LDA	INNC	98.33%	none	$\sim$ 3m
		SRC	98.30%	none	$\sim$ 3h
	SRLP	RBFSVM	98.32%	$\sim$ 5h	$\sim$ 4m
PI	none	IKSVM	97.14%	$\sim$ 1day	$\sim$ 4m
	SRLP	POLYSVM	96.84%	$\sim$ 0.3h	$\sim$ 0.5m
HOG2	INNLP	LSVM	96.51%	$\sim$ 1m	$\sim$ 1s
		RBFSVM	97.08%	$\sim$ 0.5h	$\sim$ 3m
	SRLP	IKSVM	96.81%	$\sim$ 0.5h	$\sim$ 0.5m
		INNC ($K = 14$)	97.57%	none	$\sim$ 1m
		INNC ($K = 62$)	97.65%	none	$\sim$ 3.5m
		SRC	97.53%	none	$\sim$ 1.5h
	LDA	NN	96.97%	none	$\sim$ 5s

on large datasets like GTSC, it is orders of magnitude faster for their lower dimensional projections (LDA, SRLP, INNLP). Moreover, the embeddings usually lead to better classification performance when compared with using the raw features (see Table VI). Training LSVM is fastest, followed by POLYSVM, IKSVM and finally RBFSVM. Based on their nature, least squares classifiers do not require any additional training time.

B. Testing time

Test times decrease when using the lower dimensional feature projections. In our experiments, the fastest classifier was LSVM, followed by NN and IKSVM. SRC is not applicable for real-time applications as its testing time is 10 to 1000 times slower compared to SVM classifiers, due to the optimization involved. The INNC classifier combined with INNLP projections provide a reasonable testing speed, which lies in between LSVM (10 times faster) and SRC (50 times slower).

C. BTSC results

Table VI depicts the performance achieved for different classifier and raw or projected I and PI features. The LDA projections are 61-dimensional (number of classes minus 1) while for SRLP and INNLP the dimensionality of the embedding is set to 50. The regularization parameter is set to 1 for LDA, 0.1 for SRLP and 0.01 for INNLP.

Operating directly on the image intensities provides poor results. The best 4 combinations (shown with bold) of classifier and features are within a range of 0.28 percentage points. In average the SRLP and INNLP projections over PI provide the best results. The top scores are reached by using the SRC classifier, but INNC and even LSVM are very close. Generally we found that the used classifier does not impact the quality drastically while using these features (< 1 percentage point). The decision of which combination to use is left for the reader and depends on the requirements of fast training (e.g. LDA PI + INNC) or fast evaluation (e.g. SRLP PI + LSVM).

D. GTSC results

Table VII reports the classification results achieved for different settings. The LDA projections are 42-dimensional (number of classes minus 1) while for SRLP and INNLP we fix the dimensionality of the embedding at 100. The regularization parameter is set to 1 for LDA, 0.1 for SRLP, and 0.01 for INNLP. Besides I and PI features, this datasets provides the precalculated HOGx features. In all experiments HOG2 produces better results than HOG1 and HOG3 which are therefore omitted in the table. With an average classification rate of only 91.49% (across classification methods) we further removed projected I features from the table to increase readability.

We do not evaluate raw features, since (based on table VI) we do not expect good results and the training on a large dataset such as GTSC would be computationally demanding.

Similar to the result in the previous section, the best 4 results are in a range of 0.14 percent points and the SRLP and INNLP projections provide better results than LDA. The provided HOG2 features perform better than the other features.

E. The best classification results

Merging the features usually helps improving the classification rates. In Table VIII we report the performance obtained by merging features projected via LDA, SRLP, and INNLP, tested over GTSC. All these results are better than any individual based result, using a single projected raw feature.

TABLE VI

CLASSIFICATION RATE (%) RESULTS ON BTSC

	I	PI	LDA I	LDA PI	SRLP I	SRLP PI	INNLP I	INNLP PI	Avg.
NN	77.15	88.52	92.46	97.47	93.06	97.20	93.61	97.08	92.07
SRC	91.00	95.42	94.67	97.34	95.15	**97.55**	94.80	**97.83**	**95.60**
INNC	86.15	94.79	94.71	**97.67**	95.15	97.47	94.83	97.55	94.79
LSVM	90.69	96.68	91.48	96.73	91.83	97.32	91.87	97.24	94.23
IKSVM	91.36	**97.79**	92.22	96.80	91.83	97.08	92.86	97.08	94.63
POLYSVM	89.94	96.61	91.79	96.45	92.85	97.00	93.13	96.96	94.34
RBFSVM	90.41	96.57	91.79	97.08	92.90	97.43	93.37	97.26	94.60
Avg.	88.10	95.20	92.87	96.64	93.25	**97.29**	93.50	**97.29**	

TABLE VII

CLASSIFICATION RATE (%) RESULTS ON GTSC

	LDA PI	LDA HOG2	SRLP PI	SRLP HOG2	INNLP PI	INNLP HOG2	Avg.
NN	95.83	96.97	95.26	96.56	93.28	96.47	94.05
SRC	96.33	97.20	96.69	**97.51**	96.70	**97.53**	**95.87**
INNC	96.52	97.47	96.54	**97.57**	95.15	**97.65**	95.86
LSVM	95.16	96.42	96.03	96.37	96.15	96.51	93.31
IKSVM	95.08	96.22	96.40	96.81	96.43	96.73	93.91
POLYSVM	95.36	96.58	96.84	96.93	96.68	96.84	95.22
RBFSVM	95.34	96.65	96.82	96.95	96.85	97.08	95.30
Avg.	95.66	96.79	96.37	**96.96**	95.89	**96.97**	

TABLE VIII

CLASSIFICATION RATE (%) RESULTS ON MERGED I,PI,HOG1,2,3 PROJECTIONS OVER GTSC

Classifier	LDA	SRLP	INNLP
NN	97.65	97.57	97.60
SRC	**98.28**	**98.50**	98.31
INNC	98.20	98.25	**98.53**
LSVM	97.82	97.47	97.25
IKSVM	97.78	98.15	98.13
POLYSVM	97.91	98.16	98.02
RBFSVM	97.94	98.16	98.05

TABLE IX

BEST CLASSIFICATION RESULTS ON GTSC

CR (%)	Team	Method
99.46	IDSIA [26]	Committee of CNNs
98.84	INI-RTCV [4]	Human Performance
98.53	ours	INNC+INNLP(I,PI,HOGs)
98.50	ours	SRC+SRLP(I,PI,HOGs)
98.31	sermanet [27]	Multi-scale CNNs
98.19	[28]	SRGE+HOG2
96.14	CAOR [4]	Random Forests
95.68	INI-RTCV [4]	LDA on HOG2

The same effect is found on BTSC. Concatenating LDA projections of I and of PI features boosts the performance up to 97.78% for NN and 98.20% for INNC. With SRLP projected features, INNC peaks at 98.32%.

Table IX gives an overview of the best classification rates reported so far on GTSC. With INNC and INNLP projections we get 98.53%. Our second best setup, uses SRC and is 50 times slower (see Table V). The committee of Convolutional Neural Networks (CNN) [26] gets 99.46%, significantly better than the human performance 98.84% and is therefore around 1 percentage point better than our proposed method. However, this quality improvement comes to the price of 37 hours of training on dedicated hardware (using 4 GPUs), while our INNLP+INNC method needs less time, as only the feature projections have to be calculated. At test time our method is also more lightweight.

Please note that we improve over the results of a multi-scale convolutional neural network [27] or other reported methods on this dataset such as Sparse Representation-based Graph Embedding (SRGE) [28] which extends the SRLP technique to include also the discriminative information between different classes – 98.19%. Our top results come close to the human performance, and our methods improve in training and/or testing time over the other top approaches.

VII. RECOGNITION EXPERIMENTS

In applications detection and classification are expected to run jointly. The detected signs are fed into a classifier. The previous classification experiments assumed perfectly aligned detections. Now we evaluate the full pipeline, where errors in detection affects classification.

As shown in the detection section IV, and in Figure 3, our detectors are quite accurate and reach 90% recall even with an overlap criterion of 80%. We validate the classification performance on the detections spotted on BTSC dataset (this time used for detection), from our baseline detectors

ChnFtrs and VJ+HSI. For each detector we fix the operating point by setting the score threshold to 0. This gives an aggregated detection rate (across all three superclasses) of 95.06% for ChnFtrs and 89.77% for VJ+HSI. For each detection we extract PI features, compute the LDA projection, and apply the classifiers.

On BTSC there are 7 classes for the mandatory superclass, 11 for the prohibitive and 19 for the danger super-class (covering 37 classes in total). We use classifiers trained on BTSC with all 62 classes.

The best performing classifier is INNC which provides - averaged over all classes - 95.73% correct classification rates when combined with the output of ChnFtrs detectors and 96.10% with VJ+HSI.

These rates are obtained by using the bounding boxes coming from the detector directly where the predicted bounding box might not be perfectly aligned with the traffic signs. They lie only 2 percentage points below the classification experiments when using the precise testing annotations from BTSC (Table VI).

The traffic sign recognition rate of the ChnFtrs and INNC + LDA PI system is 91.00% combining the detection rates of 95.06% and the subsequent classification rate of 95.73%. This means that out of 1945 groundtruth traffic signs, 1849 are detected, and 1770 are both detected and correctly labeled (and 175 are misslabeled as background or wrong class). For comparison VJ+HSI with LDA PI + INNC achieve only 86.27% recognition rate.

Please note that these results are obtained on a single frame. When fusing detections over multiple frames, the success rate will further increase.

VIII. Conclusion

In the past traffic sign recognition has raised the development of methods that exploit the specific shapes and colors of traffic signs. We have shown that on the most recent large datasets for detection and classification, out of the box methods reach top performance.

Most certainly adding traffic sign specific information, and temporal information would allow to further improve quality. However, we already notice that existing datasets reach saturation (most results are in the range $95\% \sim 99\%$ of the perfect solution). We believe it is thus time to move towards even larger datasets, and to include subsets recorded under adverse weather conditions (rain, fog, night, snow). Although such a dataset does not yet exist publicly, we hope its creation will further push forward the topic of traffic sign recognition.

Acknowledgments: This paper was suported by the ERC Advanced Grand VarCity and the PARIS project (IWT-SBO-Nr. 110067).

References

[1] J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Z. Kolter, D. Langer, O. Pink, V. Pratt, M. Sokolsky, G. Stanek, D. Stavens, A. Teichman, M. Werling, and S. Thrun, "Towards fully autonomous driving: Systems and algorithms," in *Intelligent Vehicles Symposium (IV)*. IEEE, 2011.

[2] R. Timofte, V. A. Prisacariu, L. J. Van Gool, and I. Reid, "Chapter 3.5: Combining traffic sign detection with 3d tracking towards better driver assistance," in *Emerging Topics in Computer Vision and its Applications*, C. H. Chen, Ed. World Scientific Publishing, September 2011.

[3] R. Timofte, K. Zimmermann, and L. Van Gool, "Multi-view traffic sign detection, recognition, and 3d localisation," *Machine Vision and Applications*, December 2011.

[4] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition," *Neural Networks*, no. 0, pp. –, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0893608012000457

[5] A. Møgelmose, M. M. Trivedi, and T. B. Moeslund, "Vision based traffic sign detection and analysis for intelligent driver assistance systems: Perspectives and survey," *IEEE Intelligent Transportation Systems Transactions and Magazine. Special Issue on MLFTSR*, 2012.

[6] R. Benenson, M. Mathias, T. Tuytelaars, and L. Van Gool, "Seeking the strongest rigid detector," in *CVPR*, 2013.

[7] J. Wright, A. Y. Yang, A. Ganesh, S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Transactions on Pattern Analysis and Machine Learning*, vol. 31, no. 2, February 2009.

[8] R. Timofte and L. Van Gool, "Iterative nearest neighbors for classification and dimensionality reduction," in *CVPR*, 2012.

[9] S. Houben, J. Stallkamp, J. Salmen, M. Schlipsing, and C. Igel, "Detection of traffic signs in real-world images: The German Traffic Sign Detection Benchmark," in *International Joint Conference on Neural Networks (submitted)*, 2013.

[10] R. Timofte and L. Van Gool, "Sparse representation based projections," in *BMVC*, 2011.

[11] P. Dollár, Z. Tu, P. Perona, and S. Belongie, "Integral channel features," in *BMVC*, 2009.

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

[13] P. Dollár, R. Appel, and W. Kienzle, "Crosstalk cascades for frame-rate pedestrian detection," in *ECCV*, 2012.

[14] R. Benenson, M. Mathias, R. Timofte, and L. Van Gool, "Pedestrian detection at 100 frames per second," in *CVPR*, 2012.

[15] S. Maji and J. Malik, "Fast and accurate digit classification," EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2009-159, Nov 2009. [Online]. Available: http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-159.html

[16] R. Fisher, "The statistical utilization of multiple measurements," *Annals of Eugenics*, vol. 8, pp. 376–386, 1938.

[17] A. Martinez and A. Kak, "PCA versus LDA," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 23, no. 2, pp. 228–233, 2001.

[18] X. He and P. Niyogi, "Locality preserving projections," in *NIPS*, 2003.

[19] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering," in *Advances in Neural Information Processing Systems 14*. MIT Press, 2001, pp. 585–591.

[20] D. Cai, X. He, and J. Han, "Srda: An efficient algorithm for large-scale discriminant analysis," *IEEE Trans. on Knowl. and Data Eng.*, vol. 20, no. 1, pp. 1–12, 2008.

[21] H. Lee, A. Battle, R. Raina, and A. Y. Ng, "Efficient sparse coding algorithms," in *NIPS*, 2006.

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

[23] S. Maji, A. C. Berg, and J. Malik, "Classification using intersection kernel support vector machines is efficient," in *CVPR*, 2008.

[24] C.-C. Chang and C.-J. Lin, *LIBSVM: a library for support vector machines*, 2001, software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[25] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "LIBLINEAR: A library for large linear classification," *J. Mach. Learn. Res.*, vol. 9, pp. 1871–1874, 2008.

[26] D. Ciresan, U. Meier, J. Masci, and J. Schmidhuber, "Multi-column deep neural network for traffic sign classification," *Neural Networks*, vol. 32, pp. 333–338, 2012.

[27] P. Sermanet and Y. LeCun, "Traffic sign recognition with multi-scale convolutional networks," in *IJCNN*, 2011, pp. 2809–2813.

[28] K. Lu, Z. Ding, and S. Ge, "Sparse-representation-based graph embedding for traffic sign recognition," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 4, pp. 1515–1524, 2012.